Multilinguality and Inclusive Language Technologies for Low-Resource Languages

Inaugural-Dissertation
zur Erlangung des Doktorgrades der Philosophie
an der Ludwig–Maximilians–Universität München

vorgelegt von
Haotian Ye
aus Shaoxing

2025

Erstgutachter: Prof. Dr. Hinrich Schütze
Zweitgutachter: Prof. Dr. Julian Schröter
Drittgutachter: Prof. Dr. Annette Hautli-Janisz

Tag der Einreichung: 4. März 2025
Tag der mündlichen Prüfung: 7. Juli 2025

Declaration on Writing Aids

ChatGPT has been used as a tool to assist in the composing of this dissertation. While some aspects of its usage cover all chapters of the dissertation, others are limited to certain chapters. Below, the specific usage cases of ChatGPT as a writing aid are summarized.

Writing refinement For all chapters, ChatGPT is used to refine the grammar and writing style. This usage includes identifying and correcting grammatical errors, refining phrasing to avoid unnatural expressions, and improving word choices, including the selection of more precise and descriptive terminologies where appropriate. Additionally, as a final step in preparing the initial draft, ChatGPT is used to inspect the text thoroughly for potential grammatical and spelling errors. All refinements suggested by ChatGPT with the goal of writing refinement are carefully reviewed to ensure they are accurate and align with the original meaning and intent.

Literature suggestions On rare occasions, a specialized GPT (Scholar GPT) is used to obtain suggestions for relevant literature in specific research domains, notably in Chapter 2. The recommended publications are subsequently checked manually to verify their correctness and relevance.

Mathematical equations Some of the mathematical equations presented in Chapter 2 are reformulated representations of the original equations from existing research. To convert the original mathematical equations into LaTeX source code, ChatGPT is used to analyze these equations, which are provided in image format. As with other applications of ChatGPT, the generated mathematical equations undergo manual inspection to ensure their correctness.

Abstract

With over 7000 languages worldwide, the development of language technologies for low-resource languages, which constitute a significant portion of the world's languages, remains a critical but understudied area in computational linguistics and natural language processing (NLP). This dissertation addresses the multifaceted challenges of multilingual NLP for low-resource and marginalized languages by unifying efforts in dataset creation, model adaptation, cross-lingual transfer learning, and a novel approach to understanding language similarity based on the alignment of linguistic concepts across languages.

The limited availability of evaluation datasets for a vast majority of the world's languages presents a constraint on the advancement of NLP capabilities for languages beyond a handful of high-resource ones, such as English and German. Compounding this issue, pre-trained language models (PLMs) and large language models (LLMs) typically support a maximum of only around 100 languages, leaving many low-resource languages without coverage and perpetuating unequal technological development. To address these gaps, we leverage tools that are more steadily available for a broader range of languages, such as static word embeddings, to extend the capabilities of PLMs to low-resource languages so far without coverage.

In this dissertation, we address several of the aforementioned challenges in multilingual NLP and make the following contributions. First, we develop Taxi1500, a massively multilingual dataset for text classification utilizing a parallel corpus of Bible texts, expanding the evaluation possibility to more than 1500 languages. By supporting large-scale multilingual evaluation, Taxi1500 aims to democratize access to NLP technologies and increase the inclusiveness across underrepresented linguistic communities.

Second, using the same parallel corpus, we conceive a novel framework for quantifying language similarity through cross-lingual conceptual alignment. The introduced similarity metric complements existing genealogical and typological measures by capturing how concepts are realized and aligned across languages, offering new insights into linguistic relationships and cultural diversity beyond lexical and geographical proximity.

We further address the adaptation of PLMs to low-resource languages through the MoSECroT framework, which stitches static word embeddings for low-resource languages with a PLM that has no prior knowledge of these languages, thereby enabling effective zero-shot transfer. Additionally, we incorporate language and script embeddings during the pre-training stage of a multilingual PLM for over 500 languages, demonstrating the positive impact of explicit language and script information on cross-lingual transfer performance.

Finally, we tackle the pressing real-world issue of online hate speech, particularly in marginalized linguistic communities, by curating culturally and contextually sensitive hate speech datasets and applying a privacy-preserving federated learning framework. This distributed approach ensures user privacy while also effectively classifying hate speech in diverse linguistic settings.

Zusammenfassung

Mit über 7000 Sprachen weltweit bleibt die Entwicklung von Sprachtechnologien für ressourcenarme Sprachen - die einen erheblichen Teil der weltweiten Sprachvielfalt ausmachen - ein kritischer, jedoch noch unzureichend erforschter Bereich in der Computerlinguistik und der Verarbeitung natürlicher Sprache (NLP). Diese Dissertation befasst sich mit den vielfältigen Herausforderungen der mehrsprachigen NLP für ressourcenarme und marginalisierte Sprachen, indem sie Ansätze zur Datensatzerstellung, Modellanpassung, sprachübergreifendem Transferlernen und einem neuartigen Ansatz zum Verständnis sprachlicher Ähnlichkeiten auf der Grundlage konzeptueller Ausrichtung zwischen Sprachen vereint.

Die begrenzte Verfügbarkeit von Evaluierungsdatensätzen für eine Mehrheit der Weltsprachen stellt eine Einschränkung für die Weiterentwicklung der NLP-Funktionen für Sprachen jenseits einer Handvoll ressourcenreicher Sprachen wie Englisch und Deutsch dar. Erschwerend kommt hinzu, dass vortrainierte Sprachmodelle (PLMs) und große Sprachmodelle (LLMs) in der Regel maximal nur rund 100 Sprachen unterstützen und viele ressourcenarme Sprachen unberücksichtigt lassen, was zu einer ungleichen technologischen Entwicklung führt. Um diese Lücken zu schließen, nutzen wir Tools, die für eine breitere Palette von Sprachen verfügbar sind, wie z.B. statische Wort-Embeddings, um die Fähigkeiten von PLMs auf ressourcenarme Sprachen auszudehnen, die bisher nicht abgedeckt wurden.

In dieser Dissertation gehen wir auf mehrere der oben genannten Herausforderungen in der mehrsprachigen NLP ein und leisten die folgenden Beiträge. Zunächst entwickeln wir Taxi1500, einen massiv mehrsprachigen Datensatz zur Textklassifizierung, der ein paralleles Korpus von Bibeltexten verwendet und die Evaluierungsmöglichkeiten auf über 1500 Sprachen erweitert. Durch die Unterstützung groß angelegter mehrsprachiger Evaluierungen zielt Taxi1500 darauf ab, den Zugang zu NLP-Technologien zu demokratisieren und die Inklusivität in unterrepräsentierten Sprachgemeinschaften zu erhöhen.

Zweitens konzipieren wir unter Verwendung desselben parallelen Korpus ein neuartiges Framework zur Quantifizierung sprachlicher Ähnlichkeiten durch sprachübergreifende konzeptionelle Ausrichtung. Die eingeführte Ähnlichkeitsmetrik ergänzt bestehende genealogische und typologische Maßnahmen, indem sie erfasst, wie Konzepte sprachübergreifend realisiert und ausgerichtet werden, und bietet neue Einblicke in linguistische Ähnlichkeiten und kulturelle Vielfalt jenseits der lexikalischen und geografischen Nähe.

Wir befassen uns außerdem mit der Anpassung von PLMs an ressourcenarme Sprachen durch das MoSECroT Framework, das statische Wort-Embeddings für ressourcenarme Sprachen mit einem PLM verknüpft, das keine Vorkenntnisse dieser Sprachen hat, und so eine effektive Zero-Shot-Transfer ermöglicht. Darüber hinaus integrieren wir Sprach- und Skript-Embeddings während der Pre-trainingsphase eines mehrsprachigen PLM für über 500 Sprachen und demonstrieren damit die positive Wirkung expliziter

Sprach- und Skriptinformationen auf die sprachübergreifende Transferleistung.

Schließlich befassen wir uns mit dem praktischen Problem der Online-Hassrede, insbesondere in marginalisierten Sprachgemeinschaften, indem wir kulturell und kontextbezogene Hassrede-Datensätze kuratieren und ein datenschutzfreundliches Federated Learning (FL) Framework anwenden. Dieser verteilte Ansatz gewährleistet die Privatsphäre der Benutzer und klassifiziert Hassrede gleichzeitig effektiv in unterschiedlichen sprachlichen Umgebungen.

Contents

Chapter 1

Introduction

1.1 Motivation

1.1.1 Language Inequality

In today's world, a vast amount of information is produced daily and influences nearly every facet of our lives, from online information retrieval to translating a foreign language. Processing this immense data volume has become both a challenge and a driver of innovation, propelling significant advancements in natural language processing (NLP), including great improvements in applications like machine translation (MT) and sentiment analysis (Zhang et al., 2018a; Dabre et al., 2021; Chronopoulou et al., 2023). Since its introduction, the Transformer architecture (Vaswani et al., 2017) has marked a paradigm shift in tackling NLP problems and has been widely adopted as the de facto go-to solution. Building on the Transformer, pre-trained language models (LMs), such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), have achieved impressive language capabilities by training on vast datasets. Some of these models, pre-trained on data from multiple languages, show strong performance not only on English tasks but also on tasks in various other languages. However, despite the existence of over 7000 languages globally (Joshi et al., 2020), most multilingual LMs cover at most around 100 languages (Devlin et al., 2019; Conneau et al., 2020; Xue et al., 2021), leaving many low-resource and endangered languages underrepresented or entirely excluded from digital tools.

The recent development of large language models (LLMs) has achieved state-of-the-art performance across different fields of NLP (Brown et al., 2020; Ouyang et al., 2022; Webb et al., 2023). However, LLMs generally support only a limited set of languages, similar to previous LMs (Scao et al., 2022; Touvron et al., 2023b). Additionally, studies indicate that LLMs tend to use English as the preferred internal language (Etxaniz et al., 2024; Wendler et al., 2024), thus unable to fully leverage their multilinguality.

This huge gap is a manifestation of language inequality in NLP technologies, as

speakers of minority languages are often restricted in accessing tools essential for information processing and digital well-being, such as hate speech detection tools. The primary driver of this inequality is data scarcity. While vast text corpora exist for a few high-resource languages like English and German, similar data, which is crucial for the training of NLP technologies, remain critically limited or nonexistent for most of the world's languages. It is thus imperative to make both current and emerging NLP technologies inclusive, supporting as many languages as possible, especially low-resource and endangered ones.

1.1.2 Cross-Lingual Transfer

The advent of pre-trained language models (PLMs) like BERT (Devlin et al., 2019) has led in a new era for NLP, where substantial performance gains are achieved through large-scale pre-training on vast amounts of unlabeled text data. The pre-training phase, which contributes to the impressive language capabilities of such models, relies heavily on the availability of extensive unlabeled data, which explains why these models initially emerged for high-resource languages where data is abundant (Antoun et al., 2020; Chan et al., 2020; Le et al., 2020). For low-resource languages, however, limited data availability presents serious challenges for building robust NLP systems, as there is often insufficient data to pre-train language models specifically for these languages. Compounding this issue are growing concerns about the environmental impact caused by large-scale pre-training, a process that demands significant time and computational resources (Gupta et al., 2022; Patel et al., 2023). Consequently, developing efficient strategies to use existing resources is essential both for the reduction of carbon footprint and the development of reliable NLP technologies for low-resource languages.

To address these challenges, cross-lingual transfer learning is a widely adopted approach for enhancing the performance of NLP systems in low-resource languages, which are often underserved by current language technologies. This approach involves leveraging data or pre-trained language models available for one language, typically a high-resource one like English, to address NLP tasks in a target language with limited resources.

In the past, cross-lingual transfer has been achieved through parallel multilingual word embeddings, where closely related words across languages are represented by similar word vectors in a shared vector space (Mikolov et al., 2013b; Gouws and Søgaard, 2015; Vulić and Korhonen, 2016). More recently, multilingual PLMs are pre-trained on multilingual corpora covering over 100 languages, sharing model parameters to enable cross-lingual transfer through zero-shot and few-shot learning. Such models can achieve strong performance on target languages by fine-tuning only on a few samples in a high-resource language, such as English (Hu et al., 2020; Gao et al., 2021; Lin et al., 2022). However, the cross-lingual transfer capabilities of PLMs remain constrained by factors such as data availability for different target languages and linguistic similarity

to the source language (Lauscher et al., 2020). To extend NLP capabilities across more languages, innovative methods must be studied to effectively extend existing tools like PLMs to languages that remain unseen or underrepresented in training data.

An additional important element in cross-lingual transfer is language neutrality, which ensures that similar words across different languages are represented by comparable embeddings. Research has explored the importance of language neutrality, including its role in effective cross-lingual transfer learning (Libovický et al., 2020; Chang et al., 2022).

By transferring knowledge from a high-resource source language to a low-resource target language, cross-lingual transfer learning makes it possible to perform NLP tasks effectively on low-resource languages with minimal or no labeled data, reducing language barriers and extending more inclusive NLP capabilities to a much broader range of communities.

1.1.3 Conceptual Diversity

Traditionally, languages have been classified according to phylogenetic typology, with similarities between languages primarily assessed through vocabulary, i.e., lexical similarity, and morphology (whether a language is synthetic or analytic). However, languages differ not only in vocabulary and morphology but also in how they conceptualize meanings, encoding ideas in ways that can vary. For example, the Yoruba word *irun* means both "hair" and "wool" in English, where English uses distinct terms for these concepts. In other words, Yoruba uses a single concept to describe both human and animal hair, whereas English distinguishes them with separate words. This phenomenon of conceptualization, or how languages conceptualize ideas differently, has been studied through the lenses of cross-lingual polysemy, where one word is associated with multiple meanings (Perrin, 2010; List et al., 2013), and colexification, where languages lexify word senses identically (François, 2008).

Similarities in conceptualization can be observed beyond the lexical or genealogical relatedness of languages, which is defined by the phylogenetic typology. For instance, Tagalog, a language spoken in the Philippines, exhibits conceptual similarities to some European languages, notably Spanish, which can be partially attributed to the Spanish colonization of the archipelago. One manifestation of such similarity is seen in the Tagalog words *dila* and *wika*, both of which can mean "language" or "tongue", similar to the Spanish word *lengua*, which also carries both meanings. Similarly, Plateau Malagasy, an Austronesian language spoken in Madagascar, shows conceptual proximity not only with Hawaiian, a geographically distant Austronesian relative, but also with geographically adjacent, yet topologically different languages like Mwani and Koti (Liu et al., 2023b). Such patterns of conceptual similarity among geographically and topologically distant languages suggest that similar words across languages may not always convey similar meanings due to conceptual divergences. This variability presents

a challenge in generalizing language technologies, particularly when adapting them to conceptually dissimilar languages. At the same time, these patterns indicate that languages may also be grouped by their conceptual relatedness, offering an alternative measure to traditional systems of linguistic classification.

The conceptual overlap (e.g., "tongue" and "language" in Tagalog and Spanish) and areas of divergence (e.g., "hair" and "wool" in Yoruba and English) motivate a deeper exploration of the conceptual language similarity. Examining these similarities and differences offers us valuable insights and deepens our understanding of not only linguistic diversity but also the underlying cognitive and cultural factors that shape language, with potential implications for improving NLP technologies.

1.1.4 Culture- and Context-Aware NLP

Section 1.1.3 illustrates how differences in conceptualizations across languages can reveal cultural and historical insights (such as those found in the Philippine languages). Such conceptual variations are not merely the result of linguistic differences, they also highlight subtle but important nuances in how people use and interpret languages within their cultural contexts. Despite considerable advances in their multilingual capabilities, NLP techniques still struggle to capture these cultural dimensions accurately, as both NLP models and datasets typically aim to cover a broad range of languages, yet fail to account for the intricate cultural variations that shape language use. This oversight risks producing inaccurate interpretations, as differences in language usage, including within the same language across regions and cultures, can lead to misinterpretations that impact specific applications.

Awareness of cultural and contextual nuances is particularly relevant in hate speech detection, a task that is highly sensitive to cultural and ethnic variations and especially important for communities that speak low-resource languages and have limited access to NLP tools. To date, most hate speech detection datasets are organized at the language level, with little attention to regional or cultural variations within the same language. This approach has the drawback of overlooking critical cultural information, particularly for languages covering large geographical areas that encompass rich cultural diversity. Recent studies, such as those by Pawar et al. (2024) and Tonneau et al. (2024a), have examined the level of cultural awareness in NLP models and datasets. Tonneau et al. (2024a) show that hate speech datasets across languages with wide geographical coverages often overrepresent certain cultural contexts while largely overlooking others, leading to classification errors for target groups whose cultural nuances are underrepresented. To address this issue, some initiatives have focused on building region-specific datasets to enrich the cultural diversity within languages (Arango Monnar et al., 2022; Tonneau et al., 2024b) in an attempt to improve generalizability to less represented cultural contexts.

Beyond cultural and regional considerations, it is equally crucial to examine the

specific context in which the data is collected and annotated. In hate speech detection, various studies reveal a discrepancy between data annotators and target groups directly affected by hate speech (Davidson et al., 2019; Sap et al., 2019). This often arises from annotators' limited familiarity with the dialectal and cultural variations of the target groups and a lack of diversity among data collectors, and can lead to insensitivity toward nuances in the annotated data. For example, texts written in the African American English (AAE) dialect are more commonly labeled as offensive, a bias that can propagate through NLP systems trained on such datasets (Sap et al., 2019). Further studies demonstrate that factors such as annotators' identity and background play an important role in determining severity ratings in toxicity datasets (Goyal et al., 2022b; Mostafazadeh Davani et al., 2022; Sap et al., 2022). To mitigate such biases, studies like Maronikolakis et al. (2022) and Shekhar et al. (2022) propose creating abusive language detection datasets in low-resource languages by directly involving affected communities in the data collection process. This approach effectively increases diversity among data collectors and aligns them with the specific contexts of affected target groups, which are essential steps for reducing bias in abusive and toxic language detection datasets.

The shift toward culturally and contextually inclusive NLP resources represents a crucial step in adapting NLP applications to reflect the rich linguistic diversity across different regions. As NLP technologies advance, prioritizing cultural and contextual awareness in data collection will be increasingly vital for the development of fair and accurate NLP systems for global communities.

1.1.5 Scalable and Privacy-Preserving NLP

Deep learning methods, including the training of current state-of-the-art LMs for NLP, are extremely data-hungry. These models have thus far relied on vast amounts of public data to achieve high performance. However, studies have shown that publicly available data may contain personally identifiable information (PII) and potentially copyright-protected contents, which may inadvertently be memorized by the models during training. Carlini et al. (2021) show that larger models are more susceptible to such memorization than smaller ones, raising concerns given the current trend toward ever-larger LMs. This poses dangers including privacy attacks like data extraction, where personal information can be retrieved from pre-trained models (Carlini et al., 2023; Ippolito et al., 2023). Moreover, verbatim memorization of entire text chunks from the training data may lead to unintentional copyright infringements (Karamolegkou et al., 2023). This issue is exacerbated in domains where personal or private information is especially sensitive, such as medicine and finance. Additionally, Villalobos et al. (2022) suggest that public data may be depleted by as early as 2026, raising serious questions about the viability of training ever-larger language models on increasingly massive datasets.

To address the challenges of limited public data and the risks of exposing sensitive or copyright-protected information during training, leveraging local data stored on end

devices in a privacy-preserving manner becomes especially relevant. Learning approaches such as federated learning (FL) (McMahan et al., 2017) offer a promising solution to this problem. Instead of gathering data from users to train a model on a remote server, FL retains private data on local devices and trains models directly on users' devices. Only the updates to the local models are collected and used to update the central model in an aggregated manner. Because FL eliminates the need to transfer and store private user data on remote servers, it ensures the data privacy of the users. Due to its privacy-preserving nature, FL has been used in areas where data privacy plays a crucial role, such as medicine (Sheller et al., 2020) and finance (Byrd and Polychroniadou, 2020).

Despite its promising aspects of enabling distributed training while preserving user data, FL still faces certain technical challenges, such as possible information leakage (Geiping et al., 2020) and vulnerability to membership inference attacks (Truex et al., 2021). Such challenges are typically addressed using techniques like differential privacy (Dwork et al., 2016; Kairouz et al., 2021), although the trade-off between the amount of noise added to increase privacy and the model accuracy remains an active area of research. As a distributed training method, FL may also encounter issues with heterogeneous or non-independent and identically distributed (non-IID) user data, which can slow the convergence of the central model (Karimireddy et al., 2020; Li et al., 2020). To alleviate this and increase the customizability of individual local models, approaches to personalize models on the participating devices have shown effectiveness in aligning models with user-specific needs (Arivazhagan et al., 2019; Bui et al., 2019).

1.2 Research Questions

While multilingual PLMs and cross-lingual transfer methods have greatly expanded the reach of NLP across languages (Section 1.1.2), the vast majority of the world's low-resource languages remain underrepresented due to, among others, limited data availability and their marginalization in favor of higher-resource languages. To promote a more inclusive advancement of NLP, it is essential to develop both cost-effective strategies for creating multilingual evaluation datasets and cross-lingual transfer methods that efficiently leverage resources available to high-resource languages. Additionally, as languages often reflect their speakers' unique cultural backgrounds, conceptual diversity is a crucial factor in how languages represent ideas. Understanding patterns that distinguish languages and quantifying the relatedness of languages based on conceptualizations could effectively enhance our understanding of language similarity. Furthermore, as NLP applications increasingly address tasks involving sensitive content, such as the moderation of online hate speech, it becomes crucial to develop methods that not only perform reliably across languages, especially marginalized ones, but also prioritize user privacy. Together, we propose the following research questions, which we aim to explore throughout the remainder of this dissertation:

i. **Evaluation of low-resource languages**: What approaches can be taken to create massively multilingual datasets that support a wide range of low-resource languages while minimizing data annotation costs? Additionally, how might such datasets impact the performance of current multilingual PLMs?

ii. **Quantifying conceptual diversity across languages**: Given that distinct conceptualization patterns reflect diverse cultural backgrounds across languages (Section 1.1.3), how can these differences be captured and measured quantitatively, potentially through a language similarity metric, to enhance cross-lingual understanding?

iii. **Effective cross-lingual transfer for low-resource languages**: What novel techniques, including architectural modifications to existing multilingual PLMs, can be developed to enhance zero-shot and few-shot transfer for low-resource languages by leveraging resources from high-resource languages?

iv. **Culturally sensitive and privacy-preserving NLP**: Using hate speech detection as a case study, how can NLP models be tailored to effectively identify hateful content in a culturally sensitive manner while prioritizing user privacy for marginalized linguistic groups, particularly in low-resource settings?

1.3 Contributions

We summarize the contributions in this work, which address the research questions identified in Section 1.2 and encompass the following four key areas: expanding NLP support for low-resource languages through datasets and tools, quantifying conceptual language similarity, enhancing cross-lingual transfer to low-resource languages via innovative model architecture modifications, and developing culturally sensitive, privacy-preserving NLP applications.

The first major contribution of this dissertation is the development of *Taxi1500*, a massively multilingual text classification dataset that supports NLP evaluation for over 1500 languages. Leveraging parallel Bible translations, this dataset is created by obtaining crowd-sourced annotations for English Bible verses and projecting the collected labels onto parallel translations of the same verses in over 1500 languages. We showcase the utility of Taxi1500 by evaluating multiple PLMs and LLMs on it, including Glot500 (Imani et al., 2023), Llama 2 (Touvron et al., 2023b), and Mistral (Jiang et al., 2023). Furthermore, we put forward an in-depth analysis of the dataset, categorizing the supported languages into three subgroups based on their representation in popular multilingual PLMs, as well as factors such as language families. Utilizing the same multilingual Bible corpus, we introduce *Conceptualizer*, a framework for cross-lingual concept alignment. The alignment is achieved by constructing a set of predefined concepts and creating a directed bipartite graph between source and target language concepts. We demonstrate

that Conceptualizer achieves high accuracy for concept alignment through the evaluation of selected concepts. Additionally, we introduce the notion of cross-lingual stability as the degree of one-to-one overlap in conceptualizations across languages. Our analysis of the relationship between cross-lingual stability and concreteness of concepts reveals that concrete concepts, such as "bird", are more stable across languages than abstract ones such as "mercy". Based on cross-lingual conceptual alignment, we propose a new conceptual language similarity based on varying conceptual patterns across languages. We show this similarity measure offers a novel perspective on linguistic similarities, complementing traditional genealogical and typological similarities.

Recognizing the challenges of adapting PLMs to low-resource languages, we present *MoSECroT*, a framework designed to address resource constraints and the high computational overhead of PLMs by enabling efficient cross-lingual transfer, particularly for low-resource languages. MoSECroT leverages static word embeddings, which are more readily available for a broader range of low-resource languages, and achieves cross-lingual transfer by aligning these embeddings with those of monolingual PLMs for high-resource languages through model stitching with the help of relative representations. This approach creates a unified embedding space between a high-resource source language and a low-resource target language, allowing the embedding layer of a PLM to be swapped to enable zero-shot transfer. Our evaluation on two text classification tasks demonstrates MoSECroT's potential to extend zero-shot cross-lingual transfer capability to low-resource languages unseen by existing multilingual PLMs.

Focusing further on facilitating cross-lingual transfer in PLMs, we introduce LANGSAMP, a language- and script-aware multilingual pre-training method that increases language neutrality in PLMs. LANGSAMP achieves this by integrating language and script embeddings into the output of Transformer blocks, thus offloading the burden of encoding language-specific information from the token embeddings. We apply LANGSAMP for continual pre-training of XLM-R (Conneau et al., 2020) as a case study and demonstrate that the inclusion of language and script embeddings leads to the model consistently outperforming the baseline without language or script embeddings across various downstream tasks. Through extensive analysis, we additionally observe that the resulting language and script embeddings, as byproducts of pre-training, capture structural and typological features that contain language-specific information and can aid in selecting the optimal source languages for cross-lingual transfer learning.

Finally, we address the urgent issue of online hate speech, which disproportionally affects marginalized communities that often lack the support of NLP tools in their languages. We release *REACT*, a collection of culture-specific hate speech detection datasets covering seven target groups in eight low-resource languages. These datasets are developed by individuals with profound background knowledge of the affected target groups and the cultural contexts in which they appear, which ensures the cultural and contextual relevance of the datasets. Given the sensitive nature of hate speech data, which

raises privacy concerns, we propose a distributed, privacy-oriented training approach using federated learning (FL), complemented by personalization techniques to tailor models to the needs of specific target groups. Within the FL framework, user data is processed locally without being collected and stored in a centralized location. This allows local hate speech filtering while maintaining privacy, and simultaneously catering to the individual needs of each user.

Together, these contributions represent advancements across several aspects. The creation of comprehensive multilingual datasets, such as Taxi1500 and REACT, provides crucial resources for the development of NLP systems for underrepresented languages. Conceptualizer introduces not only a structured framework for discovering concept alignments across diverse languages but also a new method for understanding language similarity, offering a valuable perspective on language relationships. Novel frameworks such as MoSECroT and LANGSAMP allow for efficient cross-lingual transfer, making NLP systems more adaptable to new languages without extensive retraining or resource requirements. Moreover, the integration of privacy-preserving methods, such as FL, promotes more ethically responsible NLP development, ensuring that user privacy is respected and NLP is more tailored to meet the specific needs of diverse communities. Cumulatively, this dissertation presents contributions that push further the boundaries of current low-resource NLP systems, allowing a broader range of communities to benefit from language technologies in a more inclusive and ethical way.

1.4 Outline

In this chapter, we describe the motivations behind this dissertation, outline the core research areas, and summarize the contributions made. The remainder of the dissertation is structured as follows. In Chapter 2, we provide the foundational background information for the works in subsequent chapters. In Chapter 3, we detail the process of creating a multilingual parallel text classification dataset from a large multilingual Bible corpus and evaluating the resulting dataset. In Chapter 4, we examine alignment across conceptualization patterns in different languages, conduct an extensive evaluation of conceptual language similarity, and analyze unique features contributing to conceptual divergence. In Chapter 5, we introduce a novel technique that leverages relative representations to enable cross-lingual transfer learning. In Chapter 6, we propose a new pre-training method, analyze the effectiveness of language and script embeddings, and explore their role in selecting optimal languages for cross-lingual transfer. Finally, in Chapter 7, we describe the steps taken to create culture- and context-aware hate speech datasets and present a privacy-preserving approach using federated learning (FL) to classify hate speech in a local and customizable manner.

Chapter 2

Background

2.1 Machine Learning for NLP

2.1.1 Preliminaries

Based on how models learn from the data, machine learning can be broadly categorized into three paradigms: *supervised learning*, *unsupervised learning*, and *reinforcement learning*. Among these, we focus on supervised learning in this section, which is the most relevant paradigm for natural language processing (NLP).

In supervised machine learning, there are two key components: the **dataset** and the **model**. The dataset consists of input-output pairs (x_i, y_i) where $x_i \in X$ and $y_i \in Y$. Here, X represents the set of all input samples, and Y the set of corresponding labels. Typically, the dataset is divided into three parts: a training set, a development (validation) set, and a test set.

On the other hand, the model usually has a set of parameters θ and a set of hyperparameters. The parameters θ are learned during training and are essential for making the model's predictions, while hyperparameters, such as learning rate and batch size, are set before training and adjusted based on the model's performance on the development set. The model is first trained on the training set, evaluated on the development set, and finally tested on the test set to assess its overall performance. The objective of supervised learning, then, is to learn a function $f : X \to Y$ that maps the set of inputs to the outputs accurately. This is done by iteratively adjusting θ to minimize a **loss function** $L(Y, Y')$, where Y' represents the predicted output for X.

To illustrate this with a neural network (discussed further in Section 2.1.2), at the beginning of the learning process, the parameters θ are typically initialized randomly. During training, the model processes the training data X and outputs a set of predictions Y'. The quality of these predictions is calculated using the loss function L, which quantifies the difference between the predicted outputs Y' and the true labels Y. Using an optimization algorithm like **gradient descent**, the model updates θ to reduce the loss

through a process called **backpropagation**. This training process can then be repeated until **convergence** of the model, at which stage the optimization algorithm has stabilized θ, and further training yields minimal improvements.

2.1.2 Neural Networks

Early history

The concept of neural networks was inspired by the way biological neurons in the human brain communicate with each other: a neuron receives an input, processes it, and passes on information by firing signals to other neurons. Research into learning with artificial neurons began as early as 1943, using symbolic logic operators like *AND* and *OR* (McCulloch and Pitts, 1943). Hebb (1949) expanded upon this by suggesting that artificial neurons activate in unison by drawing a parallel to biological neural connections, and proposed the concept of *cell assemblies*, which are groups of functionally interconnected neurons. This principle, known as Hebbian learning theory, states that neurons form stronger connections through repeated, simultaneous firing, much like the neural structure of the brain. Building on these ideas, Frank Rosenblatt developed the *perceptron* (Rosenblatt, 1958), a single-layer neural network designed to output a weighted sum of inputs and is able to function as a binary classifier. Later, (Minsky and Papert, 1969) conducted a comprehensive analysis of the perceptron and pointed out its limitations, notably its inability to solve the *XOR* problem. This demonstrated that a single-layer perceptron could not address non-linearly separable problems, underlining the need for more complex network structures.

Deep learning

As various studies recognized that the simplicity of single-layer perceptrons prevents them from solving complex, non-linearly separable problems like *XOR*, a possible solution was proposed by adding more layers to the model. Amari (1967) presented a solution that enabled the solving of non-linearly separable problems by using a multilayer perceptron, which was trained using stochastic gradient descent. Later, Rumelhart et al. (1986) popularized backpropagation, a learning procedure that minimizes the network's prediction errors by iteratively adjusting the weights between each layer based on gradients calculated from a loss function. Over the years, further architectural innovations have been applied to the network and greatly expanded neural networks' capabilities for specific tasks. For example, convolutional neural networks (CNN) (LeCun et al., 1998) were applied to recognize characters in documents by capturing shifts in local patterns within two-dimensional shapes, which makes CNNs particularly suited for extracting relevant features from inputs like pixel images. Similarly, Long Short-Term Memory networks (LSTM) (Hochreiter and Schmidhuber, 1997) were developed to

address the vanishing gradient problem, a common issue that arises with deeper networks (Hochreiter, 1991; Bengio et al., 1993), through the use of memory cells and gating mechanisms.

A major breakthrough in deep learning came around the 2010s with works such as Hinton and Salakhutdinov (2006), which applied autoencoders to reduce the dimensionality of image data, facilitating the flow of backpropagation through the model. In another milestone, Krizhevsky et al. (2012) achieved groundbreaking progress in image classification using deep learning on the ImageNet challenge (Russakovsky et al., 2015), significantly outperforming previous methods. This was achieved through the introduction of a deep CNN model that could be efficiently implemented on GPUs. Such progress has demonstrated the potential of deep networks and reignited the interests in deep learning. Some subsequent research has focused on techniques for efficiently training very deep neural networks with up to 1000 layers (Srivastava et al., 2015; He et al., 2016), sparking further innovations in deep learning across fields.

Structure

A neural network for NLP typically consists of three components: an embedding layer, feedforward (or fully connected) layers, and an output layer. Although specialized layers like convolutional neural networks (CNNs) and Long Short-Term Memory (LSTM) layers exist for specific NLP tasks (see above), this part focuses on the three most basic building blocks mentioned.

Embedding layer For words to be processed by the model, they first need to be converted from their categorical form into continuous word vectors. This transformation is carried out by the embedding layer, which is typically the first layer the text data passes through in a neural network. The embedding layer is represented as a matrix $E \in \mathbb{R}^{V \times D}$, where V is the predefined vocabulary size of the embedding layer, and D is the dimensionality of the word vectors. The input text is first divided into smaller units during the **tokenization** process (discussed further in Section 2.1.3), and then looked up and mapped to a corresponding vector in the embedding layer.

Feedforward layer The transformed input tokens, now in the form of dense vectors, are passed through a series of feedforward layers. Each feedforward layer receives vectors from the preceding layer as its inputs, calculates a weighted sum of the inputs, and subsequently applies a non-linear activation function to produce the outputs, which are then fed into the next layer of the network. This process can be expressed using the formula $y = \sigma(Wx + b)$, where x is the input vector, W and b are the weight matrix and bias of the feedforward layer, σ is the non-linear activation function, and y is the output vector. Because each neuron in a feedforward layer is connected to every neuron in the following layer, these layers are also called fully connected layers.

Output layer The output layer is the final layer of a neural network and is responsible for producing the final predictions for the specific task the model is deployed for, based on the outputs from preceding feedforward layers. Therefore, the number of neurons in the output layer is often determined by the concrete type of task and generally matches the number of possible output labels.

Activation functions

Activation functions add non-linearity to the outputs of a layer, enabling the model to learn complex patterns and non-linear decision boundaries that cannot be captured using linear activations alone. This capability is particularly useful in deep networks where non-linear activations are applied across multiple stacked layers. The aforementioned *XOR* problem, for example, can be solved by a two-layer neural network with non-linear activation. Additionally, non-linear activation also has the function of transforming the output of the final layer to a well-suited format for the specific target task, for example, based on whether the task is binary or multiclass classification.

Common activation functions include the sigmoid function, $\sigma(x) = \frac{1}{1+e^{-x}}$, which outputs values between 0 and 1 and can be used to represent probabilities for binary classification. Softmax, $\sigma(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$, produces a probability distribution for a set of classes, making it suitable for multiclass classification problems. Rectified linear unit (ReLU) is another straightforward yet widely used activation function and is defined as $\sigma(x) = \max(0, x)$, which outputs 0 for negative inputs and acts as a linear function otherwise.

Evaluation

A fundamental tool in the evaluation of classification tasks is the confusion matrix, which categorizes predictions into four types in the case of binary classification: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). These are displayed in a 2×2 table where each axis, with two elements, represents the predicted and actual labels respectively. This format can easily be generalized to multiclass classification.

Using the confusion matrix, the simplest metric to evaluate the model's predictions is **accuracy**, which measures the proportion of correct prediction out of all predictions:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

The accuracy is easy to calculate and straightforward to understand. However, it may not be a reliable metric in the case of imbalanced data, where a model can obtain high accuracy by always predicting the majority class. In such cases, three other metrics, **precision**, **recall**, and $\boldsymbol{F_1}$ **score**, are often more reliable metrics for evaluating a model's performance.

For a given class label, precision measures the proportion of correctly predicted instances for which the model assigns that class label, while recall measures the proportion of actual instances with the class label that the model correctly predicts:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Precision and recall prioritize different aspects, with precision focusing on the accuracy of positive predictions, and recall emphasizing the coverage. A metric that balances precision and recall is the F_1 score, which is the harmonic mean of the two and is calculated as follows:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

A more generalized form of this, the $\boldsymbol{F_\beta}$ **score**, introduces the parameter β to adjust the weighting of precision and recall depending on the task priorities. A β value less than 1 places more weight on precision, while a value greater than 1 prioritizes recall:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}}$$

Another metric, which is common in multiclass classification, is **top-k accuracy**, where a prediction is considered correct if the true label is among the top k predicted labels. This is particularly useful for tasks with a large label set and multiple classes are acceptable, for example, in cases where differences between labels are subtle.

2.1.3 Tokenization

Tokenization is a fundamental process in NLP and is usually the first step in preparing text data for machine learning tasks. It involves breaking down text into smaller units, or tokens, to facilitate linguistic understanding and model training. While the process may seem straightforward, various challenges can sometimes arise due to the linguistic complexities inherent in different languages. Tokenization methods can generally be categorized into three groups based on their granularity: word-level, subword-level, and character/byte-level tokenization. Sentence segmentation, which splits text into individual sentences, is sometimes regarded as another form of tokenization. However, we do not discuss it in detail as segmenting text at the sentence level, while beneficial for certain applications (Reimers and Gurevych, 2019; Liu et al., 2021), is generally insufficient for many NLP models and tasks, which often require finer granularity such as words or subwords.

Word-level tokenization

The simplest approach to word-level tokenization is splitting text on white spaces, which include characters like spaces and newlines (`\n`). This form of white space tokenization is easy to implement and intuitive to understand. Furthermore, splitting text on white spaces is highly efficient (it requires little computation) and interpretable, especially in combination with static word type embeddings (discussed further in Section 2.2). However, because white space tokenization depends on the presence of explicit white spaces in text, it only works well on languages with clear word boundaries, such as most alphabet-based languages. Even in these languages, word-level tokenization has some limitations, particularly in its inability to recognize related word variants, such as *go*, *going*, and *went*. To address this, lemmatization and stemming tools are often employed (Bird et al., 2009). Lemmatization removes inflections from words and reduces them to their base, or dictionary forms using linguistic rules, while stemming simplifies words by stripping affixes from them without regard to grammatical correctness. Another major challenge with word-level tokenization is the out-of-vocabulary (OOV) problem, where less frequent words not present in the often fixed vocabularies of NLP models are unrecognized. A common workaround to this is replacing unrecognized tokens with an *unknown word token*, or *UNK*. This, however, results in information loss as the meaning of the unrecognized tokens is not retained, motivating better tokenization methods that can represent words even when they are not recognized by the tokenizer.

Subword-level tokenization

One effective approach to addressing the out-of-vocabulary (OOV) problem is to preserve frequently occurring words as whole units while splitting less frequent words into smaller subunits, or subwords. This allows the semantic representations of unrecognized words to be effectively approximated by aggregating the meanings of their constituent subwords. While the derived meanings may not always be precise, they are generally more informative than a generic *UNK* token. Popular subword tokenization techniques include *WordPiece* (Schuster and Nakajima, 2012) and the closely related *Byte Pair Encoding (BPE)* (Sennrich et al., 2016). These methods construct a subword vocabulary by initially splitting text data into individual characters and iteratively merging the most frequent pairs of characters into larger units until a predefined vocabulary size is reached. *SentencePiece* (Kudo and Richardson, 2018) further eliminates the reliance on word boundaries and the need for pre-tokenized inputs. Instead, it processes raw texts directly, making it particularly useful for languages written without explicit word boundaries, such as Chinese and Japanese. SentencePiece serves as a tokenizer as well as a language-agnostic detokenizer, which reconstructs text from subwords, increasing its utility. Overall, subword tokenization approaches effectively address some of the limitations of word-level tokenization and are highly effective in reducing vocabulary

size while enhancing the representation capability of the model.

Character/byte-level tokenization

While significantly more robust than word-level tokenization, subword tokenization remains vulnerable to orthographic variations, such as spelling variants and typos, which can result in inconsistent tokenization outputs. Work by Lazaridou et al. (2021) further highlights that subword vocabularies can vary substantially depending on the temporal stamp and content of the training data. To address these limitations, an alternative tokenization approach that operates at a more fine-grained level has been proposed to break down text into individual characters or bytes (Clark et al., 2022; Tay et al., 2022; Xue et al., 2022). Because these methods represent input text sequences of Unicode characters or bytes, which are processed directly by the model, they are considered tokenization-free models. Tokenization-free models offer several advantages, including being language-agnostic and more robust to OOV words, as they do not rely on a fixed vocabulary. However, because character and byte sequences are often considerably longer compared to sequences of words or subwords, training such models is typically associated with increased computational cost.

2.2 Word Representations

Section 2.1.3 discusses various methods for tokenizing text into smaller units, or tokens, to facilitate the processing and understanding of textual input by the model. Once tokenized, these units must be converted into representations that encode essential information about them. How the tokens are represented, therefore, has a direct impact on their interpretability and the model's ability to extract meaningful semantic information from them.

2.2.1 Distributed Word Representations

Early approaches often represent words using symbolic representations, such as one-hot encoding and Bag of Words (BoW). While these methods are simple and interpretable, they suffer from significant limitations, including sparsity and high dimensionality, which prevent scalability to large vocabularies and the ability to model meaningful semantic relationships between words. Although BoW is able to infer rudimentary relationships using word co-occurrence patterns, it is not possible to represent deep semantic and contextual connections of words.

A new paradigm for learning word representations is inspired by the distributional hypothesis (Harris, 1954), which states that words occurring in similar contexts tend to have similar meanings. This principle can be applied to the learning of word representations by

optimizing tasks that leverage the semantic similarities of words sharing similar contexts. Bengio et al. (2003) are the first to propose the learning of dense, distributed word representations, also known as word embeddings, by using neural networks to predict the next word in a sequence. Collobert and Weston (2008) further highlight the potential of general purpose word embeddings pre-trained using a language modeling objective and semi-supervised learning. The effectiveness of pre-trained word embeddings has subsequently been demonstrated by works such as Turian et al. (2010) and Socher et al. (2013) on a variety of NLP tasks.

Mikolov et al. (2013a) popularized *Word2Vec*, a toolkit for training word embeddings that gained widespread adoption for its simplicity to implement and fast training speed. Word2Vec operates in two modes: *Skip-gram* and *Continuous Bag-of-Words (CBOW)*. Both modes learn word contexts using a sliding window, with Skip-gram predicting context words from a target word, while CBOW predicts the target word from context words. Word2Vec uses one-hot encoded inputs and a single hidden layer to predict the probabilities of context words across a vocabulary using softmax, deriving word embeddings from the hidden layer's parameters after training. Calculating probabilities with standard softmax, however, is computationally expensive for large vocabularies. To address this, the authors propose two optimizations: *hierarchical softmax*, which uses a Huffman tree to encode tokens by frequency and reduces the computation to logarithmic complexity; and *negative sampling* (Mikolov et al., 2013c), which updates probabilities for a small number of sampled negative words instead of the entire vocabulary, thus improving efficiency.

A limitation of Word2Vec is that it updates word embeddings based only on local co-occurrences within a limited context window, ignoring global co-occurrence patterns. While this enables Word2Vec to model linear analogical relationships, as famously demonstrated by the $king - man + woman = queen$ example, it lacks a global perspective. To address this shortcoming, *GloVe* (Global Vectors) (Pennington et al., 2014) is introduced to combine local context information with global co-occurrence data by constructing a word-to-word co-occurrence matrix. This matrix would be prohibitively large for large vocabularies and is factorized into smaller matrices using methods such as latent semantic analysis (LSA) (Deerwester et al., 1990). This results in a compact matrix of size $|V| \times D$ that represents the word embeddings, where $|V|$ is the vocabulary size and $|D|$ is the embedding dimension. GloVe embeddings retain the ability of Word2Vec to model linear relationships while incorporating global co-occurrence patterns, which often results in better performance and faster training compared to Word2Vec under similar conditions.

A further innovative embedding learning method, *fastText* embeddings (Bojanowski et al., 2017), enhances the robustness of word embeddings against noise such as misspellings by incorporating subword information through character n-grams. This approach enables the model to partially reconstruct the meaning even when the input

word is unknown or deviates from the standard form due to noise like typos or spelling variations by representing words by the sum of their subword embeddings. This allows fastText embeddings to capture both semantic and morphological information of words and makes them particularly effective in handling cases involving rare words and linguistic variations.

2.2.2 Multilingual Word Embeddings

Bilingual word embeddings (BWEs), or more generally, multilingual word embeddings (MWEs), extend the concept of distributed word representations from one language to two or more languages, aiming to represent semantically similar words across languages with similar representations in a shared vector space. The training process for MWEs largely mirrors that of training monolingual embeddings and can be classified into three main approaches: projection-based, pseudo-parallel corpora-based, and joint methods (Ruder et al., 2019).

Projection-based methods

Projection-based methods are among the simplest approaches for learning MWEs and do not require any parallel data. These methods involve training separate monolingual embeddings on unlabeled corpora of the source and target languages independently, then aligning one of the vector spaces to the other by minimizing the distances between vectors of semantically similar words in both languages. This can be achieved, for example, by training a translation matrix (Mikolov et al., 2013b). Other approaches in this category include Lazaridou et al. (2015) and Vulić and Korhonen (2016).

Pseudo-parallel corpora-based methods

Pseudo-parallel corpora-based methods rely on the construction of synthetic pseudo-parallel data to assist the disjoint training of monolingual embeddings. For instance, Vulić and Moens (2015) leverage shuffled document-level aligned data on the same topics in two languages and apply the skip-gram model to the shuffled data. By shuffling document-level parallel data, similar words in both languages are exposed to comparable contexts, thereby encouraging similar semantic representations. Similar to this method, Gouws and Søgaard (2015) propose a more flexible approach by replacing words with any counterpart belonging to the same *equivalence classes* and not limited to translations of the word. For example, a noun in the source language can be replaced by any noun in the target language (based on the equivalence class defined by part-of-speech (POS) categories). This flexibility allows embeddings to capture both cross-lingual semantic and task-specific knowledge.

Joint methods

Joint methods for learning MWEs typically require some parallel data and have a joint objective that simultaneously minimizes losses in both languages. For instance, Hermann and Blunsom (2014) minimize the distances between sentence encodings of parallel sentences using bitext data. Klementiev et al. (2012) and Gouws et al. (2015) jointly train embeddings for source and target languages utilizing signals from word- and sentence-aligned parallel data. Further studies, such as Duong et al. (2017) and Chen and Cardie (2018), extend their approaches beyond previous training methods, which mainly focus on pairs of two languages, and show that MWEs benefit from joint training using multiple languages simultaneously. These methods highlight the potential of combined information from multiple languages in a shared multilingual representation space. Moreover, approaches like Eder et al. (2021) and Woller et al. (2021) explore low-resource setups where monolingual data is limited for the target language. These methods train embeddings for low-resource languages by leveraging resources available for resource-rich counterparts and demonstrate the effectiveness with case studies on Hiligaynon and Occitan, among others. Such works show that MWEs can be effectively learned even with limited target language data.

2.2.3 Contextualized Word Embeddings

One shortcoming of traditional embedding methods, such as Word2Vec and GloVe, is that they assign a single static vector to each word, without considering the specific contexts in which the word appears. This poses limitations which are especially problematic for polysemous words, whose meanings vary depending on the context. Contextualized word embeddings are introduced as a type of dynamic word embeddings to make word representations sensitive to the surrounding contexts. Peters et al. (2018) introduce *ELMo* (Embeddings from Language Models), which capture dynamic word representations using a bidirectional LSTM (BiLSTM)-based language model. ELMo consists of a forward language model, which predicts the next word in a sequence, and a backward language model, which predicts the previous word given future context. Both language models share the same token representations. This architecture allows ELMo to encode contextual information from both directions in its model parameters. The final representations are computed as a task-specific combination of the model's layers, which can be used as input features in the target task model.

A multitude of subsequent approaches for learning contextualized word representations build on language modeling objectives leveraging the Transformer architecture and self-attention mechanism (Vaswani et al., 2017) (discussed further in Section 2.3). For example, unlike traditional autoregressive language models, which predict the next word in a sequence and thus capture unidirectional context, BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) models bidirectional word contexts

using *masked language modeling*, which effectively learns representations leveraging context from tokens on both sides. XLNet (Yang et al., 2019), another Transformer-based model, implements a *permutation language modeling* objective and allows the learning of bidirectional context information from shuffled token sequences using an autoregressive language model.

Advancements in Transformer-based language models, such as BERT and XLNet, represent a significant milestone not only in the development of more contextualized word representations but also in the ability of language models to achieve a deeper understanding of the language. Further innovations, driven by the new pre-training paradigm, have enabled state-of-the-art performance across a wide range of NLP tasks.

2.3 Pre-trained Language Models

In recent years, pre-trained language models (PLMs) have emerged as a pivotal development in representation learning and natural language understanding (NLU). Before their adoption, NLP methodologies typically relied on task-specific models, which either learned word embeddings jointly during training or used pre-trained word embeddings. These embeddings were treated merely as input features to facilitate the learning of other task-specific parameters. However, such training approaches required re-training models from scratch for each new task, resulting in inefficiencies, limited adaptability, and a waste of resources.

The introduction of PLMs, built on the Transformer architecture and its underlying self-attention mechanism (Vaswani et al., 2017), represents a new paradigm. These models are pre-trained on large-scale unlabeled text corpora to learn generic language representations that capture rich linguistic knowledge, such as syntax and semantics. This unsupervised pre-training process also allows the models to be efficiently adapted, or fine-tuned, to perform various downstream tasks. Once pre-trained, PLMs typically require much less task-specific data to perform well and have demonstrated superior performance across a wide range of NLP tasks compared to models trained on task-specific data from scratch (Howard and Ruder, 2018; Radford et al., 2018; Devlin et al., 2019).

To understand how PLMs operate, it is essential to examine the two foundational innovations that form the backbone of these models: the self-attention mechanism and the Transformer architecture, which will be discussed in the following.

2.3.1 Attention Mechanisms

Attention

Attention mechanisms were first introduced to address the limitations of encoder-decoder models for neural machine translation, specifically the decoder's reliance on a fixed-length vector representation of the input sequence. These models typically use recurrent networks, such as LSTMs, to encode entire input sequences into a single vector, derived from the encoder's final hidden state. However, the fixed-length vector often poses an information bottleneck and struggles to retain sufficient contextual information, especially for long or complex sequences.

Attention addresses this by allowing the decoder to selectively focus on all encoder hidden states, rather than relying only on the final hidden state. This is achieved by assigning importance scores to each encoder hidden state at every decoding step, which allows the decoder to extract relevant information and better capture distant dependencies. Prominent variants of attention in encoder-decoder architectures include *additive attention* (Bahdanau et al., 2015) and *multiplicative attention* (Luong et al., 2015).

Additive attention, also called *Bahdanau attention*, computes alignment scores between the decoder's previous hidden state and each encoder hidden state. These scores are normalized using softmax to produce the weights of each encoder hidden state, whereby the most relevant encoder states are emphasized. The weighted sum of encoder states is then combined with the current decoder hidden state to produce the output at each decoding step. Formally, the alignment score between the decoder hidden state at step t, h_t and the encoder hidden state at step s, $\bar{h}_s$, is given by:

$$\text{score}(h_t, \bar{h}_s) = v_a^\top \tanh(U_a h_{t-1} + W_a \bar{h}_s)$$

where v_a, U_a, and W_a are trainable weight matrices.

Multiplicative attention, or *Luong attention*, on the other hand, simplifies the calculation of alignment scores by directly computing the dot product between encoder and decoder hidden states, which is more efficient than the additive method. The alignment score in Luong attention is defined as:

$$\text{score}(h_t, \bar{h}_s) = h_t^\top W_a \bar{h}_s$$

Both Bahdanau and Luong attention mechanisms are applied to decoder hidden states to attend to relevant encoder states. In contrast, self-attention operates solely within a single sequence, capturing contextual dependencies across words in the same sequence.

Self-attention

As the name suggests, *self-attention* is a specialized attention mechanism applied within a single sequence, allowing the model to capture relationships between different positions

in the sequence. Cheng et al. (2016) apply an intra-attention mechanism on an LSTM network to enhance the reading comprehension of input sequences. This application is conceptually similar to self-attention. Subsequent studies have further shown the efficacy of intra-attention in improving language understanding (Parikh et al., 2016; Paulus et al., 2018).

Building upon these ideas, Vaswani et al. (2017) lay the groundwork for future development of Transformer-based models by introducing *scaled dot product attention*, which functions as follows: for each input embedding, queries, keys, and values in the form of vectors are created from Q, K, and V, which are trainable weight matrices. Attention scores are computed by taking the dot product of the query and key vectors, scaling the results by $\sqrt{d_k}$, where d_k is the dimensionality of the key vectors, and applying softmax to produce the weights, which are finally multiplied with the value vectors. The process can be formulated as a matrix operation and efficiently applied to all tokens in the sequence in parallel:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Further extending on this mechanism, the authors propose the use of multiple *attention heads*, which they term *multi-head attention*. Each head i performs the attention function in parallel, with its own query, key, and value matrices as follows: $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$, where $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ are again trainable parameter matrices. The outputs from all heads are concatenated and projected to produce the final values:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

Multi-head attention allows the model to attend to information at different positions in different representation subspaces, enabling the model to focus on different aspects of information.

2.3.2 Transformer Architecture

The Transformer model, introduced by Vaswani et al. (2017), adopts an encoder-decoder architecture in which recurrent layers, such as RNNs, are completely replaced with self-attention mechanisms. This enables the Transformer to process input tokens in parallel, overcoming the constraints of sequential processing in previous recurrent architectures. The architecture comprises two components: the encoder, which transforms the input sequence into a contextualized vector representation, and the decoder, which generates an output sequence token by token. A detailed schema of the Transformer architecture is shown in Figure 2.1.

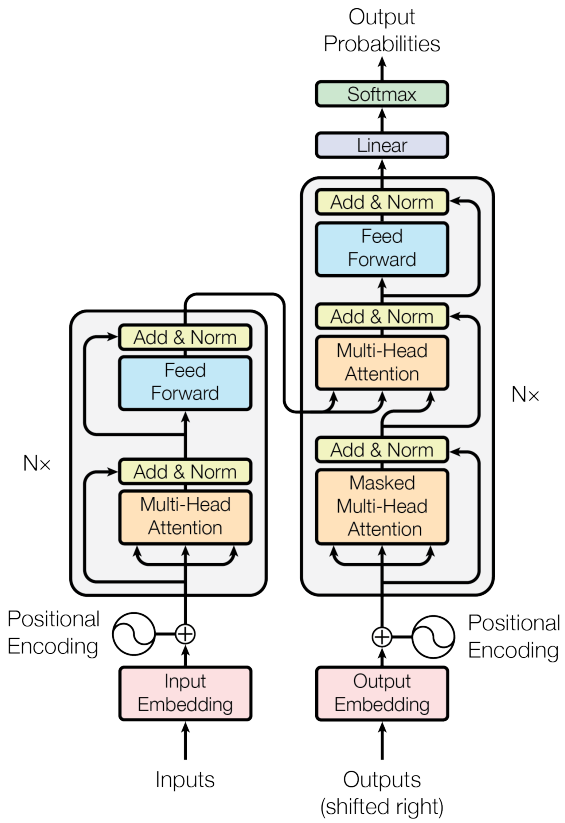


Figure 2.1: An illustration of the Transformer architecture from Vaswani et al. (2017). The architecture consists of N encoder blocks, followed by the same number of decoder blocks.

Encoder

The Transformer encoder consists of six stacked encoder blocks, each with an identical architecture but without sharing weights. Each encoder block comprises a multi-head attention layer and a position-wise feedforward layer, with a residual connection (He et al., 2016) around both layers, followed by layer normalization (Ba et al., 2016). The input embeddings are fed into the first encoder block, and the output of each block is passed to the next. The final encoder block produces outputs in the form of key and value vectors (Section 2.3.1), which are then processed by the decoders.

Decoder

The decoder part of the Transformer model also consists of six stacked blocks, similar to the encoder, and attends to the output of the encoder as well as the previous decoder output. An additional masked multi-head attention layer is inserted into decoder blocks, ensuring that the decoder can only attend to previous positions to prevent information

leakage from future positions during decoding.

Positional encoding

Unlike recurrent architectures, the Transformer processes all input tokens in parallel, making it inherently agnostic to the positional information of each token. However, natural language relies on sequential structures, and the absence of positional information would impair the Transformer's ability to model the syntax and semantics of languages. To address this, the Transformer introduces positional encodings, which are vectors of the same dimensionality as the input embeddings and can be added to the embeddings to incorporate positional information. These encodings are generated using a sinusoidal function defined as follows for even and odd positions:

$$\mathrm{PE}_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

$$\mathrm{PE}_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

where pos refers to the token's position, d_{model} is the dimensionality of embeddings and positional encodings, and i is the specific dimension in the encoding. This ensures each input position has a distinct encoding and that relative distances between positions are preserved. While positional encodings can also be learned during training instead of being generated using an encoding function (Gehring et al., 2017), Vaswani et al. (2017) demonstrate that sinusoidal encodings perform comparably to learned encodings while offering more simplicity.

2.3.3 Early PLMs

Greatly facilitated by the Transformer architecture, pre-trained language models (PLMs) mark a milestone in NLP through their pre-train-fine-tune paradigm. Under this paradigm, language models are first pre-trained on massive raw text corpora using language modeling objectives, enabling them to produce contextualized representations and encode generic linguistic knowledge in their parameters. The pre-trained models can subsequently be fine-tuned and excel in various downstream tasks. Based on their architecture, PLMs can be broadly categorized into three groups: encoder-only, decoder-only, and encoder-decoder models.

Encoder-only models

Encoder-only models leverage the encoder component of the Transformer architecture and focus on understanding and creating a representation for the input text. Many encoder-only models use an autoencoding objective, which aims to reconstruct the

original text from a corrupted input sequence. One representative training objective is *masked language modeling (MLM)*. In MLM, a portion of the tokens are replaced with either a special `[MASK]` token or random alternatives, and the model's objective is to maximize the probability $p(x_i|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n, \theta)$ for each masked position i in a sequence x. To reconstruct the original tokens, the model leverages context from both directions. Autoencoding models are thus often bidirectional. *ELMo*, mentioned in Section 2.2.3, also uses bidirectional context. However, it does so shallowly by concatenating outputs from two independent unidirectional models. Prominent examples of encoder-only models include *BERT* and its derivatives.

BERT (Devlin et al., 2019) introduces two pre-training objectives: MLM and *next sentence prediction (NSP)*. In MLM, 15% of tokens are randomly masked, with 80% of them replaced by the `[MASK]` token, 10% with random tokens, and 10% left unchanged. This scheme reduces reliance on `[MASK]` and mitigates discrepancies between pre-training and fine-tuning data. NSP is motivated by NLU tasks requiring the understanding of sentence relationships and classifies whether sentence pairs are consecutive or not.

Subsequent models have been proposed building on BERT's architecture with refined pre-training processes. *RoBERTa* (Liu et al., 2019) puts forward the argument that BERT has been undertrained and optimizes it by removing the NSP objective, expanding its pre-training data, increasing sequence length and batch size, and adopting a dynamic masking scheme. These modifications yield state-of-the-art results on multiple benchmarks.

ALBERT (Lan et al., 2020) focuses on reducing model parameters using two techniques: decoupling embedding size from the hidden layer size, which allows increasing the hidden size without expanding the embedding parameters, and cross-layer parameter sharing, which prevents the parameters from growing with an increasing model depth. This results in a model with considerably fewer parameters but better performance compared to BERT.

Autoencoding models, while powerful, suffer from a pre-train-fine-tune discrepancy caused by the introduction of `[MASK]` tokens, which are not present during fine-tuning. In addition, autoencoding models make the assumption that masked positions can be reconstructed independently, which is not always valid in practice. To address this, *XLNet* (Yang et al., 2019) introduces a *permutation language modeling* objective, which uses different shuffled token orders to train the model to predict the next token in an autoregressive manner. This effectively enables XLNet to capture bidirectional context while at the same time eliminating the pre-train-fine-tune discrepancy.

Decoder-only models

Decoder-only models, contrary to encoder-only models, utilize only the decoder component of the Transformer architecture. These models typically employ an autoregressive language modeling objective, which predicts the next token in a sequence given its preceding context. Specifically, for a sequence $x = (x_1, \ldots, x_n)$, the autoregressive

objective models the probability of the sequence as $p(x) = \prod_{i=1}^{n} p(x_i|x_{<i}, \theta)$, where θ represents the model parameters. As can be seen from their objective, decoder-only models often operate unidirectionally and are well-suited for text generation tasks.

Typical of decoder-only models are models in the *GPT (Generative Pre-trained Transformer)* series. The first GPT model (Radford et al., 2018) introduces generative pre-training on unlabeled text from the BookCorpus dataset (Zhu et al., 2015), followed by supervised fine-tuning on various downstream tasks such as natural language inference (NLI) and question answering (QA). GPT-2 (Radford et al., 2019) extends this approach by scaling up both the model size and pre-training data, using a newly created WebText dataset, which is curated from web pages with low-quality content filtered. GPT-2 demonstrates exceptional zero-shot capabilities on the tested language modeling datasets, even without supervised fine-tuning.

Encoder-decoder models

Unlike encoder-only or decoder-only models, encoder-decoder models leverage the full Transformer architecture, combining a bidirectional encoder to create a contextualized representation of the input and an autoregressive decoder to generate an output sequence. This architecture makes encoder-decoder models particularly suitable for sequence-to-sequence (seq2seq) tasks, such as machine translation and text summarization. Encoder-decoder models are commonly pre-trained using a denoising objective, where a corrupted input sequence is reconstructed to the original sequence. Formally, the denoising objective aims to maximize $p(y_1, \ldots, y_n|x_1, \ldots, x_m, \theta)$, where x is the corrupted input, y is the original sequence, and θ denotes the model parameters. Prominent encoder-decoder models include *BART*, *T5*, and *Flan-T5*.

BART (Lewis et al., 2020a) applies various sequence corruption strategies, including token masking (similar to MLM), token deletion (random token removal), and text infilling (replacing spans of text with a single `[MASK]`). The diversity of its denoising pre-training improves BART's ability to generalize over a variety of seq2seq tasks. T5 (Text-to-Text Transfer Transformer) (Raffel et al., 2020) is pre-trained on the Colossal Clean Crawled Corpus (C4) and formats both inputs and outputs as sequences of text. To specify the task to be performed, a task-specific prefix is appended to the input, for example, "translate English to German: `[input text]`". Flan-T5 (Chung et al., 2024) is an instruction-tuned (Wei et al., 2022a) version of T5, which enhances the model's performance by fine-tuning on a wide range of tasks expressed as natural language instructions. This significantly enhances the zero-shot and few-shot results, which are comparable with much larger models.

Although encoder-decoder models are primarily designed for seq2seq tasks, they can also be adapted to solve traditionally discriminative tasks like classification or linear regression, as long as the target task is reformulated as a seq2seq problem and the training data contains the desired target labels in text format. One of the strengths of

encoder-decoder models lies in their consistent training objectives during pre-training and fine-tuning. Additionally, the architecture has shown effectiveness for multitask training and transfer learning (Raffel et al., 2020; Chung et al., 2024).

2.3.4 Multilingual PLMs

Following the success of early PLMs, particularly BERT, in achieving state-of-the-art performance across a wide range of NLP tasks, numerous BERT-based models in non-English languages have been developed (de Vries et al., 2019; Le et al., 2020; Cañete et al., 2023). However, these models are still language-specific, focusing on the primary language of their pre-training data, and are typically pre-trained for high-resource languages only. In contrast, multilingual PLMs aim to support a variety of languages at the same time, including those with limited pre-training resources. These models are designed to handle tasks across different languages using a unified model architecture. Through parameter sharing, multilingual PLMs facilitate knowledge transfer from high-resource to low-resource languages, enhancing their cross-lingual capabilities.

Multilingual BERT (mBERT) (Devlin et al., 2019) follows the same pre-training strategy as BERT but with an extended subword vocabulary shared by all languages and is trained using Wikipedia data from 104 languages, selected based on their data size. To reduce the impact of data imbalance across languages, exponential smoothing is applied to undersample high-resource language data and oversample data in underrepresented languages.

XLM (cross-lingual language model) (Conneau and Lample, 2019) is introduced following three training objectives: causal language modeling (CLM), masked language modeling (MLM), and translation language modeling (TLM). CLM and MLM are objectives commonly used in autoregressive and autoencoding language models. TLM is a multilingual extension of MLM, leveraging parallel sentence pairs in two languages, which are concatenated. By masking and reconstructing tokens in both sentences, TLM enables the model to learn both language-specific knowledge and cross-lingual alignment. This is shown to be especially effective for tasks like unsupervised machine translation.

XLM-R (XLM-RoBERTa) (Conneau et al., 2020) is inspired by RoBERTa and argues that both mBERT and XLM are undertrained due to the limitation of Wikipedia data's ability to scale, particularly for low-resource languages. Instead, XLM-R uses filtered CommonCrawl data (Wenzek et al., 2020), which significantly increases data availability, especially for low-resource languages. This large-scale pre-training allows XLM-R to achieve superior performance over mBERT and XLM on multiple cross-lingual benchmarks.

Glot500 (Imani et al., 2023) scales up the language coverage from typically around 100 to over 500 languages. It uses the XLM-R model as its backbone with an expanded vocabulary and is pre-trained on Glot500-c, a curated corpus encompassing 511 languages. Glot500 demonstrates significant performance improvements across both *head*

languages (languages already in the original XLM-R model) and *tail languages* (the remaining languages previously unsupported by mPLMs).

SERENGETI (Adebara et al., 2023) focuses on addressing the underrepresentation of African languages in existing mPLMs. While current mPLMs cover only about 31 out of 2000 African languages, SERENGETI extends this coverage to 517 African languages and language varieties. It is pre-trained on curated data for these African languages, as well as some of the world's most widely spoken languages. As a result, SERENGETI outperforms other mPLMs, including some with a specific focus on African languages (Ogueji et al., 2021; Alabi et al., 2022), on various benchmarks.

2.3.5 Large Language Models

The transition from early PLMs (Section 2.3.3) and multilingual PLMs (Section 2.3.4) introduced so far - which are typically limited to parameter sizes of around one billion or less and reliant on task-specific fine-tuning for optimal performance - to a new line of large language models (LLMs) represents yet another important paradigm shift beyond the pre-train-and-fine-tune era. LLMs exhibit remarkable universal understanding of language, offering impressive zero-shot and few-shot learning capabilities without the need for task-specific fine-tuning.

One demonstration of this shift is the ability to perform in-context learning, as shown by the GPT-3 model (Brown et al., 2020). GPT-3 showcases impressive performance across tasks such as factual question answering, text summarization, and translation when provided, or prompted, with a few examples illustrating the task. This approach requires no parameter updates and facilitates generalization to unseen tasks. The effectiveness of in-context learning can be attributed to the model's profound knowledge acquired during large-scale pre-training on extensive text data. Subsequent advances in prompting techniques, such as chain-of-thought reasoning (Wei et al., 2022c), have further enhanced the reasoning and problem-solving capabilities of these models.

Instruction tuning has been introduced as another important innovation to fine-tune LLMs to follow natural language instructions, with the motivation to improve generalization to unseen tasks and reduce the sensitivity of LLMs' performance to prompt engineering (Wei et al., 2022a; Wang et al., 2022). It also has the goal of aligning model behavior more closely with human preferences (Ouyang et al., 2022), often incorporating reinforcement learning from human feedback (RLHF) (Christiano et al., 2017). RLHF uses human feedback to train a reward model, which is subsequently used to guide the model optimization process using the proximal policy optimization (PPO) algorithm (Schulman et al., 2017) and enables the generation of outputs that maximize the alignment.

A remarkable feature of LLMs as they scale is the *emergent capabilities*, or skills that are absent in smaller models but arise without being explicitly elicited in larger ones as a result of scaled training (Brown et al., 2020; Wei et al., 2022b). Such capa-

bilities include advanced multi-step arithmetic reasoning, which greatly aids the model in solving complex problems such as mathematical tasks, and improved instruction following capabilities, which helps the model to generalize to new tasks by understanding instructions.

In the following, popular LLMs and LLM families, such as the GPT and Llama series, are introduced.

GPTs

The GPT series has undergone a significant evolution process in both scale and capability. GPT-2 (Radford et al., 2019), first introduced in Section 2.3.3, already demonstrates notable generative capabilities as an autoregressive language model. Unlike GPT-2, whose parameter sizes range from 117 million to 1.5 billion, GPT-3 (Brown et al., 2020) is expanded drastically in scale with 175 billion parameters, and is pre-trained on vast internet text corpora, including CommonCrawl and Wikipedia. This large-scale pre-training enables GPT-3 to exhibit remarkable in-context learning capabilities. A refined GPT-3.5 model serves as the foundation for *ChatGPT*, a conversational agent fine-tuned leveraging techniques such as RLHF, and has attracted immense attention both within and beyond the NLP community at the time of its release. The successor model, GPT-4 (OpenAI, 2023), is an advanced multimodal model capable of processing image inputs and outputting text outputs. It forms the backbone for the enhanced ChatGPT-4, offering improved reasoning and generalization capabilities. While OpenAI has not disclosed the exact configurations of GPT-4, such as its parameter size, it is still one of the most advanced and best-performing models today.

Llama

The Llama models[1] are a series of LLMs developed by Meta AI. The original LLaMA model (Touvron et al., 2023a) ranges from 7 to 65 billion parameters in size and is pre-trained on comparable internet data used for the GPT series. While much smaller in size, and thus less computationally demanding, LLaMA shows comparable performance with GPT-3 on certain benchmarks. Llama 2 (Touvron et al., 2023b) is introduced with architectural improvements upon LLaMA, including an optimized attention mechanism (Ainslie et al., 2023), which greatly enhances its efficiency and scalability. This version also introduces a larger model size of 70 billion parameters, enabling better generalization across tasks. The most recent models in the series, Llama 3 and 3.1 (Dubey et al., 2024), leverage improved data quality, a significant increase in pre-training data, and an extended context window. They also drastically scale up the model size to reach up to 405 billion parameters and provide multimodal support and improved abilities in areas such as multilinguality, coding, and reasoning. Furthermore, unlike many of its counterparts,

[1] `https://www.llama.com`

the Llama series is open-source and can be accessed by researchers under specified conditions, thereby encouraging further innovations based on these models and reducing the usage of computational resources.

Gemini

Gemini (Anil et al., 2023a) is a family of general-purpose models trained by Google DeepMind and is built based on its predecessor, PaLM 2 (Anil et al., 2023b), a capable multilingual model fine-tuned using RLHF. Gemini is pre-trained on diverse multilingual and multimodal data, such as webpages, books, code, and videos. This enables Gemini to process visual and audio inputs in addition to text. Similar to GPT-4, configurations of Gemini models, such as model parameters and the exact training process, have not been disclosed.

BLOOM

BLOOM (Scao et al., 2022) is a 176 billion-parameter open-source model pre-trained on the ROOTS corpus (Laurençon et al., 2022), a corpus derived from diverse HuggingFace datasets and comprising 46 natural languages and 13 programming languages. The model is developed with data quality and ethical considerations in mind, pointing out existing issues with simple heuristics-based data handling and filtering methods (Dodge et al., 2021; Johnson et al., 2022). The underlying ROOTS corpus has been created through a global, collaborative effort, prioritizing the agency of data holders and involving human moderation in pre-processing. BLOOM demonstrates competitive multilingual capabilities on a wide range of benchmarks while ensuring the ethical aspect of the training data.

Claude

Claude, another prominent model family developed by Anthropic, is pre-trained on a combination of public and proprietary data carefully curated and organized by Anthropic. The models leverage Constitutional AI (Bai et al., 2022) to uphold ethical considerations and ensure harmlessness by explicitly incorporating a list of rules and principles for enhanced alignment with values such as human rights. The latest models of the line, including the Claude 3 family (Anthropic, 2024a) and the most recent release, Claude 3.5 Sonnet (Anthropic, 2024b), have multimodal capabilities, allowing them to process both textual and visual inputs. Claude 3.5 outperforms its predecessors as well as competitive models such as GPT-4o and Llama 3 across multiple benchmarks, including tasks in reasoning, math, and coding.

Mistral

The Mistral 7B (Jiang et al., 2023) is a 7-billion-parameter model developed by Mistral AI. It employs techniques such as grouped-query attention (GQA) and sliding window attention (SWA) to achieve fast inference speed and memory efficiency. The instruction-tuned variant of Mistral 7B shows superior performance and a good balance between safety and utility when compared with models of comparable or greater sizes, such as Llama 2 (13B). Building upon this, Mixtral 8x7B (Jiang et al., 2024) is a sparse mixture of experts (SMoE) model based on the Mistral architecture. Mixtral dynamically selects two experts at each timestep, allowing each token to benefit from the model's full parameters while controlling latency and ensuring computational efficiency. Mixtral demonstrates enhanced multilingual support and comparable or superior performance to Llama 2 (70B) and GPT-3.5 on multiple benchmarks.

2.4 Multilinguality

2.4.1 Multilingual Evaluation

As mPLMs expand their language coverage (shown in Section 2.3.4), the ability to comprehensively evaluate them across a wide range of languages has become imperative. Multilingual benchmarks, such as XTREME (Hu et al., 2020) and its improved version, XTREME-R (Ruder et al., 2021), are designed to assess the ability of mPLMs across a diverse range of tasks, including classification (Conneau et al., 2018), sequence tagging (Nivre et al., 2018; Pan et al., 2017), question answering (Artetxe et al., 2020a; Lewis et al., 2020b), and sentence retrieval[2]. These benchmarks are, however, limited to 40-50 languages, which falls short of the language coverage of many contemporary mPLMs. Some of the datasets that form parts of these multilingual benchmarks, such as WikiANN (Pan et al., 2017) and Tatoeba, provide broader coverage and support 282 and around 400 languages respectively (with Tatoeba's language coverage continuously expanding). However, their scope is relatively narrow, focusing on POS tagging and sentence retrieval only.

Flores-101 (Goyal et al., 2022a) is a parallel benchmark dataset designed to assess machine translation systems on a diverse set of languages and topics. It comprises 3001 sentences extracted from Wikipedia on a variety of different subjects, which are then translated into 101 languages by professional translators. Flores-200 (Costa-jussà et al., 2022) builds on Flores-101 by expanding the language coverage, especially to low-resource languages. By adopting many-to-many translation, Flores-200 incorporates bitext with non-English source languages, thereby reducing English-centrism in its data

[2] https://tatoeba.org

and improving the translation quality where English may not be the optimal source language.

Sourcing its data from the Flores-200 dataset, Belebele (Bandarkar et al., 2024) provides 900 parallel multiple-choice reading comprehension questions across 112 languages. The questions are carefully curated, and the choices are developed for English and subsequently translated into other languages by human translators. Likewise, SIB200 (Adelani et al., 2024) introduces a topic classification dataset covering 205 languages and dialects, using data from Flores-200. The labels are created for English and extended to other languages leveraging the parallelism of Flores-200.

To further push the boundaries of multilingual evaluation, Ma et al. (2023) develop Taxi1500, a text classification dataset based on Bible data. By leveraging the parallel nature of Bible translations, Taxi1500 creates automatically projected labels for over 1500 languages, enabling the evaluation of massively multilingual PLMs, including Glot500. Taxi1500 will be discussed in detail in Chapter 3.

2.4.2 Cross-Lingual Transfer

Cross-lingual transfer learning is commonly deployed to address data scarcity in low-resource languages by leveraging data or models that are typically more readily available for high-resource languages to perform tasks in low-resource languages. One class of solutions employs translation-based approaches, where annotated training or test data in resource-rich languages are translated into low-resource ones, or vice versa (Mayhew et al., 2017; Fei et al., 2020; Unanue et al., 2023). Other techniques make use of multilingual word embeddings (Section 2.2.2) and multilingual PLMs (Section 2.3.4) to transfer representational spaces across languages (Gouws et al., 2015; Imani et al., 2023). These include alignment-based methods that enhance low-resource language embeddings through post-alignment with trained high-resource language embeddings, which are generally of better quality (Artetxe et al., 2017; Lample et al., 2018b). With the advent of Transformer-based models, more recent work on cross-lingual transfer learning has shifted toward adapting PLMs cross-lingually, leveraging their extensive parameter spaces and representational power to enable cross-lingual transfer (Artetxe et al., 2020a; Minixhofer et al., 2022; Pham et al., 2024).

Translation-based transfer learning

Translation-based transfer learning leverages lexica or machine translation (MT) systems to obtain low-resource language data from labeled high-resource language datasets, and has been widely adopted in early studies on cross-lingual transfer due to its simplicity and effectiveness. These approaches typically translate annotated source language data into target languages and project the original labels onto the translated datasets. For instance, Mayhew et al. (2017) employ dictionary-based “cheap” translation to create

training data for cross-lingual named entity recognition (NER). They then transfer labels from the original English data to target languages, achieving substantial improvements over state-of-the-art methods relying only on target language data. Similarly, Fei et al. (2020) adopt a translation-based strategy for semantic role labeling (SRL) and show that combining source and translated target language data achieves significant performance gains.

Translation-based methods can also work in reverse, as demonstrated by Unanue et al. (2023), who translate target language test data into English and demonstrate significantly improved performance on multiple multilingual text classification tasks over a competitive mPLM baseline. Furthermore, Etxaniz et al. (2024) introduce *self-translate*, which uses the few-shot translation capabilities of an mPLM to first translate non-English prompts into English before performing the specific tasks. This is shown to outperform direct inference in target languages and approaches the performance of MT-based methods, especially with larger model sizes.

Despite their simplicity, Ebrahimi and von der Wense (2024) and Zhou et al. (2024) highlight that MT systems regularly fail to produce accurate translations that capture necessary task-specific nuances of the data that are critical for optimal performance. Furthermore, Artetxe et al. (2023) explore both *translate-test* (translating target language test data into English) and *translate-train* (translating English training data into target languages), and reveal that the choice is largely task-dependent. While translate-train is more advantageous for shallow tasks, such as sentiment analysis, complex tasks that require reasoning, such as natural language inference (NLI), benefit more from a translate-test approach.

Cross-lingual adaptation of PLMs

Building on the effectiveness of translation-based transfer learning, Transformer-based PLMs and mPLMs have increasingly been used to complement these strategies. Their strong language understanding capabilities and rich multilingual representations, especially in mPLMs, enable effective generalization across languages. Despite the utility of mPLMs discussed in Section 2.3.4, questions have been raised about how to extend them to unseen languages, which remain a challenge to cross-lingual transfer. In addition, studies have highlighted limitations of mPLMs compared to their monolingual counterparts (Wu and Dredze, 2020; Rust et al., 2021), further motivating efforts to explore the abovementioned questions. Approaches to tackle the cross-lingual adaptation of PLMs can be broadly categorized into two directions: adapting PLMs or mPLMs to enhance the performance of a specific language, and extending existing PLMs or mPLMs to effectively support unseen languages.

Adapting PLMs to a specific language Several approaches leverage English PLMs and adapt them to a target language, focusing on the lexical level. Artetxe et al. (2020a)

challenge the assumption that both a shared vocabulary and joint pre-training are essential for cross-lingual transfer. They first pre-train an English LM, swap in target language embeddings, fine-tune it with English data, and subsequently achieve zero-shot transfer on the target language. Similarly, Tran (2020) initializes target language embeddings to align with the English vector space through mapping and fine-tunes them jointly with shared encoder layers, producing a bilingual model. de Vries and Nissim (2021) follow previous findings on high-density information in the lexical layers of PLMs (de Vries et al., 2020), and re-train target language embeddings while keeping the Transformer layers of a GPT-2 model frozen, which demonstrates effective transfer performance. Minixhofer et al. (2022) improve on this by initializing subword representations using aligned multilingual embeddings, which achieves consistent improvements. Kuratov and Arkhipov (2019) leverage an mPLM, mBERT, to enhance vocabulary initialization for a monolingual Russian LM. Their approach addresses the tokenization inefficiencies caused by multilingual tokenizers, specifically that of high subword fertility (Rust et al., 2021) - which refers to a high average number of subwords - and the associated computational inefficiency.

Extending mPLMs to unseen languages Approaches that extend mPLMs to new languages typically do so by modifying the vocabulary of an existing mPLM. Common methods include allocating new vocabulary entries (Wang et al., 2020b; Ebrahimi and Kann, 2021; Imani et al., 2023) or learning a joint vocabulary leveraging techniques such as BPE (Chronopoulou et al., 2020), followed by continued pre-training with target language data. Other approaches bypass the need for modifying the original mPLM's vocabulary, for example, by leveraging knowledge from related languages already in the mPLM (Muller et al., 2020). Soft-prompt tuning, proposed by Chen and Chen (2024), offers an efficient solution by introducing minimal additional parameters to achieve effective zero-shot transfer without adapting the vocabulary. Alabi et al. (2022) introduce multilingual adaptive fine-tuning (MAFT) to adapt mPLMs to 17 African languages. By removing vocabulary corresponding to non-African scripts, their method increases the specialization of mPLMs on these languages and demonstrates improved zero-shot transfer abilities. Pham et al. (2024) propose an initialization method for language-specific embeddings leveraging both lexical and semantic alignment from PLMs. Their method further determines the optimal vocabulary size for each target language, which significantly enhances the efficacy of cross-lingual transfer.

Adapters

Adapters (Rebuffi et al., 2017) have emerged as a modular and parameter-efficient alternative for transfer learning. Unlike traditional full-model fine-tuning, which requires updating all model parameters, adapter-based methods inject lightweight adapter modules between layers of a PLM. These modules are initialized as identity functions for stable

training and have significantly fewer parameters than the model itself. In a typical fine-tuning setup using adapters, only the adapter parameters are updated, whereas the rest of the model parameters remain frozen. This design effectively limits task-specific updates to the adapter parameters, thereby avoiding catastrophic forgetting typically associated with applications such as multitask learning (French, 1999).

The concept is first introduced in the form of residual adapters for computer vision by Rebuffi et al. (2017), which demonstrate their adaptability across diverse visual domains. Houlsby et al. (2019) extend this approach to NLP by fine-tuning an adapter-injected BERT model for diverse text classification tasks and achieve comparable performance to full-model fine-tuning by updating only a fraction of the parameters.

The application of adapters for cross-lingual transfer learning is motivated by the drawbacks of mPLMs, including a trade-off between language coverage and performance, and suboptimal results even for some high-resource languages (Eisenschlos et al., 2019; Conneau et al., 2020; Wu and Dredze, 2020). To address these challenges, Pfeiffer et al. (2020) introduce MAD-X, an adapter-based framework to facilitate cross-lingual transfer. MAD-X employs three types of adapters: language adapters for learning language-specific transformations, task adapters for encoding task-specific knowledge, and invertible adapters for facilitating embedding adaptation. The modular and model-agnostic design allows for flexible integration of MAD-X and demonstrates strong performance on both high-resource languages in mPLMs and low-resource languages unseen by mPLMs. Following the same idea, Parović et al. (2022) propose BAD-X, which learns bilingual adapter pairs instead of individual language adapters. This approach effectively captures the interplay between source and target languages, leading to strong zero-shot transfer performance between language pairs. Another work by Lee et al. (2022) introduces FAD-X, an adapter fusing method for composing task adapters for low-resource languages leveraging different pre-trained adapters for other languages.

While individually trained adapters are effective for single tasks or languages, they lack the ability to share useful information across tasks or languages. To address this, Pfeiffer et al. (2021) introduces AdapterFusion, a method that utilizes a fusion layer to adaptively combine information from multiple task adapters. This, however, focuses on task-specific capabilities and overlooks the interdependence between task and language abilities. To solve this limitation, AdaMergeX (Zhao et al., 2024) decouples task and language knowledge by splitting target task ability into task and language abilities and then adaptively merging task and language adapters. This not only enhances the cross-lingual transfer ability of AdaMergeX but also allows it to mimic the $king - man + woman = queen$ analogy of Word2Vec (Mikolov et al., 2013a).

2.5 Summary

In this chapter, we have provided a comprehensive overview of the essential technical background to this work. We began by introducing the foundational concepts of neural networks and machine learning, detailing the steps of their operation and evaluation. Following this, we explored the motivation and advancements in multilingual NLP, highlighting its significance in addressing linguistic diversity and identifying challenges faced within the domain. Building on this, we will delve into diverse practical topics in the remainder of this dissertation. The discussions in the following will be directed toward addressing the research questions outlined in Section 1.2.

Chapter 3

Scaling NLP Datasets to 1500 Languages

This chapter corresponds to the following work:

> Chunlan Ma, Ayyoob ImaniGooghari, **Haotian Ye**, Ehsan Asgari, Hinrich Schütze (2023). Taxi1500: A multilingual dataset for text classification in 1500 languages.

Declaration of Co-Authorship. Ayyoob ImaniGooghari conceived the idea of constructing a massively multilingual parallel dataset for text classification, which motivated this project. Chunlan Ma, Ayyoob ImaniGooghari, and I coordinated the data collection effort, employing external annotators, and collaborated extensively during the process of data validation and the development of the survey used by external data annotators. Ehsan Asgari contributed 1000Langs, a corpus of parallel Bible texts, and the accompanying data crawler, developed as part of his previous research. These resources were provided to be used for the creation of our dataset. Chunlan Ma conducted evaluations using the developed dataset and performed an analysis of the corpus statistics. I worked on the analysis of the experimental results as well as their presentation, and wrote the first draft which was originally submitted to ACL 2023, rejected, and later accepted at NAACL 2025. All authors except Ehsan Asgari reviewed the draft.

3.1 Introduction

Despite significant advancements in NLP, progress remains predominantly focused on widely spoken languages with higher resources, leaving a vast majority of the world's over 7000 languages underrepresented (Joshi et al., 2020). This underrepresentation highlights a pervasive global issue of language inequality, often reflected by the fact that minority and low-resource languages are systematically excluded from language technologies. Such exclusion contributes to virtual barriers such as the *digital language divide* (Young, 2015), often limiting access to information and tools for speakers of underrepresented languages and further exacerbating existing inequalities.

Recent advancements in mPLMs, such as BERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020), and Glot500 (Imani et al., 2023), as well as more recent multilingual LLMs like BLOOM (Scao et al., 2022) and Aya (Üstün et al., 2024), have demonstrated the potential to extend language support to a wide range of previously underrepresented languages. However, one challenge remains with the lack of comprehensive knowledge about low-resource languages, particularly typologically complex ones, which often contributes to the neglecting of these languages (Ponti et al., 2019). Moreover, most existing mPLMs fail to achieve a higher coverage of languages due to a lack of more linguistically diverse and representative evaluation datasets. Notably, models such as mBERT and XLM-R are evaluated on a much smaller subset of languages than those covered by their pre-training data, largely because existing benchmark datasets lack sufficient linguistic diversity.

To address these limitations, we propose a novel multilingual text classification dataset spanning 1504 languages. For its development, we leverage Bible translations and develop generalizable topics that widely apply to a large number of verses. The verse-level alignment of Bible translations facilitates the projection of annotations across languages. Annotations for English verses are obtained through crowdsourcing and subsequently projected onto parallel verses across other languages without compromising data quality. To ensure data quality, we implement quality control measures prior to and during the annotation process, and calculate inter-annotator agreement scores using Krippendorff's α.

In addition to providing an overview of our dataset, we present a comprehensive benchmark evaluating the multilingual performance of four mPLMs (mBERT, XLM-R Base, XLM-R Large, and Glot500) and six LLMs (Llama 2 7B (Touvron et al., 2023b), Mistral 7B (Jiang et al., 2023), and BLOOM in various sizes (560M, 1B, 3B, 7B) on our dataset. The mPLMs are evaluated on all 1504 languages, while the LLM evaluation is conducted on a subset of 64 languages. Our results highlight the superior multilingual capabilities of Glot500, which can be attributed to its inclusion of a broader range of languages during pre-training. Additionally, evaluations of LLMs reveal their competitive performance with fewer prompts using low-resource language data compared to traditional fine-tuning of mPLMs.

3.2 Related Works

To date, most datasets designed for multilingual evaluation cover no more than a few hundred languages (Artetxe et al., 2020a; Ruder et al., 2021; Goyal et al., 2022a; Adelani et al., 2024), falling significantly short of the total number of the world's languages. These datasets therefore represent only a limited part of the world's linguistic diversity. The absence of high-quality evaluation datasets poses a restriction on the development and evaluation of NLP tools, including language models, particularly for low-resource ones. A more thorough overview of some of these multilingual evaluation datasets and their limitations is provided in Section 2.4.1.

Parallel corpora play a crucial role in multilingual research, as they commonly serve as cross-lingual bridges that enable the understanding and processing of underrepresented languages by leveraging higher-resource ones through their alignment. Such corpora facilitate both the training and evaluation of models under cross-lingual and multilingual settings and enable NLP systems to generalize effectively to low-resource languages. In this study, we leverage translations of the Bible as the source of parallel data due to its coverage over a linguistically diverse expanse of languages. The Bible is also considered a valuable source of parallel data due to its inherently consistent verse structure and high-quality translations. Specifically, we employ two sources of Bible collections: the Parallel Bible Corpus (PBC) (Mayer and Cysouw, 2014), which contains Bible translations in 1304 languages, and the 1000Langs dataset[1], a collection of Bible translations compiled from multiple online sources. Together, these two resources amount to 1504 languages, which greatly exceeds the number of supported languages in most contemporary multilingual evaluation datasets. Our dataset, Taxi1500, built upon this joint collection of Bible translations, represents the most linguistically diverse dataset available for multilingual NLP.

3.3 Dataset

3.3.1 Sentiment Classification

While exploring possibilities to create a classification task based on Bible translations, we initially attempt to formulate verse classification as a sentiment classification task by categorizing Bible verses into three conventional polarity labels: *positive*, *neutral*, and *negative*. Because many low-resource languages only have translations of the New Testament, we base our dataset only on verses from the New Testament. Inspired by Dufter et al. (2018), we implement a similar task to classify English Bible verses in the PBC into the three polarity labels. This is carried out using a RoBERTa-based sentiment classification model (Hartmann et al., 2021), which has been fine-tuned on social media

[1] https://github.com/ehsanasgari/1000Langs

posts. We apply this model to the New World Translation (2013) and obtain 6233 verses as positive, 1441 as negative, and 23459 as neutral.

Given the predominance of neutral verses, we further conduct emotion classification on the verses classified as either positive or negative in the previous step, as we assume that these verses are more likely to exhibit distinguishable emotions. For this, we employ a DistilBERT model[2] fine-tuned with six emotion labels: *sadness*, *joy*, *love*, *anger*, *fear*, and *surprise*.

With this method, we are able to divide the positive and negative verses into the six emotion categories with the corresponding numbers of verses: *sadness* (1171), *joy* (1952), *love* (870), *anger* (4201), *fear* (457), and *surprise* (29). However, closer examination of the classified results demonstrates the impracticality of this method. As most Bible verses are inherently objective, forcing them into one of the emotion categories leads to a high number of misclassifications. In fact, because the majority of verses in the Bible do not convey a single, definitive sentiment or emotion, sentiment or emotion classification approaches become impractical. We consequently abandon these approaches and explore alternative classification topics that better align with the nature of the data.

3.3.2 Topic Design

Latent Dirichlet Allocation

The failure of sentiment and emotion classification in Section 3.3.1 illustrates that subjective classification tasks are not well-suited for Bible data. We thus shift to creating objective topics in an automatic manner using Latent Dirichlet Allocation (LDA) (Blei et al., 2003), which detects latent topics within Bible verses.

We first remove stop words using the NLTK package (Bird et al., 2009), and filter out words with very high frequencies, such as *God* and *Jehova*, as well as meaningless character combinations. The output of LDA consists of lists of tokens that together represent latent topics. However, the results generated by LDA on the Bible verses do not indicate meaningful or interpretable topics. The following examples illustrate some topics produced by the LDA model:

Topic 1 : $[house, people, one, may, david, sons, become, day, according, saying]$

Topic 2 : $[david, son, one, house, man, things, came, king, hand, land]$

Topic 3 : $[sons, israel, one, like, king, house, man, people, us, men]$

Topic 4 : $[land, one, let, people, men, us, went, took, go, brought]$

Topic 5 : $[one, israel, king, people, may, like, man, days, seven, Moses]$

[2] https://huggingface.co/bhadresh-savani/distilbert-base-uncased-emotion

These examples show significant overlaps among the tokens produced for different topics, which make it challenging to identify distinct themes among the topics. At the same time, the results created are hard to interpret and concretize into meaningful topics. These limitations likely result from the fact that LDA is optimized for processing longer documents, while Bible verses are typically limited to fewer than 50 tokens, which lack sufficient contextual information for LDA to extract meaningful latent topics. The LDA's modeling could furthermore be impacted by certain verses that do not focus on a single topic. This outcome demonstrates the challenges with automated topic identification and motivates an alternative approach for developing suitable topics for the Bible verses, which involves manual topic curation.

Manual topic engineering

Due to the inability of LDA to produce meaningful topics for Bible verses, we turn to manual engineering of relevant topics by directly working with the verses. The curation process undergoes seven refinement iterations in total, which are listed as different versions in Table 3.1. The final version contains six topics: *Recommendation*, *Faith*, *Description*, *Sin*, *Grace*, and *Violence*, whose definitions are shown in Table 3.2. For conciseness, we omit definitions for topics from earlier iterations. The detailed process of topic refinement is described below.

In the first iteration, nine topics have been chosen: *Rules*, *Phenomenon*, *Conflict*, *Relation*, *Place*, *Character*, *Reward*, *Punishment*, and *Command*. While collecting example verses for these topics, however, we recognize overlapping definitions among certain topics, as some verses could be categorized as multiple options. To reduce the overlap and develop more precise topics, we seek assistance from experts in theology, as well as searching for relevant topics from online resources.

Using various online resources, including preaching websites such as *ProPreacher*[3], we select relevant topics that are referred to as v2 in Table 3.1. These topics are curated by balancing two principles: selecting enough topics so as to make the benchmark challenging, and at the same time ensuring each selected topic has sufficient verses for robustness.

Before initiating crowdsourcing, topics in v2, together with random example verses, are presented to three NLP students for feedback, based on which some of the more abstract topics, including *Eschatology*, *Philosophy*, *Theology*, and *Moral*, are removed. On the other hand, more concrete topics, including *Repentance*, *Friendship*, *Thankfulness*, *Forgiveness*, and *Suffering*, are introduced as part of v3. In addition, *Persecution* is renamed to *Heresy* to broaden the coverage. With minor revisions, we submit the verses with topics in v4 for an initial round of crowdsourcing, from which we obtain further feedback. Based on this feedback, adjustments are made in later refinement iterations.

[3] `https://www.propreacher.com/100-sermon-topics`

version	topics	num. topics
v1	*Rules*, *Phenomenon*, *Conflict*, *Relation*, *Place*, *Character*, *Reward*, *Punishment*, *Command*	9
v2	*Eschatology*, *Grace*, *Family*, *Creation*, *Philosophy*, *Revival*, *Cults*, *Compromise*, *Persecution*, *Hospitality*, *Conflicts*, *Theology*, *Morals*, *Commandments*, *Sacrifice*	15
v3	*Creation*, *Grace*, *Violence*, *Conflict*, *Hospitality*, *Sacrifice*, *Heresy*, *Repentance*, *Faith*, *Suffering*, *Forgiveness*, *Thankfulness*, *Friendship*, *Temptation*	14
v4	*Creation*, *Grace*, *Violence*, *Conflict*, *Hospitality*, *Sacrifice*, *Heresy*, *Repentance*, *Faith*, *Suffering*, *Forgiveness*, *Thankfulness*	12
v5	*Creation*, *Commandment*, *Genealogy*, *Violence*, *Sacrifice*, *Money*, *Salvation*, *Sin*	8
v6	*Creation*, *Commandment*, *Genealogy*, *Violence*, *Sacrifice*, *Money*, *Grace*, *Sin*	8
v7	*Recommendation*, *Faith*, *Description*, *Sin*, *Grace*, *Violence*	6

Table 3.1: Versions of topics during each refinement iteration. Version 1 contains the initial set of self-designed topics, developed with the help of a linguist. Version 2 contains topics derived from an online preaching website. Version 3 removes some abstract topics - *Eschatology*, *Philosophy*, *Theology*, and *Morals* - and introduces some new topics - *Repentance*, *Friendship*, *Thankfulness*, *Forgiveness* and *Suffering*. Version 4 is used for crowdsourced annotations on Amazon MTurk. Versions 5 and 6 merge similar topics from Version 4 and adjust the names of several topics to improve clarity. Version 7 is the final version of topics used by the dataset.

3.3.3 Annotation

Following the development of the six final topics, which are treated as classes of the verses, verses belonging to each class are extracted and annotated in a preliminary annotation round by three annotators. We retain only verses whose class labels are agreed upon by at least two annotators, and remove verses that are noisy (covering multiple topics) or irrelevant (not considered as any topic by the annotators). This step is performed to reduce ambiguity for annotators in subsequent crowdsourcing rounds and to control annotation costs. The resulting dataset contains 1077 verses, which are submitted to Amazon Mechanical Turk (MTurk) for annotation, specifying the US as the worker[4] location. Each verse is subsequently annotated ten times by different workers, and the final class label is assigned based on majority voting.

We expect that annotation quality issues may emerge under two circumstances: confusion about the task and the worker's lacking care or attention. To minimize confusion related to the annotation task, we provide detailed guidelines and examples with the survey. Additionally, workers are required to pass a qualification test to demonstrate their

[4]MTurk annotator.

class	definition
Recommendation	An imperative statement which suggests to act or believe in certain ways.
Faith	Display of belief and love toward God, instructions on how to maintain faith, stories of faith and its consequences.
Description	Describes a person, relationship, phenomenon, situation, etc.
Sin	Describes what is considered sin, stories of sinful people, and sinful actions.
Grace	Describes God's love, blessing, and kindness towards humans.
Violence	Describes wars, conflict, threats, and torture; destructions of people, cities, and nations.

Table 3.2: Definitions of classes in Taxi1500.

understanding of the task. For quality control, we implement a performance threshold using "pseudo-gold standard" data by estimating the class labels from the majority votes of all annotators. We calculate the macro F_1 score from each worker's annotations and reject annotations from workers whose F_1 score is below 0.40, republishing their verses for a new round of annotation.

To assess the inter-annotator agreement, we calculate Krippendorff's α (K-α), a metric chosen for its ability to handle missing annotations, which is critical as each worker annotates only a subset of the verses. We obtain an overall K-α of 0.44 on the 1077 verses, which can be raised at the cost of reducing the dataset size. As shown in Table 3.3, higher K-α values can be achieved by raising the threshold of the minimum votes required to assign a majority class label, which however, also significantly decreases the number of available verses. A clear tradeoff between the number of accepted verses and K-α is also demonstrated in Figure 3.1. Considering this tradeoff and the inherent subjectivity of the topics in our dataset, we choose to maintain the K-α without excluding any data. Notably, similar K-α values have been observed in previous work and do not necessarily imply poor data quality (Price et al., 2020).

votes $\geq$	3	4	5	6	7	8	9
num. verses	1077	1055	941	755	563	388	233
K-α	0.44	0.44	0.48	0.55	0.63	0.73	0.83

Table 3.3: A higher threshold for the minimum number of votes required to determine the majority class leads to a higher K-α value but at the same time decreases the number of verses.

Table 3.4 shows an overview of the six classes with their corresponding numbers of verses in the English dataset, including an example verse for each class. Among the 1077

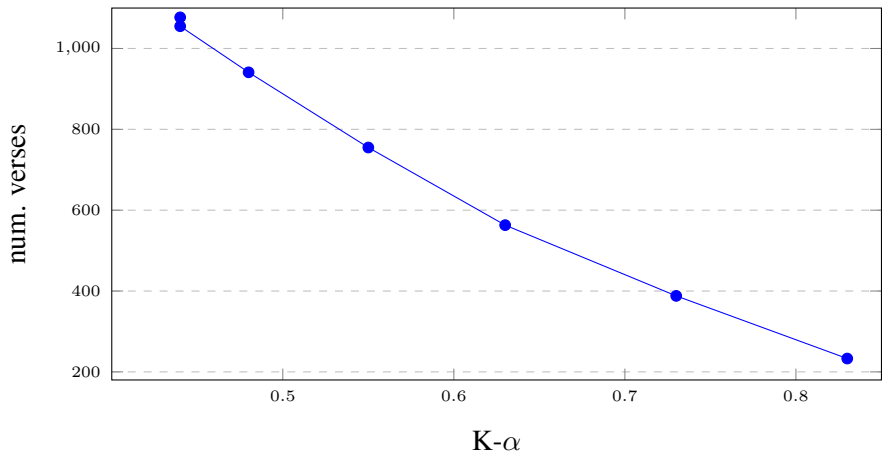


Figure 3.1: Tradeoff between the K-α value, which can be increased through a higher threshold of the minimum number of required votes, and the number of verses. The dots represent the minimum number of required votes $\in \{3, 4, 5, 6, 7, 8, 9\}$ for a verse to be accepted.

verses, *Recommendation* has the highest frequency (281 instances), while *Violence* has the lowest (59 instances). Due to incomplete translations of the New Testament in some languages, these languages have fewer verses than the annotated English dataset. To ensure the consistency of verses among all languages of the dataset, we exclude languages with fewer than 900 of the 1077 annotated verses. This results in a multilingual dataset covering 1504 languages from 113 language families, representing a wide geographical span across the globe [5].

class	example	num. verses
Recommendation	If you love me , you will observe my commandments	281
Faith	Most truly I say to you , whoever believes has everlasting life	260
Description	There was a man of the Pharisees named Nicodemus , a ruler of the Jews	184
Sin	Jesus answered : “ I do not have a demon , but I honor my Father , and you dishonor me	153
Grace	The Father loves the Son and has given all things into his hand	140
Violence	He put James the brother of John to death by the sword	59

Table 3.4: Example verses and the total number of verses for each class in the crowdsourced English dataset.

[5] Family and geographical data sourced from `https://glottolog.org`

3.4 Evaluation

Experiment setup

To demonstrate its utility, we apply Taxi1500 to evaluate four mPLMs: mBERT, XLM-R Base, XLM-R Large, and Glot500 on all languages. We additionally evaluate six LLMs of different sizes: Llama 2 7B, Mistral 7B, and BLOOM 560M, 1B, 3B, and 7B, on a subset of 67 languages. For a fair comparison, we categorize the languages into three subsets: *head languages*, *Glot500-only languages*, and *tail languages*. Head languages refer to the languages covered by the pre-training data of the four mPLMs. Glot500-only languages are exclusively in the pre-training data of Glot500. Tail languages are languages absent from the pre-training data of all four mPLMs.

The experiments consist of three settings: zero-shot, in-language, and three-shot prompting (for LLMs only). Datasets for all 1504 languages are partitioned into training, development, and test sets with an 80/10/10 split ratio. In the zero-shot transfer setting, models are fine-tuned on English training data and tested directly on the target language test set. For in-language learning, models are fine-tuned and tested on the target language data. We further control the training set size $\in \{50, 100, 200, 400, 600, 860\}$, where 860 indicates the full training set, to study the impact of training set sizes on classification performance and to estimate the minimum number of training samples required to achieve acceptable classification results.

Hyperparameter setup

For all evaluated mPLMs, we use the AdamW optimizer with a learning rate of $2e^{-5}$ and a batch size $\in \{16, 32\}$, selected based on the validation performance. Early stopping is applied based on the performance on the development set. All experiments on the mPLMs are computationally efficient and can be completed within minutes on a single GeForce GTX 1080Ti GPU.

3.5 Results and Analysis

Baseline

As a baseline, we train a Bag-of-Words (BOW) classification model on Taxi1500 and present the results in Section A. The results demonstrate very low baseline performance, with most F_1 scores under 0.10, indicating that classifying Taxi1500 data requires robust semantic representation for its languages, which a simple BOW model lacks.

Zero-shot experiments

Figure 3.2 shows three stacked bar charts representing the number of languages that fall into each F_1 interval. Each chart represents one group of languages: head languages, Glot500-only languages, or tail languages. For head languages, Glot500, XLM-R Base, and XLM-R Large have high F_1 scores (0.4-0.8) for 68, 65, and 69 languages respectively, outperforming mBERT, which has only 26 languages within this F_1 range. This contrast may be attributed to the smaller pre-training corpus used by mBERT compared to the other models.

For Glot500-only languages, Glot500 significantly outperforms the other models with 117 languages within the F_1 range of 0.2-0.8, while the other models have fewer than 30 languages within the same range. This distribution is not surprising as these languages are exclusively covered by Glot500's pre-training data.

On tail languages, Glot500 again shows better performance than the other models, achieving F_1 scores of over 0.2 on 70-80 more languages. This suggests that unseen languages potentially benefit through knowledge transfer from related languages in the pre-training data of Glot500, which covers a broader range of languages. Overall, the zero-shot results on Taxi1500 indicate that our dataset effectively highlights the advantage of pre-training models on a broader range of languages. We show the complete zero-shot results on mBERT, XLM-R Base, XLM-R Large, and Glot500 in Section A.

In-language experiments

head lang.	ISO	script	family	tail languages	ISO	script	family
German	deu	Latin	Indo-European	Cherokee	chr	Cherokee	Iroquoian
Basque	eus	Latin	Basque	Gagauz	gag	Latin	Turkic
Hebrew	heb	Hebrew	Afro-Asiatic	Hixkaryana	hix	Latin	Cariban
Japanese	jpn	Japanese	Japanic	Nga La	hlt	Latin	Sino-Tibetan
Kazakh	kaz	Cyrillic	Turkic	Komi-Zyrian	kpv	Cyrillic	Uralic
Korean	kor	Korean	Koreanic	Kumyk	kum	Cyrillic	Turkic
Malayalam	mal	Malayalam	Dravidian	Aringa	luc	Latin	Central Sudanic
Burmese	mya	Burmese	Indo-European	Magahi	mag	Devanagari	Indo-European
Persian	pes	Arabic	Indo-European	Dibabawon Manobo	mbd	Latin	Austronesian
Chinese	zho	Chinese	Sino-Tebietan	Middle Watut	npl	Latin	Uto-Aztecan

Table 3.5: A selection of 20 languages for in-language fine-tuning, 10 head languages (left) and 10 tail languages (right). Languages are shown with their ISO 639-3 codes, writing systems, and language families.

We perform in-language fine-tuning on a set of 20 languages, 10 head languages and 10 tail languages. These languages are carefully curated to represent a diverse set of languages and span 13 language families and 11 writing systems. They include both high-

and low-resource languages, with or without coverage in the mPLMs' pre-training data. Table 3.5 lists the ISO 639-3 codes, writing systems, and families of these languages.

For a concise comparison, results for in-language fine-tuning, compared with the zero-shot transfer performance of each respective languages, are shown for mBERT and XLM-R Base in Tables 3.6 and 3.7. As expected, the in-language performance improves with a larger training set size for both models. For mBERT, zero-shot performance on head languages is comparable to in-language performance with 100 training samples when comparing the average F_1. For XLM-R Base, this number is raised to 400, suggesting that models with more parameters may need more training data to reach comparable zero-shot performance. Additionally, both models consistently perform better on head languages compared to tail languages in zero-shot settings, indicating their stronger generalization capabilities on languages in their pre-training data.

head lang.	training samples 0	50	100	200	400	600	860	tail lang.	training samples 0	50	100	200	400	600	860
deu	0.39	0.20	0.13	0.34	0.42	0.44	0.52	chr	0.05	0.24	0.21	0.29	0.35	0.30	0.35
eus	0.17	0.15	0.12	0.31	0.44	0.46	0.43	gag	0.12	0.21	0.29	0.35	0.39	0.45	0.38
heb	0.36	0.24	0.24	0.36	0.33	0.38	0.41	hix	0.07	0.30	0.27	0.35	0.35	0.39	0.41
jpn	0.39	0.37	0.40	0.32	0.49	0.63	0.66	hlt	0.08	0.16	0.25	0.33	0.34	0.44	0.49
kaz	0.29	0.30	0.36	0.38	0.50	0.48	0.48	kpv	0.08	0.19	0.24	0.45	0.41	0.39	0.46
kor	0.41	0.36	0.36	0.45	0.56	0.50	0.60	kum	0.14	0.28	0.27	0.35	0.37	0.42	0.46
mal	0.09	0.13	0.25	0.25	0.31	0.35	0.34	luc	0.08	0.27	0.23	0.46	0.41	0.45	0.35
mya	0.22	0.32	0.31	0.41	0.41	0.40	0.46	mag	0.19	0.14	0.38	0.38	0.37	0.43	0.34
pes	0.43	0.30	0.36	0.55	0.53	0.52	0.56	mbd	0.08	0.18	0.33	0.36	0.36	0.39	0.42
zho	0.36	0.24	0.46	0.47	0.62	0.54	0.59	npl	0.06	0.21	0.30	0.38	0.39	0.40	0.40
avg.	0.31	0.26	0.30	0.38	0.46	0.47	0.51	**avg.**	0.10	0.22	0.28	0.37	0.37	0.41	0.41

Table 3.6: Zero-shot transfer and in-language fine-tuning results using mBERT on 20 selected languages. These include 10 head languages (left): German, Basque, Hebrew, Japanese, Kazakh, Korean, Malayalam, Burmese, Persian, and Chinese; and 10 tail languages (right): Cherokee, Gagauz, Hixkaryana, Nga La, Komi-Zyrian, Kumyk, Aringa, Magahi, Dibabawon Manobo, and Middle Watut. Under *training examples*, 0 indicates zero-shot, and 860 indicates the full training set.

LLM evaluation

We further evaluate the performance of six LLMs of different sizes on a subset of 64 languages, which represent a diverse set of language families, using three-shot in-context learning. The LLMs explored are Llama 2 7B, Mistral 7B, and BLOOM 560M, 1B, 3B, and 7B. We show detailed results on 64 languages in Table 3.9 and summarize the average scores in Table 3.8. Among the LLMs, Mistral 7B achieves the highest average F_1 score of 0.55. BLOOM 1B performs best among its variations, with an average F_1 of 0.50, while the 560M model has the lowest performance, with an F_1 of 0.46. Interestingly, three-shot prompting of LLMs generally has performance on par

head lang.	training samples 0	50	100	200	400	600	860	tail lang.	training samples 0	50	100	200	400	600	860
deu	0.52	0.16	0.18	0.43	0.49	0.52	0.51	chr	0.09	0.15	0.20	0.15	0.24	0.21	0.28
eus	0.26	0.09	0.26	0.25	0.34	0.37	0.34	gag	0.33	0.17	0.13	0.14	0.45	0.32	0.54
heb	0.15	0.10	0.13	0.18	0.16	0.33	0.35	hix	0.06	0.18	0.17	0.22	0.3	0.43	0.49
jpn	0.62	0.25	0.39	0.53	0.57	0.61	0.68	hlt	0.05	0.14	0.07	0.19	0.40	0.20	0.50
kaz	0.57	0.23	0.35	0.47	0.41	0.55	0.56	kpv	0.09	0.09	0.21	0.23	0.41	0.38	0.53
kor	0.63	0.35	0.55	0.58	0.65	0.53	0.70	kum	0.13	0.13	0.17	0.22	0.27	0.37	0.45
mal	0.07	0.10	0.13	0.22	0.08	0.21	0.24	luc	0.11	0.12	0.11	0.30	0.30	0.39	0.39
mya	0.42	0.18	0.30	0.21	0.45	0.45	0.64	mag	0.38	0.11	0.23	0.41	0.48	0.38	0.51
pes	0.66	0.17	0.55	0.47	0.65	0.64	0.71	mbd	0.11	0.18	0.14	0.25	0.30	0.30	0.38
zho	0.63	0.33	0.49	0.52	0.45	0.51	0.68	npl	0.05	0.14	0.08	0.25	0.41	0.41	0.43
avg.	0.45	0.20	0.33	0.39	0.43	0.47	0.54	**avg.**	0.14	0.14	0.15	0.24	0.36	0.34	0.45

Table 3.7: Zero-shot transfer and in-language fine-tuning results using XLM-R on 20 selected languages. These include 10 head languages (left): German, Basque, Hebrew, Japanese, Kazakh, Korean, Malayalam, Burmese, Persian, and Chinese; and 10 tail languages (right): Cherokee, Gagauz, Hixkaryana, Nga La, Komi-Zyrian, Kumyk, Aringa, Magahi, Dibabawon Manobo, and Middle Watut. Under *training examples*, 0 indicates zero-shot, and 860 indicates the full training set.

with in-language fine-tuning of mPLMs using the full dataset of 860 verses, indicating that LLMs are capable of attaining similar multilingual capabilities compared to mPLMs with much less data.

model	Llama 2	Mistral	BLOOM			
size	7B	7B	560M	1B	3B	7B
avg. acc.	0.45	**0.55**	0.46	<u>0.50</u>	0.48	0.48

Table 3.8: Average three-shot in-context prompting performance across six LLMs of various sizes. Results are measured in accuracy on 64 selected languages.

Evaluation results by language family

Figures 3.3 and 3.4 present the zero-shot transfer and in-language learning results across all languages using XLM-R Base and Glot500, aggregated by language families. For both models, head languages consistently outperform Glot500-only and tail languages across language families. Across the three groups of languages—head, Glot500-only, and tail—Indo-European languages achieve higher performance than other families. This discrepancy can likely be attributed to the higher proportion of Indo-European languages present in the pre-training data of both models. An interesting finding from the zero-shot results detailed in Section A shows that XLM-R Large underperforms the remaining models on most languages. This may be attributed to its larger parameter count compared to the other models, which increases the risk of overfitting on the small dataset used for

language	L 7B	M 7B	B 560M	B 1B	B 3B	B 7B	language	L 7B	M 7B	B 560M	B 1B	B 3B	B 7B
alt_Cyrl	0.44	0.46	0.49	**0.53**	0.48	0.45	lzh_Hani	0.55	**0.66**	0.51	0.56	0.53	0.54
arb_Arab	0.43	**0.62**	0.46	0.53	0.49	0.49	mai_Deva	0.45	**0.62**	0.45	0.52	0.49	0.49
ary_Arab	0.32	**0.56**	0.34	0.43	0.36	0.39	mar_Deva	0.49	**0.56**	0.49	0.49	0.49	0.53
arz_Arab	0.32	**0.54**	0.35	0.44	0.41	0.45	mdy_Ethi	0.40	**0.55**	0.47	0.46	0.45	0.43
asm_Beng	0.46	**0.56**	0.36	0.45	0.49	0.55	mhr_Cyrl	0.47	0.46	0.48	0.50	**0.51**	0.46
azb_Arab	0.40	**0.51**	0.43	0.47	0.41	0.48	mkd_Cyrl	0.52	**0.67**	0.54	0.61	0.57	0.57
bak_Cyrl	0.45	0.49	0.45	**0.51**	0.47	0.49	mya_Mymr	0.45	0.51	0.51	**0.53**	0.41	0.44
bel_Cyrl	0.48	**0.56**	0.46	0.51	0.45	0.49	myv_Cyrl	0.40	0.45	0.36	**0.47**	0.45	0.41
ben_Beng	0.41	**0.58**	0.41	0.48	0.48	0.52	nep_Deva	0.45	**0.67**	0.51	0.58	0.54	0.63
bul_Cyrl	0.45	**0.61**	0.41	0.44	0.47	0.49	npi_Deva	0.51	**0.67**	0.56	0.55	0.59	0.56
che_Cyrl	0.38	**0.42**	0.36	0.41	0.33	0.37	ori_Orya	0.43	0.51	0.51	**0.56**	0.54	0.51
chv_Cyrl	0.43	0.45	0.47	**0.51**	0.42	0.45	ory_Orya	0.44	0.58	0.53	0.51	**0.59**	0.49
ckb_Arab	0.44	**0.48**	0.45	0.47	0.43	0.45	oss_Cyrl	0.49	0.48	0.48	**0.52**	0.47	0.49
cmn_Hani	0.49	**0.61**	0.44	0.54	0.54	0.53	pan_Guru	0.41	0.46	0.44	**0.47**	**0.47**	**0.47**
crh_Cyrl	0.49	**0.57**	0.48	0.49	0.51	0.48	pes_Arab	0.51	**0.65**	0.54	0.50	0.49	0.59
dzo_Tibt	**0.45**	**0.45**	0.42	0.41	0.43	0.41	prs_Arab	0.51	**0.66**	0.57	0.60	0.57	0.56
ell_Grek	0.49	**0.58**	0.44	0.49	0.49	0.49	rus_Cyrl	0.49	**0.58**	0.43	0.47	0.45	0.51
fas_Arab	0.49	**0.67**	0.53	0.53	0.49	0.58	sah_Cyrl	0.41	0.46	**0.49**	**0.49**	0.46	0.44
guj_Gujr	0.46	**0.52**	0.46	0.48	0.51	**0.52**	sin_Sinh	0.40	0.38	0.41	**0.47**	0.42	0.40
hin_Deva	0.51	**0.65**	0.55	0.48	0.47	0.49	snd_Arab	0.44	**0.62**	0.54	0.56	0.49	0.57
hne_Deva	0.56	**0.61**	0.56	**0.61**	0.58	0.54	suz_Deva	0.47	0.43	0.42	**0.48**	0.45	0.42
hye_Armn	0.46	**0.55**	0.46	0.52	0.52	0.46	tam_Taml	0.44	**0.60**	0.55	0.55	**0.60**	0.59
kat_Geor	0.41	**0.45**	0.43	**0.45**	0.43	0.42	tat_Cyrl	0.48	**0.53**	0.43	**0.53**	0.48	0.46
kaz_Cyrl	0.49	**0.55**	0.45	0.51	**0.55**	0.51	tel_Telu	0.33	**0.54**	0.39	0.52	0.51	0.51
khm_Khmr	0.52	**0.56**	0.52	**0.56**	0.52	0.49	tgk_Cyrl	0.42	**0.56**	0.46	0.54	0.48	0.49
kir_Cyrl	0.51	0.53	**0.62**	**0.62**	0.57	0.48	tha_Thai	0.43	**0.58**	0.45	0.47	0.41	0.43
kjh_Cyrl	0.44	0.48	0.42	**0.49**	0.42	0.45	tir_Ethi	0.30	0.40	0.38	**0.41**	0.32	0.28
kmr_Cyrl	0.40	0.40	0.39	0.44	0.43	**0.45**	tyv_Cyrl	0.39	**0.48**	0.36	0.45	**0.48**	0.43
kor_Hang	0.49	**0.72**	0.49	0.51	0.52	0.49	udm_Cyrl	0.37	0.41	0.42	**0.45**	0.43	0.42
krc_Cyrl	0.46	**0.55**	0.45	0.49	0.46	0.49	ukr_Cyrl	0.52	**0.63**	0.51	0.49	0.49	0.51
ksw_Mymr	0.44	0.44	0.40	**0.49**	0.42	0.42	uzn_Cyrl	0.46	**0.59**	0.43	0.49	0.43	0.45
lao_Laoo	0.45	0.45	0.48	0.51	**0.57**	0.47	yue_Hani	0.43	**0.63**	0.46	0.54	0.53	0.53

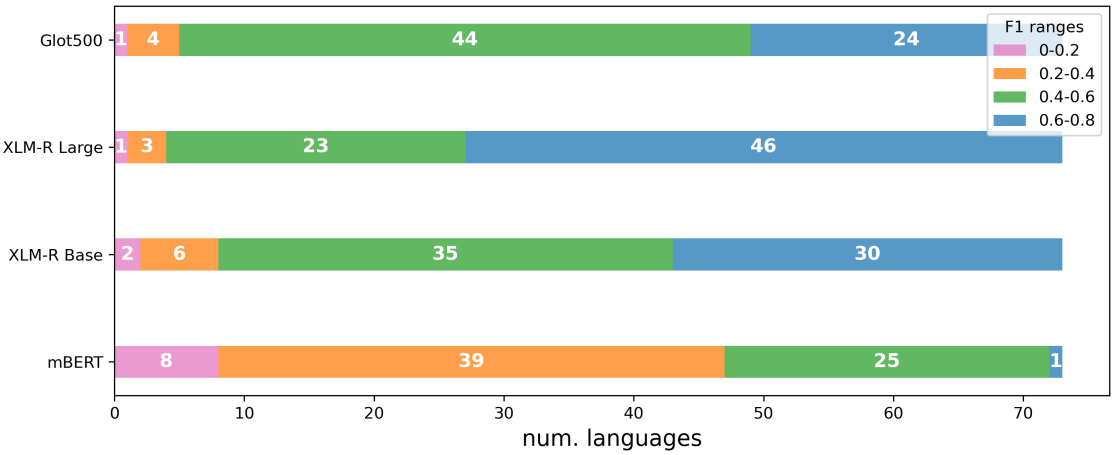
Table 3.9: Performance of three-shot in-context prompting across six LLMs of different sizes on 64 selected languages. L: Llama 2, M: Mistral, B: BLOOM.

its evaluation. Furthermore, when comparing zero-shot and in-language performance of XLM-R Base, extremely low-resource languages with non-Latin writing systems, such as Yawa-Saweru, Lengua-Mascoy, and Hmong-Mien, exhibit more notable performance boosts (around 0.4) under the in-language learning setting. This is an indication that the model is not equally effective for non-Latin script languages and Latin script languages.

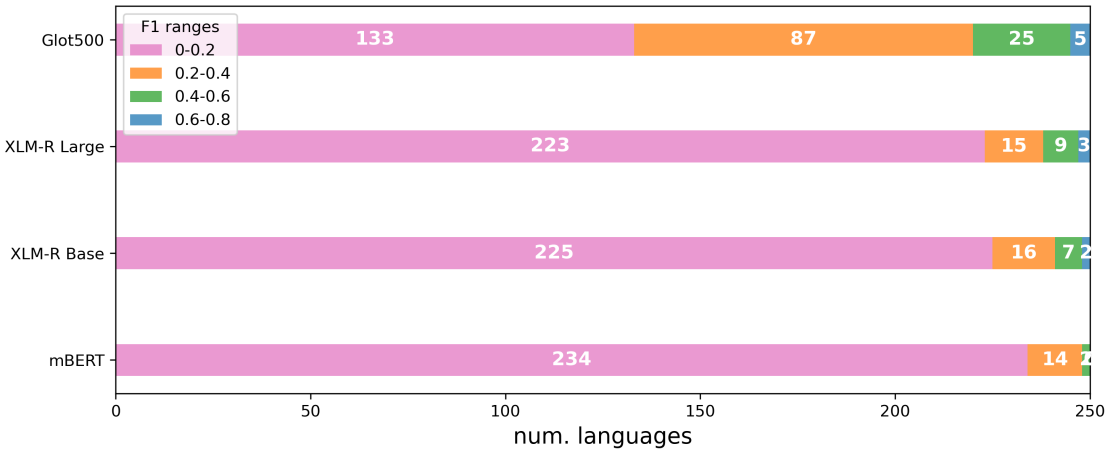
3.6 Conclusion

Evaluation of mPLMs and multilingual LLMs is often constrained by the limited annotated datasets for low-resource languages, which constitute the majority of the world's languages. The limitation is demonstrated by the under-evaluation of many multilingual models, often on only a fraction of their supported languages. One leading reason is that annotating data for every language is not only prohibitively expensive but also impractical

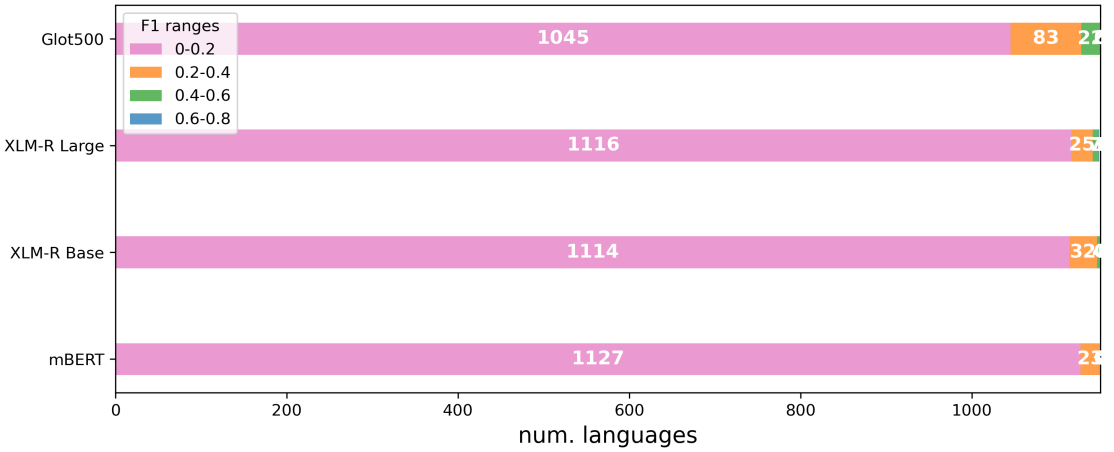
due to the limited availability of resources and annotators for many languages. To tackle this challenge, we introduce Taxi1500, a text classification dataset comprising annotated Bible verses in 1504 languages. We obtain labels for English verses through crowd-sourcing and subsequently project these labels to other languages leveraging the parallel nature of Bible verses. We demonstrate the utility of Taxi1500 through comprehensive evaluations of four mPLMs with varying language coverages and six LLMs of different sizes. The results illustrate that Taxi1500 can serve as an effective benchmark dataset for evaluating the multilingual capabilities across different models.



(a) Head languages.



(b) Glot500-only languages.



(c) Tail languages.

Figure 3.2: Zero-shot performance of mBERT, XLM-R Base, XLM-R Large, and Glot500, with numbers of languages in each F_1 interval shown. Each subfigure shows performance for head, Glot500-only, and tail languages, respectively.

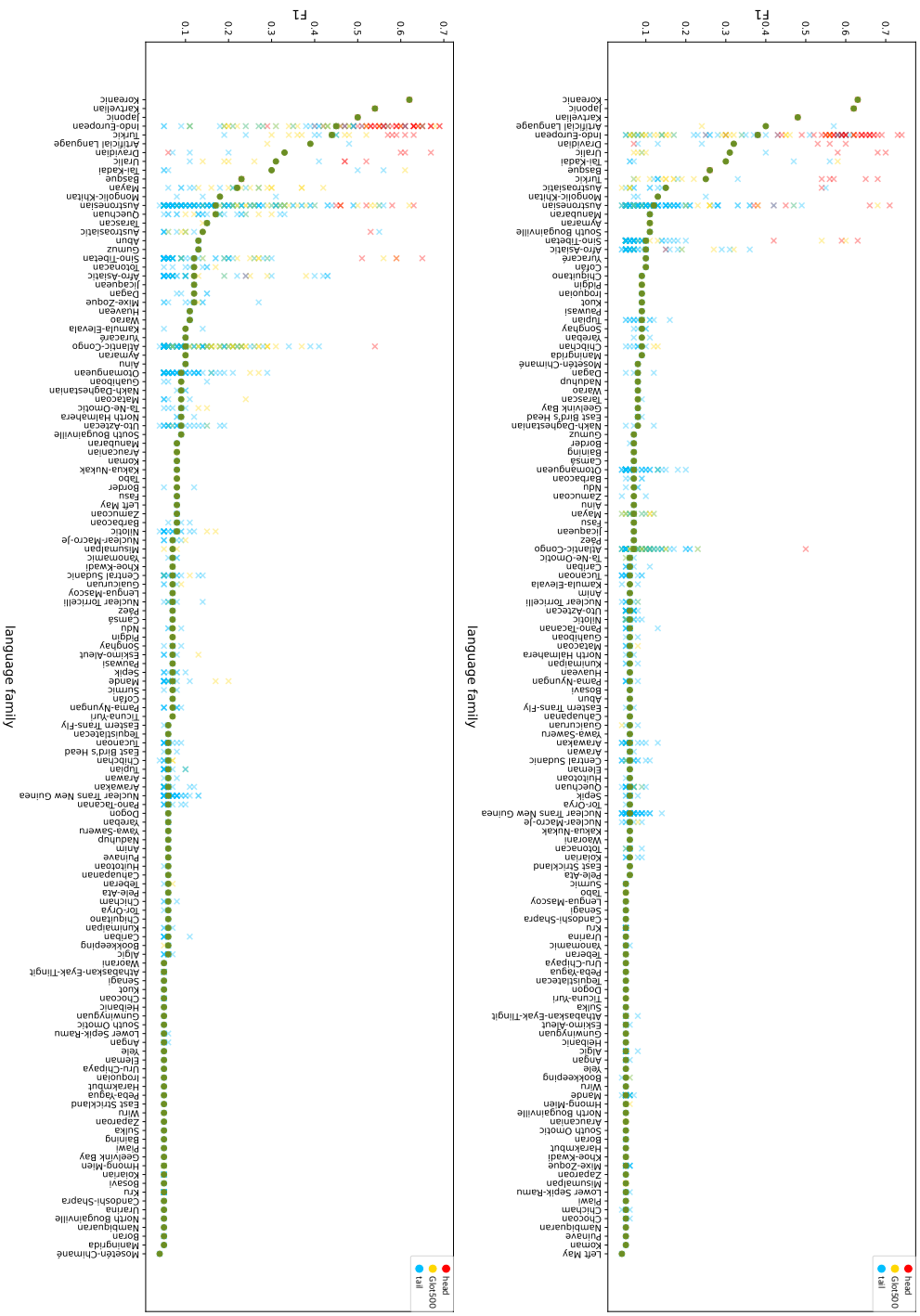


Figure 3.3: Zero-shot transfer performance of XLM-R Base (top) and Glot500 (bottom) aggregated per language family, with the families sorted by their average F_1. Crosses represent individual languages and the green dot represents the family average. Colors of the crosses indicate the types of language: head, Glot500-only, and tail.

Figure 3.4: Zero-shot transfer performance of XLM-R Base (top) and Glot500 (bottom) aggregated per language family, with the families sorted by their average F_1. Crosses represent individual languages and the green dot represents the family average. Colors of the crosses indicate the types of language: head, Glot500-only, and tail.

Chapter 4

Conceptual Language Similarity

This chapter corresponds to the following work:

> **Haotian Ye**, Yihong Liu, Hinrich Schütze (2023). A study of conceptual language similarity: comparison and evaluation.

Declaration of Co-Authorship. The notion of dividing up languages based on their conceptualizations was originally conceived by Yihong Liu in a publication prior to this, on which I also collaborated. Building on the foundation of conceptual language similarity, proposed in the previous publication, I conducted all experiments and analyses for the project which is discussed in this chapter. I completed the initial draft for the work discussed in this chapter, which was subsequently reviewed by all co-authors. All other co-authors also provided feedback on the draft.

4.1 Introduction

More than 7000 languages are spoken in the world today, which are grouped into over 400 distinct language families (Joshi et al., 2020; Hammarström et al., 2022). The growing availability of unlabeled data in a large number of languages in diverse forms has significantly facilitated its processing and usage by machine learning algorithms, contributing to the progress in multilingual NLP. This has been demonstrated by the development of mPLMs (Conneau et al., 2020; Xue et al., 2021; Imani et al., 2023). The advancements, however, have largely excluded the majority of the world's low-resource languages, mainly due to data scarcity for these languages. To address this challenge and improve support for low-resource languages, a number of approaches have been proposed to leverage linguistic information from high-resource languages, such as English and Chinese, to benefit less-represented languages. One prominent approach involves cross-lingual transfer learning, which is detailed in Section 1.1.2. Language similarity plays an important role in the success of such methods, as linguistically similar languages have been demonstrated to enhance the performance of transfer learning (Kim et al., 2017; Ahmad et al., 2019; Lauscher et al., 2020) and joint learning (Cohen et al., 2011; Navigli and Ponzetto, 2012; Wang et al., 2021). Furthermore, Chronopoulou et al. (2023) leverage typological information to create language family-specific adapters for groups of similar languages, and Gerz et al. (2018) show that typological features exert a strong impact on the performance of mPLMs on diverse languages.

While most language similarity measures rely on lexical or typological features, such as word order and verbal inflection, recent work has established a novel definition of language similarity based on the representation of basic concepts in each language. This type of *conceptual language similarity* is shown to be complementary to existing lexical or typological similarities, which often categorize languages based on geographical (e.g., the continent of the language), phylogenetic (genealogical relationships), or structural (e.g., syntax- or grammar-related) features. The primary sources of typological features are manually constructed databases containing curated features, such as Glottolog (Hammarström et al., 2022), PHOIBLE (Moran and McCloy, 2019), WALS (Dryer and Haspelmath, 2013), and Grambank (Skirgård et al., 2023). Alternatively, some approaches implement automatic inference of typological features, for example, through word alignment (Mayer and Cysouw, 2012; Östling, 2015), in the cases that the coverage of existing databases is inadequate.

Unlike the aforementioned similarity measures, we focus on conceptual language similarity, introduced in our previous work (Liu et al., 2023b). Conceptual similarity relies on *Conceptualizer*, a two-step framework that aligns basic concepts across 1335 languages leveraging the Parallel Bible Corpus (PBC) (Mayer and Cysouw, 2014). The idea behind Conceptualizer is founded on the assumption that languages divide the world into concepts and associate with them in diverse ways, and that such divergence can be leveraged to capture similarities and differences in languages. For instance, three

geographically and culturally related languages, Chinese, Japanese, and Korean, share a common association of the "mouth" concept with "entrance", which is influenced by the Chinese character "口". In contrast, this association is absent in most European languages, a phenomenon that reflects a conceptual divergence between the three East Asian languages and European languages with respect to the "mouth" concept. A belief grounded in previous research shows a link between a language's conceptualization patterns and its speakers' thoughts (Deutscher, 2010). In a similar fashion, conceptual similarity provides a novel way to quantify language similarity by reflecting a perspective that is complementary to conventional similarity measures based on lexical or typological features.

We divide the content of this chapter into two parts. (1) We elaborate on our previous work (Liu et al., 2023b) and conduct a deeper investigation into the conceptual similarity, and (2) extend our previous work by extensively evaluating different language similarity measures and comparing conceptual similarity to existing measures. Specifically, we evaluate language similarity measures on a binary classification task to predict whether most of a language's neighbors belong to the same language family. To the best of our knowledge, no prior study has carried out empirical evaluations and comparisons on different language representations for predicting genealogical language similarity. Our findings show that, in terms of classification accuracy, conceptual similarity does not outperform existing similarities based on lexical or typological features. However, from a linguistic perspective, it provides valuable insights by highlighting the similarities and divergence in the conceptual patterns of languages, which makes it a valuable tool alongside existing language similarity measures.

4.2 Related Work

Substantial research has been dedicated to the study of language similarity, the majority of which leverages lexical or typological features. We present some common categories of language similarity measures and existing works on the phenomenon of *colexification*, which forms the foundation of the Conceptualizer framework.

4.2.1 Lexical Similarity

Lexical similarity is a surface similarity measure commonly used to assess the level of similarity between two languages and whether they may be considered dialects. Notably, it is used by *Ethnologue*, which considers a language variant with a lexical similarity of greater than 85% potential dialects (Eberhard et al., 2024).

Lexical similarity is typically measured using multilingual lexicostatistical lists, such as the PanLex Swadesh list (Kamholz et al., 2014), which contains 100 words describing basic concepts across over 2000 languages. It also provides an extended 207-word

version of the Swadesh list (Swadesh, 2017), available in fewer languages. Larger lists, such as NorthEuraLex (Dellert et al., 2020), which covers 1016 concepts in 107 languages, have also been used (Rama et al., 2020). However, these resources generally have much more limited language coverage compared to the PanLex Swadesh list.

Further efforts have explored possibilities to optimize the Swadesh-100 list. For instance, Holman et al. (2008) propose a shortened Swadesh-100 list consisting of only the 40 most stable concepts, which is shown to increase the accuracy of language classification. The refined list is incorporated into the ASJP database, which contains the list in 5590 languages (Søren et al., 2022). Using ASJP, Östling and Kurfali (2023) evaluate lexical distances between 1012 languages by calculating the mean normalized Levenshtein distance between each concept pair. Alternatively, the pairwise Levenshtein distance can be replaced by a simple longest common substring method, which effectively quantifies the shared lexical information between two languages.

4.2.2 Genealogical Similarity

Genealogical similarity is measured based on the positions of two languages within a genealogical or phylogenetic language tree. Without considering the further branching of the over 400 top-level language families, the simplest measure of genealogical similarity is a binary indicator of whether two languages belong to the same top-level family, such as Indo-European or Sino-Tibetan. We can assign a similarity of 1 if they belong to the same family or 0 otherwise. This metric can be further refined by introducing intermediate levels of the language tree. For example, the two paths below illustrate the complete genealogical hierarchies of Hungarian (hun) and Estonian (ekk), with all tree levels shown (data from Glottolog).

hun: Uralic → Hungarian
ekk: Uralic → Finnic → Coastal Finnic → Neva → Central Finnic → Estonian

Based on such hierarchical paths, one approach to quantifying language similarity is to treat each level in the path as a node and compute the Jaccard index between the two paths. Alternatively, a more refined metric can be devised based on the number of edges from the leaf node to the lowest common ancestor node (“Uralic” in this case).

However, both methods face a limitation posed by the uneven depths of language trees, with some language families featuring more fine-grained subdivisions and therefore having more nodes in their paths. As a result, languages with shallower paths tend to appear more similar to others due to fewer subdivisions in their paths, which means fewer divergent nodes.

4.2.3 Typological Similarity

Many widely used language similarity measures rely on typological features. One comprehensive resource for such features is the WALS database (Dryer and Haspelmath, 2013), which contains binary encodings of around 200 typological features for 2662 languages. The features span several categories, including phonology, lexicon, and word order. The URIEL (Littell et al., 2017) database extends the set of typological features by incorporating phylogenetic and geographic features. It integrates features sourced from multiple typology databases: syntax features from WALS and SSWL (Collins and Kayne, 2009), phonology features from WALS and Ethnologue (Eberhard et al., 2024), and phonetic inventory features from PHOIBLE (Moran and McCloy, 2019). To address the issue of missing features in some of these databases, URIEL employs a weighted k-nearest-neighbors algorithm to infer their values with high accuracy. In addition, a toolkit named lang2vec[1] is released to facilitate access to the URIEL database. Another prominent typology database is Grambank (Skirgård et al., 2023), which is the largest grammatical database to date and comprises 195 features for 2467 languages and dialects. Compared to previous typology databases, Grambank offers a more systematic and comprehensive feature set, including features associated with cognition and cultural nuances, such as the distinction of politeness levels in the second person. A key strength of Grambank is its high feature coverage, with only 24% missing values, a much lower number compared to other typology databases.

4.2.4 Representational Similarity

Several studies have explored the use of dense word or language representations to compute language similarities. Conneau and Lample (2019) incorporate language embeddings into their XLM model to enhance its performance on machine translation. These language embeddings are, however, learned during large-scale pre-training and are limited to specific language pairs, making the approach impractical to extend to thousands or even hundreds of languages. Yu et al. (2021) train language embeddings for 29 languages using denoising autoencoders, which remains a small set of languages. Rama et al. (2020) investigate language distances leveraging representations from mBERT and fastText embeddings (Bojanowski et al., 2017) by calculating the average pairwise distances between word vectors from a multilingual word list. However, due to the limited language coverage of mBERT and fastText, this method is also restricted in its scalability.

[1] `https://github.com/antonisa/lang2vec`

4.2.5 Colexification

Conceptualization is closely tied to the notion of colexification, which is defined by François (2008) as the phenomenon where two concepts are associated with the same lexical form in a given language. In linguistics, colexification has been applied for constructing semantic maps (Haspelmath, 2003) and analyzing cross-lingual polysemies (Perrin, 2010; List et al., 2013), among other tasks. Additionally, it has been explored in practical applications, such as analyzing variations in semantic networks, for example, in relation to emotions (Jackson et al., 2019; Thompson et al., 2020).

Most existing colexification datasets rely on manual curation or annotation, including BabelNet (Navigli and Ponzetto, 2010), CLICS (Rzymski et al., 2020), and Concepticon (List et al., 2024). However, Liu et al. (2023b) introduce the first approach for automatic identification of colexification patterns by leveraging unannotated parallel text corpora, distinguishing it from previous work in the field.

4.3 Conceptualizer

Conceptualizer, introduced in our previous work (Liu et al., 2023b), is a pipeline designed to measure language similarity based on conceptualization patterns in 1335 languages available in the PBC. The pipeline is built by first selecting 83 concepts, 32 from the Swadesh-100 list (Swadesh, 2017) and 51 derived from the Bible. They are selected based on the following criteria. For Swadesh concepts, we select those with a frequency between 5 and 500 occurrences in both the New Testament and the Hebrew Bible, as these books cover the majority of languages in the PBC. For Bible concepts, we first extract strings of between 4 and 15 characters from the English New World Translation (1984) Bible[2]. The chosen strings must fulfill a certain coverage across 12 other randomly selected languages (the algorithm described below). We then filter out named entities and include only nouns specific to the Bible contexts and are not already included in the Swadesh list. Table 4.1 shows an overview of the 83 selected concepts.

Throughout this chapter, we adopt the convention of denoting a 'concept' with single quotation marks and a concrete "word" or "string" with double quotation marks. Figure 4.1 illustrates the Conceptualizer pipeline, which is described in detail below. Using English as the source language, we construct a set of source nodes $\mathcal{S}$, each representing a concept (e.g., 'belly') as a set of strings in English corresponding to the concept (e.g., {`$belly$`, `$bellies$`}), where `$` denotes word boundaries. We then define target nodes $\mathcal{T}$ as triplets comprising a target language l, a verse ID, and a set of correlated strings in l.

We implement the concept aligning pipeline by constructing a directed bipartite graph $\mathcal{G} \subset \mathcal{S} \times \mathcal{T} \cup \mathcal{T} \times \mathcal{S}$, which is shown in Figure 4.1. This process consists of two steps, a

[2]This edition is chosen because it has the largest number of verses.

Swadesh concepts		Bible concepts		
‘fish’	‘knee’	‘babe’	‘peace’	‘generation’
‘bird’	‘belly’	‘hypocrit’	‘secret’	‘contrary’
‘dog’	‘neck’	‘soldier’	‘faith’	‘prophesy’
‘tree’	‘breast’	‘scroll’	‘woe’	‘decision’
‘seed’	‘sun’	‘demon’	‘throne’	‘request’
‘leaf’	‘moon’	‘boat’	‘wisdom’	‘weakness’
‘root’	‘star’	‘olive’	‘disciple’	‘journey’
‘flesh’	‘water’	‘prayer’	‘obeisance’	‘public’
‘blood’	‘rain’	‘mercy’	‘truth’	‘appearance’
‘horn’	‘stone’	‘trumpet’	‘memor’	‘expression’
‘hair’	‘cloud’	‘angel’	‘governor’	‘marriage’
‘ear’	‘smoke’	‘prison’	‘poor’	‘wrath’
‘mouth’	‘path’	‘savior’	‘blind’	‘trouble’
‘tooth’	‘mountain’	‘tomb’	‘spiritual’	‘promise’
‘tongue’	‘white’	‘husband’	‘justice’	‘power’
‘foot’	‘night’	‘bride’	‘courage’	‘pleasure’
		‘talent’	‘purpose’	‘thought’

Table 4.1: A total of 83 concepts are selected for building the Conceptualizer pipeline, comprising 32 from the Swadesh-100 list and 51 derived from the Bible. Concepts are selected based on their frequency in the PBC.

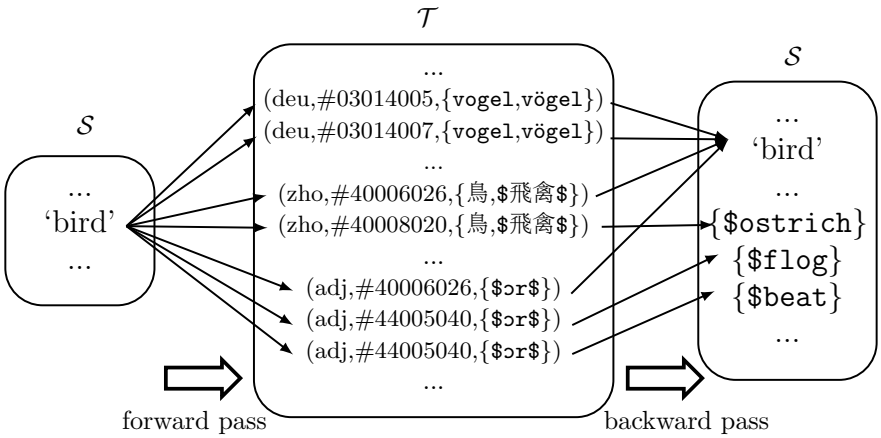


Figure 4.1: Example figure from Liu et al. (2023b) illustrating the directed bipartite graph that forms the base of the Conceptualizer pipeline. The figure shows the alignment process for the concept ‘bird’. Each node in $\mathcal{S}$ is a set of strings representing the concept (e.g., {$bird$, $birds$}). Each node in $\mathcal{T}$ represents a triplet of a language, a verse ID, and a set of correlated strings.

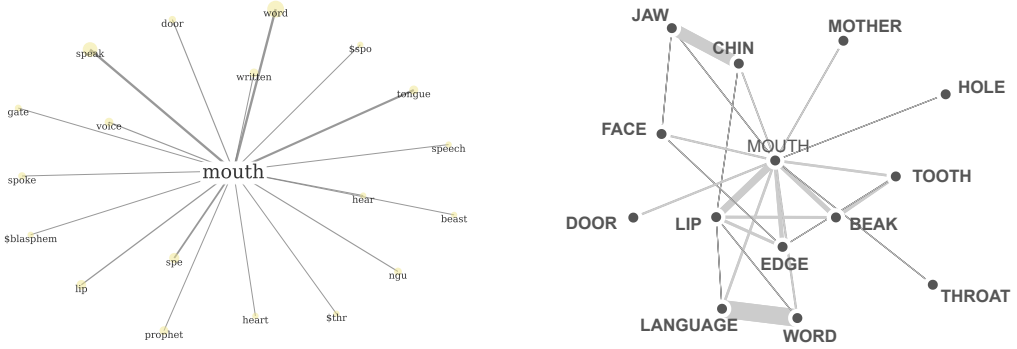


Figure 4.2: Visual representations of colexification patterns for the concept ‘mouth’. The illustration on the left shows colexifications for languages of the PBC, identified by Conceptualizer. Connections between the nodes represent colexifications and the thickness of the edges indicates the number of languages sharing the connection. On the right is the colexification network for ‘mouth’ from CLICS. Both graphs share some associations, such as those to “word” and “door”.

forward pass (FP) and a backward pass (BP). For a given concept (e.g., ‘belly’), we first define V as the set of verses containing any of the source language strings representing it (e.g., {`$belly$` or `$bellies$`}). In the forward pass, the pipeline iteratively searches for target language strings t with the highest correlation to V measured by the χ^2 score $\chi^2(l, t, V)$. The search continues until a threshold α, defined as the fraction of V covered by the most strongly correlated n-grams, is reached. The backward pass is essentially the reversed process of the forward pass and identifies the most strongly correlated strings in the source language given $\mathcal{T}$. Notably, although the PBC data is pre-tokenized, the Conceptualizer pipeline does not rely on explicit word boundaries for identifying relevant strings and can technically cross word boundaries. Comparing results from backward passes of different l’s thus allows us to identify languages with similar or divergent colexification patterns.

We note that our pipeline is strongly linked to the linguistic phenomenon of colexification, which is discussed in Section 4.2.5. Figure 4.2 presents conceptual alignments of ‘mouth’ obtained through Conceptualizer alongside a corresponding colexification network for the same concept from CLICS (Rzymski et al., 2020), a database of cross-lingual colexifications. Both graphs, for example, demonstrate associations of ‘mouth’ with strings such as “word” and “door”, which are shown in the form of connecting edges.

4.4 Conceptual Similarity

To measure language similarity, we represent each language as a concatenation of 83 vectors, where each vector corresponds to one of the 83 concepts:

$$\vec{v}(l) = [\vec{v}(l, F_1); \vec{v}(l, F_2); \ldots; \vec{v}(l, F_{83})]$$

In the above equation, F_j denotes a concept, and $\vec{v}'(l, F_j)$ is a 100-dimensional vector. Each dimension, $\vec{v}'(l, F_j)_i$, represents the number of paths from F_j to an English n-gram e_i, considering only nodes $c = (l', v, T)$ where $l' = l$, i.e., nodes specific to l. Formally, $\vec{v}'(l, F_j)_i$ is defined as:

$$\vec{v}'(l, F_j)_i = |\{c|(F_j, c) \in \mathcal{G} \wedge (c, \{e_i\}) \in \mathcal{G}\}|$$

The first dimension, $\vec{v}'(l, F_j)_1$, always corresponds to the queried concept itself. The remaining 99 dimensions represent English n-grams e_k most frequently associated with F_j across languages. The final vector is normalized by $\sum_k \vec{v}'(l, F_j)_k$. For example, for the concept 'mouth', the FP identifies a connection between 'mouth' and the Chinese string '口'. In BP, the retrieved string "口" not only associates with the string "mouth" but also with "entrance". Thus, the first dimension of $\vec{v}'(l,$ 'mouth'$)$ represents the number of paths between 'mouth' and "mouth", while another dimension represents the number of paths between 'mouth' and "entrance".

Using these language vectors, metrics such as cosine similarity can be used to quantify conceptual relatedness and group languages sharing similar conceptualization patterns. We describe our process of evaluating conceptual language similarity in Section 4.5. Our findings, which are elaborated in more detail in Section 4.6, reveal that conceptual similarity complements geographical and genealogical closeness traditionally used to describe the proximity of languages. For instance, Plateau Malagasy, an Austronesian language spoken in Madagascar, shows similarities in conceptualization patterns not only with Hawaiian, its geographically distant Austronesian relative, but also with geographically adjacent but genealogically distinct Atlantic-Congo languages like Mwani and Koto, which are spoken in the neighboring countries of Madagascar. Similarly, Masana, an Afro-Asiatic language spoken in Nigeria, shows conceptual similarities with neighboring languages Yoruba, Igbo, and Twi, despite the letter three belonging to a different language family. As shown in Table 4.2, all four languages associate 'hair' with "wool" and 'mouth' with "entrance".

We find that historical influences, for example resulting from trade, cultural exchanges, and colonization, also emerge as factors affecting conceptual similarity. Table 4.3 provides two examples illustrating this. In the 'mouth' example, three East Asian languages that share a close historical background, Japanese, Korean, and Chinese, also share a conceptual association between 'mouth' and "entrance", a pattern that is absent in languages like French. This likely reflects the historical influence of the Chinese character

Concept	Lang.	n-grams
'tree'	yor	$tree$, $trees$, wood, $stake, $frankincense$, $thornbush, $palm-tree$
	ibo	$tree$, $trees$, $pole, wood, $impal, $stake, $panel
	mcn	$tree$, $trees$, wood, $stake, $impale, $cedar, $timber
	twi	$tree$, $trees$, $wood, $panel$, $pole, $figs$, $timber
'hair'	yor	$hair$, $hairs$, $wool$
	ibo	$hair$, $hairs$, $wool, $shear, $beard
	mcn	$hair$, $hairs$, $wool$, $shave, $baldness$, $shear, goat
	twi	$hair$, $hairs$, $beard, $shave, $head$, $wool
'mouth'	yor	$mouth$, $mouths$, $entrance, $kiss, $palate$, $marvel, $suckling
	ibo	$mouth$, $mouths$, $gate, $entrance, $lip, curse, $precious$
	mcn	$mouth$, $mouths$, $lips$, fulfill, $denie, $disown, $entrance
	twi	$mouth$, $mouths$, $gat, $collect, $lip, $entrance, $registered$

Table 4.2: Comparison of conceptual associations with three concepts in four African languages. yor: Yoruba, ibo: Igbo, mcn: Masana, twi: Twi.

Concept	Target lang.	Translations in target lang. (in English)
'mouth'	jpn	口 (mouth, opening, entrance)
	kor	구(口) (entrance, gate, mouth)
	zho	口(mouth, gate, entrance), 嘴(mouth, lips)
	fra	bouche (mouth)
'tongue'	spa	lengua (tongue, language)
	tgl	dilà (tongue, language), wikà (tongue, language)
	ceb	dila (tongue), pinulongan (tongue, language)
	msa	lidah (tongue), oojoo leeda (tongue)

Table 4.3: Comparisons of conceptualization patterns for two concepts, 'mouth' and 'tongue'. The three East Asian languages - jpn: Japanese, kor: Korean, zho: Chinese - share a common conceptual association between 'mouth' and "entrance", a pattern absent in French (fra). Two Philippine languages - tgl: Tagalog, ceb: Cebuano - demonstrate similar conceptualizations of 'tongue' as "language", likely due to Spanish (spa) influence, while another Austronesian language, Standard Malay (msa), does not.

"口". In another example, certain Philippine languages display a similar conceptualization pattern to Spanish for the 'tongue' concept, whereas Standard Malay, another Austronesian language, does not. This divergence can be explained by the influence of Spanish colonization in the Philippines, which has likely shaped conceptualization patterns in these languages.

4.5 Evaluation

In this section, we evaluate four language similarity and distance measures, including the proposed conceptual language similarity. **Conceptual cosine similarity**, as introduced in Liu et al. (2023b), quantifies the similarity between two languages using the cosine

similarity of their conceptual vectors (explained in Section 4.4). **Conceptual Hamming distance** measures the distance between two languages as the number of differing elements in their binarized conceptual vectors. Compared to cosine similarity, we apply binarization to enhance the interpretability of conceptual representations by weighting all dimensions equally and highlighting different dimensions. Östling and Kurfali (2023) calculate **lexical distances based on ASJP word lists** employing mean normalized Levenshtein distance. We evaluate language similarity using the distance matrix provided. Finally, we evaluate **typological distance** based on syntactic, phonological, and phonetic inventory features from the URIEL database (Littell et al., 2017). An overview of the evaluated measures is provided in Table 4.4.

Following Liu et al. (2023b), we evaluate conceptual language similarity and compare it against the other measures using a binary language family classification task. This task determines whether the majority of a language's k nearest neighbors belong to the same family. Using data from Glottolog (Hammarström et al., 2022), we construct a language tree with its genealogical hierarchies. We focus on the six top-level language families with at least 50 languages in the PBC for stable results, which are Atlantic-Congo (ATLA), Austronesian (AUST), Indo-European (INDO), Nuclear Trans New Guinea (GUIN), Otomanguean (OTOM), and Sino-Tibetan (SINO). The classification accuracy results for all measures evaluated are shown in Tables 4.5 and 4.6. It is important to note that classification accuracy under this setting reflects how well a specific similarity or distance measure aligns with the languages' genealogical relationships. For conceptual similarity measures specifically, the results do not reflect how effectively they capture true conceptual similarity, as conceptually similar languages may not always belong to the same family.

4.5.1 Conceptual Cosine Similarity

In Liu et al. (2023b), cosine similarity is used to compare conceptual vectors of languages, which are obtained by concatenating concept vectors. Specifically, we devise three subsets of concepts: 32 Swadesh concepts only, 51 Bible concepts only, and all 83 concepts. Table 4.5 shows classification accuracy on these concatenations. For most families, accuracy improves with an increasing number of neighbors (k) up to 8, after which it is likely reduced by noise from other families. Conceptual similarity achieves high accuracy for ATLA and INDO families (.80 and .87), with INDO performing the best, possibly due to English being used as Conceptualizer's source language. This likely makes associations to INDO languages more easily retrieved during BP. For AUST, GUIN, and OTOM families, accuracy is around 50% in about half of the cases. SINO performs the worst, indicating a low level of conceptual similarity within the family. Large differences in accuracy between Swadesh and Bible concepts can occasionally be observed, particularly for INDO and OTOM families, indicating that the abstractness of Bible concepts can lead to variable results.

Measure	Similarity/Distance	Explanation
Conceptual cosine similarity (4.5.1)	Similarity	Measures the similarity between two languages by computing the cosine similarity between their conceptual vectors. Captures the degree to which conceptual patterns of two languages overlap (Liu et al., 2023b).
Conceptual Hamming distance (4.5.2)	Distance	Calculates the number of differing dimensions between binarized conceptual vectors. Assigns equal weight to all dimensions to enhance interpretability and highlight differences across conceptual dimensions.
Lexical distance based on ASJP (4.5.3)	Distance	Computes surface-form language distance using mean normalized Levenshtein distance based on aligned word lists of basic vocabulary from ASJP (Östling and Kurfali, 2023).
Typological distance based on URIEL (4.5.4)	Distance	Computes language distance based on syntactic, phonological, and phonetic inventory features from the URIEL database. Integrates multiple typological datasets into unified representations (Littell et al., 2017).
Typological distance based on Grambank (4.5.5)	Distance	Computes language distance based on typological features from Grambank, the largest grammatical database to date. Grambank provides a more systematic and comprehensive feature set compared to earlier typological databases (Skirgård et al., 2023).
Chinese-based conceptual measures (4.7)	Similarity/Distance	Computes conceptual cosine similarity 4.5.1 and Hamming distance 4.5.2. Conceptual vectors are generated using Chinese as the Conceptualizer source language.
Korean-based conceptual measures (4.7)	Similarity/Distance	Computes conceptual cosine similarity 4.5.1 and Hamming distance 4.5.2. Conceptual vectors are generated using Korean as the Conceptualizer source language.

Table 4.4: An overview of the similarity and distance measures evaluated in this chapter, including conceptual, lexical, and typological approaches, with brief explanations for each.

k	# concepts	ATLA	AUST	INDO	GUIN	OTOM	SINO	All
2	32	.21	.20	.53	.09	.14	.00	.13
	51	.24	.19	.26	.08	.04	.03	.11
	83	.29	.31	.49	.11	.14	.04	.17
4	32	.54	.41	.80	.24	.39	.15	.29
	51	.52	.45	.48	.18	.12	.09	.24
	83	.63	.51	.77	.31	.28	.09	.32
6	32	.63	.49	.85	.30	.43	<u>.16</u>	.33
	51	.64	.57	.57	.20	.13	.13	.30
	83	.74	<u>.60</u>	.83	.40	.37	.12	.37
8	32	.68	.53	**.87**	.34	<u>.51</u>	**.18**	.36
	51	.71	.59	.60	.22	.14	.15	.32
	83	<u>.78</u>	<u>.60</u>	<u>.86</u>	**.42**	.36	**.18**	**.39**
10	32	.73	.56	.84	.34	**.54**	**.18**	.37
	51	.74	**.61**	.61	.21	.09	.12	.32
	83	**.80**	**.61**	.83	<u>.41</u>	.28	<u>.16</u>	<u>.38</u>

Table 4.5: Classification accuracy based on nearest neighbors predicted using cosine similarity of conceptual language vectors. Column headers from left to right: number of nearest neighbors, set of concepts (Swadesh (32), Bible (51), or All (83)), and language families (see text). **Bold** (<u>underlined</u>): best (second-best) result per column. ATLA and INDO families have the highest accuracy (.80 and .87), whereas SINO has the lowest accuracy (.18).

4.5.2 Conceptual Hamming Distance

We use Hamming distance to measure the conceptual dissimilarity of languages using binarized vectors. These vectors are structured in a similar manner as described in Section 4.4, while each dimension is either 1 if the concept associates with it or 0 otherwise. Table 4.6 shows that accuracy using Hamming distance is low for all families except INDO, which likely benefits from English as the source language (see Section 4.5.1). This bias may cause many non-INDO languages to have predominantly INDO neighbors. Detailed analysis on the distribution of the nearest neighbors in Section 4.6.1 confirms this and indicates that INDO languages indeed constitute the majority of nearest neighbors across all six families.

4.5.3 ASJP Lexical Distance

Östling and Kurfali (2023) calculate the lexical distances between 1012 languages in the PBC using ASJP word lists. We evaluate their distance matrix on the language family classification task. Table 4.6 shows near-perfect accuracy across all six families, highlighting that lexical similarity is a strong indicator of genealogical language similarity.

k	Measure	ATLA	AUST	INDO	GUIN	OTOM	SINO	All
2	CosSim	.21	.20	.53	.09	.14	.00	.13
	Hamming	.03	.08	.67	.02	.04	.00	.08
	ASJP	.94	.99	.99	.90	.95	1.00	.87
	URIEL	.98	.99	.92	.84	.97	1.00	.83
4	CosSim	.54	.41	.80	.24	.39	.15	.29
	Hamming	.13	.15	.91	.05	.08	.01	.13
	ASJP	.98	1.00	1.00	.95	.98	1.00	.88
	URIEL	.99	.99	.96	.99	.99	1.00	.87
6	CosSim	.63	.49	.85	.30	.43	.16	.33
	Hamming	.11	.13	.96	.03	.05	.00	.12
	ASJP	.98	1.00	1.00	.97	.98	1.00	.88
	URIEL	.99	1.00	.96	1.00	.99	1.00	.86
8	CosSim	.68	.53	**.87**	**.34**	.51	.18	.36
	Hamming	.13	.12	.97	.02	.03	.00	.12
	ASJP	.98	1.00	1.00	.95	.95	1.00	.88
	URIEL	.99	1.00	.96	1.00	.99	1.00	.86
10	CosSim	**.73**	**.56**	.84	**.34**	**.54**	.18	**.37**
	Hamming	.11	.10	.97	.02	.01	.00	.11
	ASJP	.99	1.00	1.00	.93	.95	1.00	.86
	URIEL	.99	1.00	.96	1.00	.99	1.00	.84

Table 4.6: Classification accuracy based on nearest neighbors predicted using various similarity and distance measures. Column headers from left to right: number of nearest neighbors, type of measure, and language families (see text). Results are calculated using the 32 Swadesh concepts. Best result per family: **bold** (CosSim), red (Hamming), teal (ASJP), blue (URIEL). Hamming distance yields high accuracy for INDO but performs poorly for other families. Measures based on ASJP and URIEL have comparably good results.

4.5.4 URIEL Typological Distance

We use typological features from the URIEL database, including syntactic, phonological, and phonetic inventory features. Languages are represented as 289-dimensional binary vectors by concatenating the typological vectors, with possibly missing values inferred using kNN. Language similarity is ranked using the Hamming distance. Table 4.6 shows that typological features yield accuracy on par with ASJP lexical distance in family classification.

4.5.5 Grambank Typological Distance

Grambank contains 195 categorical features for 2467 languages, but not all features are coded for every language. Due to the categorical nature of the features, similarity can only be compared when languages share the same subset of features. Increasing the

number of compared features inevitably reduces the number of comparable languages as a tradeoff. Therefore, we select the 50 most frequently coded features and use the Hamming distance for comparison. Evaluation is performed focusing on the five largest language families with at least 50 languages in Grambank: Austronesian (AUST), Sino-Tibetan (SINO), Atlantic-Congo (ATLA), Afro-Asiatic (AFRO), and Indo-European (INDO), with additional results for GUIN and OTOM reported for comparability with other measures. Table 4.7 shows strong performance for all seven families overall, with an accuracy of over 80% for four of the five largest families in Grambank and an average of 63% across all families. GUIN and OTOM families, which have fewer languages in Grambank compared to the PBC, have worse performance (45% and 58%).

k	sim.	AUST	SINO	ATLA	AFRO	INDO	GUIN	OTOM	All
2	Grambank	.75	.58	.78	.52	.61	.09	.42	.48
4		.89	.76	.86	**.63**	.76	.27	**.58**	.61
6		.89	.80	**.90**	**.63**	.78	**.45**	**.58**	**.63**
8		.91	**.81**	.88	**.63**	**.82**	**.45**	**.58**	**.63**
10		**.92**	.80	**.90**	.60	.78	**.45**	.42	.62

Table 4.7: Classification accuracy based on nearest neighbors for the five largest families in Grambank, along with GUIN and OTOM for better comparability with other measures. Families are listed in order of their number of languages in Grambank. **Bold** (underlined): best (second-best) result per column. High accuracy (over 80%) is observed for four of the largest families in Grambank, while AFRO archives a moderate but far above-random accuracy (63%). Accuracy is much lower for GUIN and OTOM (45% and 58%), which have significantly fewer languages in Grambank.

4.6 Analysis

4.6.1 Distribution of Nearest Neighbors

We analyze the distribution of families within the 10 nearest neighbors for each language in the six largest families of the PBC and show the results in Table 4.8. When using conceptual Hamming distance, we observe that non-INDO languages consistently have over 50% INDO languages among their nearest neighbors. This bias explains why Hamming distance achieves high accuracy for INDO languages only but performs poorly for other families. In the case of cosine similarity, INDO languages similarly have the highest proportion of same-family neighbors than other families, which aligns with their stronger classification performance.

For ASJP, the proportions of same-family neighbors are high for AUST, ATLA, INDO, and SINO languages (ranging from 89% to 99%), but lower for GUIN and OTOM languages (70% and 76%). This difference likely accounts for the slightly weaker

performance on GUIN and OTOM families. GUIN languages also frequently include non-GUIN languages from the geographically proximate Papunesia region, such as Wiru and Tabaru. This reflects the high linguistic diversity of the region, which has 29 of the 120 language families in ASJP, just under South America. A similar observation can be made for OTOM languages, whose non-OTOM neighbors are frequently South American languages which are geographically close to OTOM languages. These findings, alongside the results in Table 4.5, indicate a higher degree of diversity in the conceptualizations within these two families.

For URIEL typological features, proportions of same-family neighbors are slightly lower for INDO and GUIN (88%) families, which explains the slightly lower classification accuracy for the two families.

For Grambank (Table 4.9), we also assess the neighbors of AFRO, one of its largest families. The five majority families in Grambank exhibit predominantly same-family neighbors (ranging from 62% to 79%). The two additional families, GUIN and OTOM, show much smaller proportions of same-family neighbors. These findings are consistent with the classification accuracy in Table 4.7.

4.6.2 WALS Features

WALS (Dryer and Haspelmath, 2013) is known to have significant gaps in its feature coverage, especially if not augmented with automatic feature inference (Littell et al., 2017; Skirgård et al., 2023). These gaps may stem from differences in linguistic expertise among feature contributors and varying relevance of features across language families. Linguists specializing in specific families may concentrate on features most relevant to their area and ignore those less relevant. This variability in the feature inventories can reduce comparability across languages or families, and can potentially make genealogical classification easier.

To investigate, we calculate the coverage of WALS syntactic and phonological features for the six largest families in the PBC. Feature coverage is generally higher for INDO and SINO languages, with more variability for other families. Some features, such as "ergative-absolutive mark", have low coverage across all families (less than 10%). Other features have higher coverage for specific families. For example, "polar question word" has a higher coverage for INDO and SINO (40%) but is less represented in other families (around 20%). A small number of features are entirely absent for specific families. For example, features related to oblique positions are missing for all GUIN languages. This suggests that some families, such as INDO and SINO, are better studied and thus have more comprehensive feature coverage compared to other families.

Measure	Source lang.	% predicted neighbors					
		ATLA	AUST	INDO	GUIN	OTOM	SINO
CosSim	ATLA	41	7	13	1	4	5
	AUST	19	31	9	2	3	5
	INDO	13	3	55	0	1	2
	GUIN	15	17	2	18	3	4
	OTOM	18	5	2	1	24	3
	SINO	19	9	21	1	1	14
Hamming	ATLA	20	4	56	0	0	1
	AUST	12	14	52	0	0	1
	INDO	8	3	69	0	0	0
	GUIN	13	6	52	5	0	3
	OTOM	15	1	51	0	5	1
	SINO	10	6	58	0	0	2
ASJP	ATLA	89	2	0	1	0	1
	AUST	0	99	0	0	0	0
	INDO	0	0	99	0	0	0
	GUIN	7	7	1	70	0	1
	OTOM	4	5	1	3	76	2
	SINO	1	1	0	0	0	97
URIEL	ATLA	96	0	0	0	0	0
	AUST	0	99	0	0	0	0
	INDO	0	2	88	0	0	0
	GUIN	0	0	0	88	0	0
	OTOM	0	1	0	0	98	0
	SINO	0	1	0	0	0	96

Table 4.8: Distributions of language families within the 10 nearest neighbors for languages of the six largest language families. Source lang.: source language for which the nearest neighbors are predicted; % predicted neighbors: average percentages of languages from each family among the 10 nearest neighbors. All families have predominantly INDO neighbors when using Hamming distance. A lower percentage of same-family neighbors may correlate with a lower classification accuracy (see Section 4.6.1).

4.6.3 Grambank Features

Grambank (Skirgård et al., 2023) provides systematic feature encodings for 2467 languages, with a nearly complete feature set for most languages. In practice, however, we find many entries marked as "unknown" which are unusable for similarity comparison. Furthermore, only a few features are shared by a large number of languages. This phenomenon is mentioned by Lesage et al. (2022), who highlight a strong variability in the "description level" of Grambank features across languages.

Figure 4.3 illustrates a clear tradeoff between the size of the feature set and the number of comparable languages, as the number of comparable languages decreases sharply with an expanding feature set. For instance, while 1105 languages can be compared using 40 features, expanding the set to include 100 features reduces the number of comparable

sim.	Source lang.	% predicted neighbors						
		AUST	SINO	ATLA	AFRO	INDO	GUIN	OTOM
Grambank	AUST	79	2	2	1	0	0	1
	SINO	3	62	0	0	1	2	0
	ATLA	7	1	76	5	2	0	0
	AFRO	6	1	15	47	7	0	1
	INDO	3	3	2	3	64	0	0
	GUIN	0	17	0	0	0	24	0
	OTOM	45	0	0	0	0	0	29

Table 4.9: Distributions of language families within the 10 nearest neighbors using Grambank features. Source lang.: source language for which the nearest neighbors are predicted; % predicted neighbors: average percentages of languages from each family among the 10 nearest neighbors. The five largest families in Grambank have predominantly same-family neighbors. The proportions of same-family neighbors for the two additional families that are less represented in Grambank (GUIN and OTOM) are much lower, which is consistent with their lower accuracy in Table 4.7.

languages by more than three-quarters.

We examine the feature coverage of Grambank across its five largest families. Although Grambank features are more evenly distributed than those in WALS, discrepancies across families still remain. For example, feature number 325: *Is there a count/mass distinction in interrogative quantifiers?* is missing for half of the AFRO languages but covers over 80% of INDO languages.

4.7 Source Language

We have so far proposed the hypothesis that conceptual similarity is influenced by biases from the source language, which is English in our previous experiments, demonstrated by the high proportions of INDO neighbors for non-INDO languages (Section 4.6.1). To test this, we apply the Conceptualizer framework using two additional source languages: **Chinese**, a member of the SINO family, and **Korean**, a language isolate that constitutes the Koreanic family. Classification results for conceptual cosine similarity and conceptual Hamming distance are detailed in Tables 4.10 and 4.11. The experiments are performed using the 32 Swadesh concepts for Chinese. For Korean, we exclude the concept 'tooth', leaving 31 Swadesh concepts. This adjustment addresses the high ambiguity of the common term referring to 'tooth', 이, which occurs in about one-third of the verses. Less ambiguous alternatives, 이빨 and 치아, though more specific, are infrequent and rarely appear in the Korean Bible. We note that ambiguity in Korean texts, often caused by frequent homonyms, is common in general with the declining use of *hanja*, or Chinese characters. The proneness to ambiguity of other concepts in Korean is examined in the following part (see "Limitations of Korean").

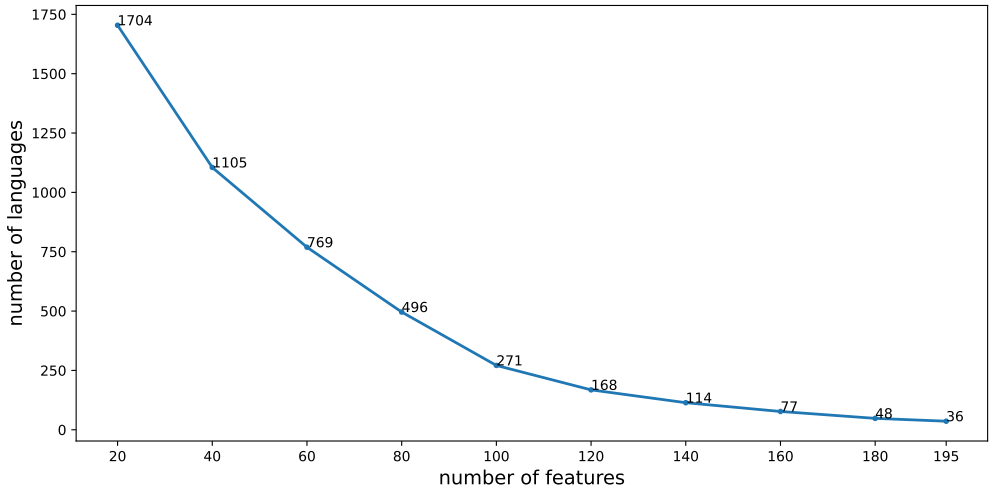


Figure 4.3: Languages must share the same set of available features to be comparable. A tradeoff between the feature set size and the number of comparable languages is shown in this graph. For example, increasing the feature set size from 40 to 100 more than quarters the set of comparable languages.

Lang.	k	ATLA	AUST	INDO	GUIN	OTOM	SINO	All
cmn	2	.13	.05	.12	.02	.01	.07	.05
	4	.43	.21	.36	.05	.03	.10	.15
	6	.50	.28	**.53**	.10	.04	.16	.20
	8	.55	.31	.52	.15	**.05**	.18	.21
	10	**.56**	**.37**	.51	**.18**	.04	**.24**	**.23**
kor	2	.16	.17	.19	.07	.09	.05	.09
	4	.33	.51	.50	.15	.14	.09	.23
	6	.46	.63	.56	**.24**	.21	**.17**	**.29**
	8	.47	.66	.58	.20	.18	.14	**.29**
	10	**.48**	**.67**	**.60**	.19	**.22**	.12	**.29**

Table 4.10: Classification accuracy based on nearest neighbors predicted using cosine similarity between conceptual representations. Conceptual representations are generated using Chinese (cmn, upper half) and Korean (kor, lower half) as the source language. **Bold** (underlined): best (second-best) result per column per source language. Using Chinese improves SINO accuracy by 6%, but results in drops of 16-49% for other families. Using Korean (agglutinative) improves AUST (many members are agglutinative) accuracy (0.30 increase over Chinese) but brings down OTOM (many members are fusional) accuracy (0.32 decrease compared to English). These results suggest a potential correlation between the source language's morphology and conceptual similarity.

Representations generated using Chinese

Despite SINO languages having a relatively low level of intra-family conceptual similarity, using Chinese as Conceptualizer's source language results in noticeable improvements

Lang.	k	ATLA	AUST	INDO	GUIN	OTOM	SINO	All
cmn	2	.07	.03	.07	.00	.00	.18	.04
	4	.32	.08	.16	.00	.03	.38	.11
	6	.49	.18	.23	**.01**	**.04**	**.50**	.17
	8	.55	**.20**	.25	.00	**.04**	.47	.18
	10	**.70**	.19	**.31**	.00	.03	.44	**.20**
kor	2	.01	.02	.06	.01	.04	**.01**	.01
	4	.11	.14	.30	.02	.04	**.01**	.08
	6	.28	.26	.46	**.04**	.04	**.01**	.15
	8	.37	.31	.57	**.04**	**.05**	**.01**	.19
	10	**.41**	**.32**	**.59**	**.04**	.04	**.01**	**.20**

Table 4.11: Classification accuracy based on nearest neighbors predicted using Hamming distance between conceptual representations. Conceptual representations are generated using Chinese (cmn, upper half) and Korean (kor, lower half) as the source language. **Bold** (underlined): best (second-best) result per column per language. Using Chinese as the source language increases SINO accuracy from near-zero (when using English) to 0.50. However, when Korean is used as the source language, SINO accuracy drops back to a similar level (0.01). In contrast, INDO achieves the highest accuracy using Korean as the source language, likely due to conceptual associations between Korean and INDO loan words.

for SINO languages. Classification accuracy is increased by up to 0.06 with cosine similarity and substantially from near zero to 0.50 with Hamming distance. This highlights the impact of the source language on languages of the same family.

Interestingly, while classification performance on SINO languages improves, accuracy for other families sees drops ranging from 0.16 to 0.34, while OTOM experiences the largest drop of 0.49. Family distributions among the predicted nearest neighbors (Table 4.12) reveal that using Chinese increases the proportion of SINO neighbors not only for SINO languages but also other families, both using conceptual cosine similarity and Hamming distance. However, ATLA languages are the most prominent neighbors in all families despite using Chinese as the source language, followed by AUST languages, which are the second most frequent neighbors for families other than INDO and SINO for conceptual cosine similarity. For Hamming distance, SINO languages are the second most prominent neighbors after ATLA, which supports the conclusion that the source language imposes a strong influence on conceptual Hamming distance.

Representations generated using Korean

As a language isolate, Korean provides a unique perspective on conceptualization. Using Korean as the source language leads to declines in cosine similarity accuracy of up to 0.32 for all families except AUST, which achieves the highest accuracy of 0.67, 0.11

Measure	Source lang.	% predicted neighbors					
		ATLA	AUST	INDO	GUIN	OTOM	SINO
Chinese (cmn)-based CosSim	ATLA	28	9	12	1	4	8
	AUST	21	20	11	2	2	8
	INDO	16	9	29	0	1	8
	GUIN	16	13	3	10	5	10
	OTOM	28	10	5	1	8	7
	SINO	22	9	12	1	3	15
Chinese (cmn)-based Hamming	ATLA	32	8	6	0	1	18
	AUST	27	15	7	0	0	16
	INDO	24	7	19	0	0	14
	GUIN	28	9	3	1	1	21
	OTOM	32	4	5	0	5	24
	SINO	27	6	9	0	1	24

Table 4.12: Distributions of language families within the 10 nearest neighbors. Conceptual representations are generated using Chinese. Source lang.: source language for which the nearest neighbors are predicted; % predicted neighbors: average percentages of languages from each family among the 10 nearest neighbors. A higher percentage of SINO neighbors is observed across all source language families compared to when using English as the source language.

higher than when using English and 0.30 higher than when using Chinese as the source language. This may suggest that the source language's morphological characteristics have an effect on conceptual similarity, as a large number of AUST languages show patterns of agglutination similar to Korean (Himmelmann, 2005; Blust, 2013). The significant declines in accuracy for OTOM languages, which exhibit fusional characteristics (Campbell, 2016; Palancar, 2016; Baerman et al., 2019), using Chinese (-0.49) and Korean (-0.32) also align with this hypothesis.

Though still much lower than using English, accuracy for the INDO family is higher when using Korean compared with Chinese (0.07 increase), likely due to the presence of loan words from INDO languages, which is much more common in Korean than in Chinese. Despite the strong influence of the Chinese language on Korean, accuracy for the SINO family is the lowest, similar to results using English. This indicates that conceptualization patterns of Korean may intrinsically differ from Chinese or other non-Chinese SINO languages. In addition, the high conceptual divergence within the SINO family may also limit the alignment.

For Hamming distance, SINO accuracy drops from 0.50 to near zero compared to using Chinese as the source language, a similar level to when using English. INDO

achieves the highest accuracy of 0.59, likely benefiting from binarization amplifying conceptual associations to loanwords. Family distributions of nearest neighbors in Table 4.13 shows more variability for the six families, with the three larger families, especially INDO, being dominant neighbors.

Measure	Source lang.	% predicted neighbors					
		ATLA	AUST	INDO	GUIN	OTOM	SINO
Korean (kor)-based CosSim	ATLA	30	18	13	0	2	4
	AUST	13	35	11	1	3	4
	INDO	13	13	37	0	1	1
	GUIN	9	24	4	10	6	4
	OTOM	10	26	4	1	14	4
	SINO	15	13	19	1	1	10
Korean (kor)-based Hamming	ATLA	22	8	12	0	0	1
	AUST	13	18	14	0	0	1
	INDO	8	10	26	0	0	1
	GUIN	12	9	11	2	0	1
	OTOM	13	12	10	0	4	2
	SINO	11	8	15	0	0	3

Table 4.13: Distributions of language families within the 10 nearest neighbors. Conceptual representations are generated using Korean. Source lang.: source language for which the nearest neighbors are predicted; % predicted neighbors: average percentages of languages from each family among the 10 nearest neighbors. The family distribution exhibits greater variability compared to results using English and Chinese as source languages but skews toward the three largest families, a pattern consistent with using Chinese as the source language.

Visualizations of conceptual representations

Figure 4.4 shows t-SNE clusters of languages based on family and geographic region using English, Chinese, and Korean as source languages. The graphs illustrate that family clustering is weaker when using Chinese or Korean as the source language compared to English. For both Chinese and Korean, apart from an INDO cluster, languages of most families or regions are more dispersed. Some trends can be observed across three source languages. For example, INDO and Eurasian languages form denser clusters than other families or regions, and GUIN languages are more spread out, indicating a high conceptual variability within the family despite the geographic proximity of its members. Based on the visualizations, we suggest that the earlier observation of ATLA and AUST

dominance in predicted neighbors may result from two factors: 1) their large family sizes, and 2) linguistic factors in conceptual representations created using Chinese and Korean.

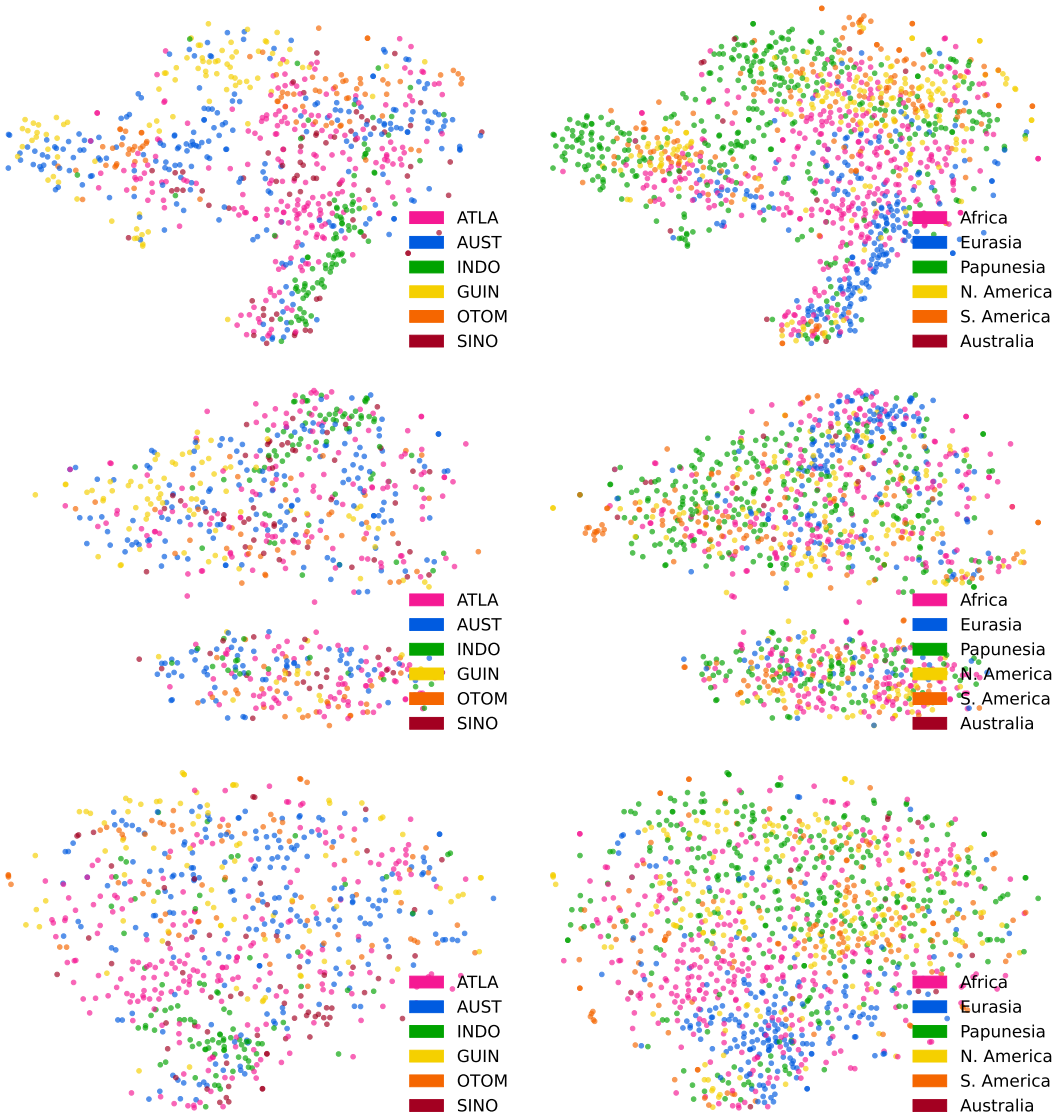


Figure 4.4: t-SNE visualizations of languages using their conceptual representations based on 32 (31 when Korean is the source language) Swadesh concepts, with **eng** (top two), **cmn** (middle two), or **kor** (bottom two) as the source language. The colors indicate different language families or geographic areas.

Influence of family size

The six families studied vary greatly in size, which potentially impacts the neighbor distributions. ATLA and AUST are the largest of the six families, with over 200 languages

each, while SINO, the smallest family, has 68 languages. It is thus more likely for ATLA and AUST languages to dominate the predicted neighbors than the much less frequent SINO languages. The influence of family size is supported by the higher accuracy for the three largest families (ATLA, AUST, INDO) using Korean as the source language (Tables 4.10 and 4.11).

Limitations of Chinese

The use of Chinese as the source language introduces unique challenges, including ambiguous tokenization due to the absence of explicit word boundaries like spaces. This can lead to noisy BP alignments, such as sequences that are excessively long or meaningless. For example, one possible translation of 'bird' is "飞禽". However, associated n-grams retrieved by BP include not only "飞禽" but also unrelated fragments like "飞" ("to fly") and "禽" ("fowl"). Frequent redundancies with the possessive marker "的" are observed, such as "鸽子" (pigeon) and "鸽子的" ("pigeon's"), which are among the most common associations of many concepts. Additionally, frequent free translations in the Chinese Bible, for example by using related terms instead of exact translations, further increase the noise in BP.

Limitations of Korean

Contrary to Chinese, the explicit marking of word boundaries in Korean reduces the potential noise from ambiguous tokenizations. However, the agglutinative morphology of Korean leads to redundant associations due to its rich inventory of suffixes, or attachable particles to a noun to serve different functions. For instance, variations of the string "sheep" are shown to be associated with the concept: 양 ("sheep"), 양이 ("sheep", subject), 양을 ("sheep", object), 양들 ("sheep", plural), 양들은 ("sheep", plural topic), and 양과 ("with the sheep"). Ambiguity due to homonymy is also prevalent, especially for common single-syllable words with native Korean origins (as opposed to having Chinese roots). For example, the word 새 ("bird") also means "new" or "between", while 이 ("tooth") more commonly refers to "this" or "two". Historically, words like these were written using different *hanjas*, or Chinese characters, which usually served to mitigate ambiguity for polysemous words but are no longer commonly used in modern-day Korean. Ambiguity misleads BP to retrieve noisy n-grams, as illustrated by the example concept 'mouth' (입 in Korean). 입, however, also occurs in verbs such as 입다, meaning "to wear", which leads to associations related to clothes such as 옷 ("clothes"), 벗(다) ("to take off"), and 망토 ("cloak").

Diversity of conceptual representations

The redundancy in associations across different concepts is an indication that conceptual representations generated using Chinese or Korean are more diverse, or contain more non-zero elements, than English. To investigate this, we analyze conceptual representations generated with the three source languages and rank the language families based on two criteria: 1) **most frequent neighbors** - the number of languages most frequently appearing within the 10 nearest neighbors, and 2) **most diverse families** - the average diversity of representations of each family's members. While the percentage of the most frequent neighbors remains the same for English and Chinese (52%), and is only slightly higher for Korean (56%), the percentage of the most diverse families is considerably higher for Chinese and Korean. Specifically, the three most diverse families account for 36% of all languages when English is the source language, whereas this percentage rises to 50% and 53% using Chinese and Korean, respectively. This shows that conceptual representations tend to be more diverse on average when using Chinese or Korean as the Conceptualizer's source language, making languages conceptually more similar.

4.8 Conclusion

To the best of our knowledge, this work presents the first empirical evaluation of diverse types of language representations with respect to their predictive performance for genealogical language similarity. Our evaluation includes recently proposed works on conceptual language similarity (Liu et al., 2023b) and the grammatical feature database, Grambank (Skirgård et al., 2023), making its first application to language similarity prediction and an analysis of its limitations.

Our previous findings have demonstrated interesting complementarities of conceptual language similarity to traditional measures, such as typological similarity. For example, languages typically not considered similar within traditional genealogical or typological frameworks, such as Tagalog and Spanish, exhibit noticeable similarities on a conceptual level. As indicated by evaluation results in Table 4.6, conceptual similarity is less effective in predicting genealogical relationships, reflected by performance on language family classification, than lexical or typological similarity measures. This is not surprising, as lexical and typological features often result from or are influenced by genealogical proximity, particularly within the same language family. Many typological features, such as those coded in WALS, are related to word order or lexicon, which strongly correlate with genealogical or geographic proximity. Furthermore, as noted in Section 4.5, the primary objective of the language family classification task is to examine the level of correlation between conceptual and genealogical similarities. The high classification accuracy observed for some language families, as shown in Table 4.5, highlights the utility of conceptual similarity. Consequently, we note that current evaluation tasks do

not fully align with the conceptual view of language similarity. Therefore, if the primary focus is classification accuracy, typological or lexical features provide strong signals for genealogical relatedness. However, for high-level comparisons that extend beyond the language family boundaries, conceptual similarity nevertheless offers unique and valuable insights.

Lastly, we analyze the effect of the source language in generating conceptual representation, specifically comparing Chinese and Korean to English in the original Conceptualizer setup. Our findings support the hypothesis that conceptual similarity is inherently biased toward the source language and suggest a consistent influence of the sizes of the language families on conceptualization. Among the three source languages studied, English produces the most stable results. We believe that this stability may have resulted from the simpler morphology of English, which mitigates some of the limitations discussed in Section 4.7.

Chapter 5

Model Stitching for Cross-Lingual Zero-Shot Transfer

This chapter corresponds to the following work:

> **Haotian Ye***, Yihong Liu*, Chunlan Ma*, Hinrich Schütze (2024). MoSECroT: Model Stitching with Static Word Embeddings for Crosslingual Zero-shot Transfer. Proceedings of the Fifth Workshop on Insights from Negative Results in NLP, 2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics. *equal contribution.

Declaration of Co-Authorship. I conducted all experiments and analyzed the results. The draft was written by me in collaboration with the other co-authors.

5.1 Introduction

Transformer-based pre-trained language models (PLMs) (Devlin et al., 2019; Conneau et al., 2020) have demonstrated exceptional performance across a wide range of NLP tasks (Artetxe et al., 2020a; Imani et al., 2023), representing an important milestone in the field. However, these models require substantial computing resources for pre-training and are generally limited to no more than a hundred languages for which sufficient pre-training data is available, leaving the vast majority of the world's low-resource languages behind. In contrast, static word embeddings are computationally efficient to train and require significantly less data for their training, making them more tangible for low-resource languages.

In this chapter, we present a novel framework, the first that leverages relative representations to construct a shared embedding space for a source language PLM and static word embeddings of a target language. Specifically, our approach leverages (1) a PLM in a high-resource source language, (2) static word embeddings in a target language, which are more readily available and inexpensive to train for many low-resource languages, and (3) a technique known as *model stitching* to enable zero-shot transfer to the target language without requiring any pre-training.

The contributions mentioned in this chapter are threefold: (i) we introduce **MoSECroT**, (**Mo**del **S**titching with Static Word **E**mbeddings for **Cro**sslingual Zero-shot **T**ransfer), a novel and challenging task designed for cross-lingual zero-shot transfer, especially to low-resource languages where static word embeddings are available; (ii) we present a method that employs relative representations to align the source (English in our case) and target language embeddings in a common space, allowing zero-shot transfer for the target languages; (iii) we evaluate the proposed framework extensively on two text classification datasets.

Despite the theoretical support for the framework, we nevertheless show that while it exhibits competitive results with weaker baselines, it struggles to match the performance of strong baselines. We discuss possible implications of the negative results, identify potential limitations of our framework, and provide insights for future directions that could address the challenges encountered by our framework.

5.2 Related Work

Aligned cross-lingual word embeddings facilitate transfer learning by creating a shared representation space for source and target languages. These embeddings are typically obtained through either joint training (Hermann and Blunsom, 2014; Vulic and Moens, 2016) or post-alignment methods (Lample et al., 2018a; Artetxe et al., 2018). In this work, we adopt a transformation method similar to post-alignment approaches to align two embedding spaces, where the source embeddings are derived from a PLM and the

target embeddings are pre-trained static word embeddings.

Building on the consensus that neural networks, regardless of their architecture or domain, tend to learn similar internal representations (Kornblith et al., 2019; Vulić et al., 2020), Moschella et al. (2023) propose a method for aligning latent spaces using a set of samples which they name *parallel anchors*. Their approach involves transforming the original, absolute embedding space into a space defined by relative coordinates of the parallel anchors. The transformed embeddings in the new relative space are referred to as relative representations and effectively encode relationships of these embeddings relative to the anchors.

Model stitching is a technique originally proposed to integrate components of different neural networks. Trainable stitching layers were first proposed by Lenc and Vedaldi (2015), with subsequent studies demonstrating their effectiveness across various tasks (Bianchi et al., 2020; Bansal et al., 2021). In this work, we extend these ideas by employing model stitching to enable cross-lingual transfer without requiring re-training of the components.

5.3 Task Setting

The task setting of MoSECroT is as follows: given a PLM of a high-resource language (regarded as the source language in our framework) and static word embeddings of a low-resource language (the target language), our objective is to enable zero-shot transfer by directly integrating the target language embeddings with the source language PLM through embedding layer stitching. This is achieved by first aligning the source and target embedding spaces and subsequently swapping the embedding matrix of the PLM with the target embeddings. We propose a novel method that utilizes relative representations for embedding space alignment. The details of our methodology are presented in Section 5.4.

5.4 Methodology

Parallel anchor selection

To establish a set of parallel anchors, we first extract bilingual lexica between the source and target languages. For most high-resource languages, high-quality bilingual lexica are available from MUSE[1]. For low-resource languages, we obtain translations of source language vocabulary by crawling PanLex[2] and Google Translate[3]. We use

[1] https://github.com/facebookresearch/MUSE
[2] https://panlex.org
[3] https://translate.google.com

Google Translate for Tatar (tt), one of the low-resource target languages, because PanLex uses a mix of Latin and Cyrillic scripts for Tatar, while Google Translate consistently uses Cyrillic, the same script used in the datasets and pre-trained embeddings. For the remaining low-resource target languages, PanLex translations are used.

From the bilingual lexica, we derive parallel anchors A by retaining only those lexical pairs that exist in the embedding matrices of both the source and target languages. The source language is always English in our setting, and source language embeddings are derived from the token embeddings of English BERT (Devlin et al., 2019). Target language embeddings are pre-trained static word embeddings from fastText (Bojanowski et al., 2017).

Relative representations

Following Moschella et al. (2023), we construct relative representations (RRs) for each token in the shared embedding space based on their similarities with the anchor tokens in the respective languages. Specifically, for each token, we calculate the cosine similarity between its embedding and the embeddings of all anchor tokens. This process is performed independently in the source and target language embedding spaces. For example, the similarity of a source language token x_i with an anchor a_j is defined as:

$$r^s_{(i,j)} = \text{cos-sim}(\boldsymbol{E}^s_{\{x_i\}}, \boldsymbol{E}^s_{\{a_j\}})$$

where $\boldsymbol{E}^s_{\{x_i\}}$ and $\boldsymbol{E}^s_{\{a_j\}}$ are the embeddings of x_i and a_j in the embedding matrix of the source language PLM, $\boldsymbol{E}^s$. The relative representation of x_i is then defined as:

$$\boldsymbol{R}^s_{\{x_i\}} = [r^s_{(i,1)}, r^s_{(i,2)}, r^s_{(i,3)}, \cdots, r^s_{(i,|A|)}]$$

The same procedure is applied to tokens in the target language. Importantly, the relative representations are sensitive to the order of the anchors, which must remain consistent for all tokens and languages. This computation results in a matrix $\boldsymbol{R}^s \in \mathbb{R}^{|V^s| \times |A|}$ of source language embeddings and a matrix $\boldsymbol{R}^t \in \mathbb{R}^{|V^t| \times |A|}$ of target language embeddings, where $|V^s|$ and $|V^t|$ are the vocabulary sizes of the source and target languages, respectively, and $|A|$ is the number of parallel anchors.

Embedding mapping

The relative representations of both source and target languages obtained in the previous step are vectors in $\mathbb{R}^{|A|}$, which does not match the hidden dimension D of the Transformer body of the source PLM. To address this, we map the relative representations back to $\mathbb{R}^D$, the dimensionality of $\boldsymbol{E}^s$. For a token x_i in the source language (respectively, token y_i in the target language), the transformed embedding is computed as:

$$\boldsymbol{F}^s_{\{x_i\}} = \frac{\sum_{n \in \mathbb{N}(x_i)} (\boldsymbol{R}^s_{\{x_i\},n} / \tau \cdot \boldsymbol{E}^s_{\{n\}})}{\sum_{n \in \mathbb{N}(x_i)} \boldsymbol{R}^s_{\{x_i\},n} / \tau}$$

$$\boldsymbol{F}^t_{\{y_i\}} = \frac{\sum_{n\in\mathbb{N}(y_i)}(\boldsymbol{R}^t_{\{y_i\},n}/\tau \cdot \boldsymbol{E}^s_{\{n\}})}{\sum_{n\in\mathbb{N}(y_i)} \boldsymbol{R}^t_{\{y_i\},n}/\tau}$$

where $\mathbb{N}(x_i)$ (respectively, $\mathbb{N}(y_i)$) represents the set of top-k closest anchors in terms of cosine similarity in $\boldsymbol{R}^s_{x_i}$ (respectively, $\boldsymbol{R}^t_{y_i}$), $\boldsymbol{R}^s_{\{x_i\},n}$ (respectively, $\boldsymbol{R}^s_{\{y_i\},n}$) is the cosine similarity between $\boldsymbol{E}^s_{\{x_i\}}$ (respectively, $\boldsymbol{E}^t_{\{y_i\}}$) and $\boldsymbol{E}^s_{\{n\}}$ (respectively, $\boldsymbol{E}^t_{\{n\}}$), and τ is the temperature. This transformation ensures that embeddings of both $\boldsymbol{F}^s_{\{x_i\}}$ and $\boldsymbol{F}^t_{\{y_i\}}$ are in $\mathbb{R}^D$. In essence, any token, whether from the source or target language, is represented as a weighted sum of a number (determined by k) of source language anchor embeddings.

Zero-shot model stitching

The transformed target language embeddings, $\boldsymbol{F}^t$, now align with the hidden dimension D of the source PLM's Transformer body. Similarly, the embedding matrix of the source PLM is also manipulated while keeping its original dimensions, resulting in $\boldsymbol{F}^s \in \mathbb{R}^{|V^s|\times D}$. To enable zero-shot transfer, we can first fine-tune the source language PLM, consisting of $\boldsymbol{F}^s$ and the Transformer body, on source language training data of a downstream task. Subsequently, we can assemble a target language model by replacing $\boldsymbol{F}^s$ with $\boldsymbol{F}^t$, enabling it to perform the task in the target language without any additional training.

5.5 Experiments

5.5.1 Setup and data

We use the cased version of the English BERT model (`bert-base-cased`) as the source language PLM and consider eight target languages. Among the target languages, three are high-resource: German (**de**), Spanish (**es**), and Chinese (**zh**), while the remaining five are low-resource: Faroese (**fo**), Maltese (**mt**), Eastern Low German (**nds**), Sakha (**sah**), and Tatar (**tt**). Pre-trained static word embeddings for all target languages are available from fastText[4], except for Eastern Low German, for which we use another set of fastText embeddings available on Huggingface[5].

Using the methodology described in Section 5.4, we extract pairwise parallel anchors between English and each target language. The size of the anchor set varies depending on the overlap between the English lexicon and each target language's lexicon. The anchor set sizes for each target language are shown in Table 5.1.

We evaluate the proposed method on two text classification datasets, as described below.

[4] `https://fasttext.cc/docs/en/pretrained-vectors.html`
[5] `https://huggingface.co/facebook/fasttext-nds-vectors`

en-de	en-es	en-zh	en-fo	en-mt	en-nds	en-sah	en-tt
11836	11395	7662	1577	2600	1309	3242	9275

Table 5.1: Sizes of the parallel anchor set for each target language.

Multilingual Amazon Reviews Corpus

Introduced by Keung et al. (2020), this dataset contains product reviews in six languages. The original dataset features five labels corresponding to star ratings, which we aggregate into three classes: positive, neutral, and negative. We fine-tune the source language PLM using the English training data and select the best model checkpoint based on performance on the English development set. The three high-resource languages (de, es, zh) are evaluated on this dataset.

Taxi1500

Taxi1500 (Ma et al., 2023), presented in Chapter 3, is a classification dataset comprising six classes for more than 1500 languages, including all of the target languages considered in this work. We follow the original training procedure and hyperparameters, with the exception of the learning rate, which is adjusted to $1e^{-5}$ from $2e^{-5}$, which works better in our experiments.

5.5.2 RR weighting

In addition to the standard weighting scheme described in Section 5.4, we propose two alternative settings for computing relative representation weights during the mapping step. The first applies a softmax function over the relative representation weights and the second uses sparsemax (Martins and Astudillo, 2016), which produces sparse weight distributions contrary to softmax, concentrating similarities on fewer anchors.

To determine the optimal number of top-k closest anchors, we conduct preliminary experiments for $k \in \{1, 10, 50, 100\}$, in addition to a full anchor set (6731). The optimal value for k is determined based on zero-shot performance on the German and Chinese subsets of the Amazon Reviews Corpus. Results for different values of k in Table 5.2 indicate that using the top 50 anchors yields the best performance.

5.5.3 Baselines

To evaluate the effectiveness of our proposed method, we compare it against three baselines.

k	de	zh
1	0.44	0.41
10	0.51	0.38
50	0.50	0.40
100	0.51	0.38
6731	0.44	0.21

Table 5.2: The number of closest parallel anchors (k) and the corresponding zero-shot performance on German (de) and Chinese (zh) subsets of the Amazon Reviews Corpus.

Logistic regression (LR)

We implement a simple logistic regression classifier trained on the target language data, using the average of static word embeddings of the words as the input sentence embedding. While this approach does not require computationally expensive training, it assumes the availability of sufficient labeled training data in the target language, a condition that is rarely met for most low-resource languages in real-world scenarios.

mBERT

We fine-tune multilingual BERT (mBERT) (Devlin et al., 2019), a PLM pre-trained on over 100 languages, using English training data. We then perform zero-shot predictions directly on the target language test data. While mBERT is expected to perform competitively on languages in its pre-training data, performance on low-resource or unseen languages will likely vary.

Least squares projection (LS)

Inspired by embedding alignment methods like VecMap (Artetxe et al., 2018), we project target language embeddings into the same vector space as the English PLM embeddings. Specifically, a transformation matrix $\boldsymbol{W} \in \mathbb{R}^{D^t \times D}$ is learned by minimizing the Frobenius norm $||\boldsymbol{A}^t\boldsymbol{W} - \boldsymbol{A}^s||_F^2$, where $\boldsymbol{A}^t \in \mathbb{R}^{|A| \times D^t}$ represents the anchor embeddings in the target language, and $\boldsymbol{A}^s \in \mathbb{R}^{|A| \times D}$ represents the anchor embeddings in the English PLM. We then apply the learned transformation matrix $\boldsymbol{W}$ to project all target language embeddings into the PLM's embedding space and use them to replace the PLM's original embedding layer.

5.5.4 Computing Resources

The proposed method is computationally inexpensive. For the Multilingual Amazon Reviews Corpus, training can be completed within three hours using eight NVIDIA

GeForce GTX 1080 Ti GPUs. For Taxi1500, training is significantly faster, taking about 30 minutes using a single NVIDIA GeForce GTX 1080 Ti GPU.

5.6 Results

We present the evaluation results using relative representations under all three weighting schemes (Section 5.5.2) and compare them with the baselines in Tables 5.3 and 5.4. Macro F_1 scores are used due to class imbalance in both datasets.

The results indicate that the naive least squares (LS) baseline is consistently outperformed by multiple RR settings across most languages on both datasets. An exception is observed for Eastern Low German (nds) in Table 5.4, where both LS and RRs perform poorly. This finding highlights that RRs can leverage the semantic similarities encoded in different types of embeddings more effectively than LS.

Zero-shot results with mBERT, as expected, demonstrate strong performance for high-resource languages in both datasets. However, mBERT underperforms logistic regression (LR) by a significant margin for low-resource languages in Taxi1500. This outcome is likely affected by two factors. First, as noted in prior work (Wu and Dredze, 2020), representations in mBERT are not well-aligned across low-resource languages, likely due to data sparsity during pre-training. This discrepancy is reflected in mBERT's strong performance on high-resource languages but suboptimal results for low-resource languages. Second, Taxi1500 is a relatively easy classification task, where a model with good cross-lingual word-level alignment is expected to perform well. This argument is consistent with findings by Liu et al. (2023a), which show that well-aligned word embeddings outperform multilingual PLMs in zero-shot cross-lingual transfer for a wide range of languages in Taxi1500.

For LR, results are competitive for most low-resource languages, where it outperforms other baselines and RR settings. This demonstrates the robustness of LR for simple classification tasks like ours given sufficient target language labeled data.

Although none of the RR settings outperforms mBERT on high-resource languages, where mBERT demonstrates strong cross-lingual capabilities, RRs consistently outperform mBERT for all five low-resource languages not covered by mBERT's pre-training data. The performance margin ranges from +0.12 for Sakha (sah) to +0.01 for Eastern Low German (nds). These results suggest that RRs represent a promising alternative for low-resource languages not supported by mPLMs.

5.7 Analysis

In this section, we discuss possible reasons for the suboptimal performance of our model stitching approach on the MoSECroT task.

	de	es	zh
LR	0.52	0.51	0.50
mBERT	**0.61**	**0.65**	**0.51**
LS	0.46	0.46	0.30
RRs standard top-50	0.53	0.51	0.38
RRs softmax top-50	0.50	0.53	0.38
RRs sparsemax top-50	0.56	0.57	0.24

Table 5.3: Evaluation results on the Amazon Reviews Corpus. Reported scores are macro F_1s on the test sets of three high-resource target languages. de: German, es: Spanish, zh: Chinese. **Bold**: highest score per column.

	de	es	zh	mt	sah	fo	nds	tt
LR	**0.30**	0.32	0.56	**0.38**	**0.48**	**0.47**	**0.18**	**0.43**
mBERT	0.24	**0.60**	**0.62**	0.08	0.07	0.18	0.12	0.18
LS	0.14	0.26	0.24	0.08	0.12	0.06	0.08	0.07
RRs standard top50	0.20	0.44	0.28	0.14	0.16	0.16	0.06	0.14
RRs softmax top50	0.20	0.48	0.28	0.15	0.19	0.16	0.06	0.17
RRs sparsemax top50	0.24	0.37	0.13	0.15	0.18	0.20	0.13	0.21

Table 5.4: Evaluation results on the Taxi1500 dataset. Reported scores are macro F_1s on the test sets of eight target languages. de: German, es: Spanish, zh: Chinese, mt: Maltese, sah: Sakha, fo: Faroese, nds: Eastern Low German, tt: Tatar. Results are averaged over five runs using different random seeds. **Bold**: highest score per column.

Anchor selection

The quality of the parallel anchors largely relies on that of the bilingual lexica from which they are derived. Depending on the language, the lexica may contain polysemous words to varying degrees, which may influence the alignment quality. In addition, normalization of lexicon entries may cause ambiguities. For example, all words in MUSE lexica are converted to lowercase, resulting in ambiguities like the German word `sie`, which has three corresponding entries in the German-English lexicon: `you`, `she`, and `they`. Our anchor selection approach has two limitations: (1) for target language words with multiple listed translations, we consider only the last entry for these words in the lexicon, which may not be the most accurate; (2) all target language words whose translations exist in the source language vocabulary are treated as anchors, increasing the likelihood of noisy translation pairs.

To mitigate the influence of potentially noisy anchors, we experiment with reduced anchor set sizes of 3000 and 500 (from the original 6731 anchors used in preliminary

experiments) through random sampling. This approach is motivated by Moschella et al. (2023), who observe that uniform selection from an anchor set is both straightforward and effective. Additionally, stop words, which tend to have less stable translations, are removed from the anchor set. However, neither modification has led to improved results over the full anchor set (see Table 5.5, results obtained on the German and Chinese subsets of the Amazon Reviews Corpus). One possible explanation is that translation quality varies across anchors, making it difficult to predict the quality of sampled anchors.

$\lvert\boldsymbol{A}\rvert$	de	zh
500	0.39	0.19
3000	0.19	0.19
6731	0.44	0.21

Table 5.5: The total number of parallel anchors and the corresponding zero-shot performance on the German (de) and Chinese (zh) subsets of the Amazon Reviews Corpus.

Translation quality

We observe that a large portion of translations retrieved from PanLex is of low quality, partly because PanLex often relies on intermediate languages when direct translations are unavailable for a given language pair. To address this, we filter translations using empirically set thresholds on the translation quality scores, which are obtained through the PanLex API for every translation. However, we notice that high translation quality scores do not guarantee accurate translations, and conversely, many translations with low translation quality scores are, in fact, good translations upon manual examination. We thus believe that the limited availability of high-quality parallel lexica is a possible contributing factor preventing RRs from reaching their full potential, particularly for low-resource languages.

Reinitialized embedding space

Our method involves swapping the original PLM embeddings with the transformed English RRs before fine-tuning on English data. This reinitialization can cause the RR's embedding space to diverge substantially from the original PLM's embedding space. Consequently, it is unclear whether the rest of the PLM parameters can be adapted to the new embeddings during fine-tuning, especially on smaller datasets like Taxi1500. The alteration of the embedding space through reinitialization with RRs is likely another factor contributing to the failure to create a good representation space and thus suboptimal performance.

5.8 Conclusion

In this chapter, we introduce MoSECroT, a novel and challenging task designed to evaluate zero-shot transfer capabilities, particularly for low-resource languages where static word embeddings are available but other resources are scarce. In addition, we propose, for the first time, a method that leverages relative representations (RRs) for embedding space alignment, enabling effective zero-shot transfer. Our approach involves fine-tuning a monolingual English PLM using only English data, swapping its embeddings with target language embeddings aligned using RRs, and evaluating the model's zero-shot performance on the target language. This approach avoids the need for additional pre-training or fine-tuning for the target language. Through extensive experimentation on eight target languages and adjustments to the RR configurations, we demonstrate that the proposed method shows promising results compared to mBERT on unseen languages, although the observed improvements remain modest. We discuss several possible factors contributing to the suboptimal results and identify possibilities for future research.

Chapter 6

Language-Script Aware Multilingual Pretraining

This chapter corresponds to the following work:

> Yihong Liu*, **Haotian Ye***, Chunlan Ma, Mingyang Wang, Hinrich Schütze (2024). LangSAMP: Language-Script Aware Multilingual Pretraining. *equal contribution.

Declaration of Co-Authorship. Yihong Liu conceived the idea of improving the language neutrality of the transformer outputs by delegating the encoding of language-specific information to the proposed language/script embeddings. He also conducted the pre-training of the augmented model and evaluated the language and script embeddings on downstream tasks, providing the corresponding analyses. I performed experiments and evaluations related to selecting the best donor languages for transfer learning based on the trained language and script embeddings, and provided the analysis for this part. Chunlan Ma contributed by analyzing the visualization of the distribution of language and script embeddings. The draft was reviewed by all co-authors.

6.1 Introduction

Encoder-only multilingual PLMs (mPLMs) are widely regarded as universal text encoders (Cer et al., 2018; Huang et al., 2019; Yang et al., 2020), whose sentence- or token-level representations are applied to diverse downstream tasks in multilingual settings (Wei et al., 2021). One of their most valuable applications lies in cross-lingual transfer (Zoph et al., 2016; Wu and Dredze, 2019; Artetxe et al., 2020a), where the model fine-tuned on a single source language can be directly applied to other languages without further training. This is particularly useful for low-resource languages with scarce annotated training data (Artetxe et al., 2020b).

The success of cross-lingual transfer relies heavily on the transferability of the underlying representations of mPLMs. However, prior studies have shown that representations of recent mPLMs encode substantial language- and script-specific information (Datta et al., 2020; Chang et al., 2022; Wen-Yi and Mimno, 2023). This generally has a negative impact on *language neutrality*, i.e., the ability to produce representations for different languages that share a unified subspace, which is crucial for effective cross-lingual transfer (Libovický et al., 2020; Chang et al., 2022; Hua et al., 2024). While post-alignment approaches have been explored to improve the language neutrality of these representations (Cao et al., 2020; Pan et al., 2021; Liu et al., 2024b; Xhelili et al., 2024), limited efforts have addressed the issue from an architectural perspective or during pre-training.

Early mPLMs, such as XLM (Conneau and Lample, 2019), introduced language embeddings, which are learnable vectors assigned to individual languages and are added to the token embeddings before they are fed into Transformer layers to alleviate the burden of encoding language-specific information within the token embeddings. This configuration improves the language neutrality of token embeddings and aids tasks like machine translation by guiding generation toward the correct target language (Conneau and Lample, 2019; Song et al., 2019; Liu et al., 2022). However, more recent mPLMs, including XLM-R (Conneau et al., 2020) and mBERT (Devlin et al., 2019), have discarded language embeddings, primarily for two reasons: (1) mPLMs are expected to have a single, unified set of parameters for all languages and (2) function seamlessly as universal text encoders without requiring language IDs. However, this removal shifts the burden of encoding all language-specific information to the token representations, reducing their language neutrality and potentially impairing cross-lingual transfer.

To address this limitation, we propose **Lang**uage-**S**cript **A**ware **M**ultilingual **P**re-training (LANGSAMP), a method to integrate language and script embeddings to enhance representation learning during pre-training, while maintaining a simple architecture. Unlike previous methods that add these embeddings to token embeddings before feeding them into Transformer blocks, LANGSAMP incorporates them into the final contextual token embeddings output by the Transformer blocks. These enhanced representations are then passed to the language modeling head, as shown in Figure 6.1. This design ensures that the Transformer backbone, comprised of token embeddings and Transformer blocks,

does not require language or script IDs as input, similar to most recent mPLMs.

During pre-training, language and script IDs are used to obtain language and script embeddings that offload the burden of encoding language- and script-specific information from token embeddings, which improves the capability to decode specific tokens in masked language modeling. After pre-training, the backbone can operate seamlessly as a universal text encoder and be fine-tuned for downstream tasks without requiring language or script IDs as input.

We validate our approach by continually pre-training XLM-R using LANGSAMP on Glot500-c (Imani et al., 2023), a multilingual corpus encompassing over 500 languages. The resulting model is evaluated across diverse downstream tasks, including sentence retrieval, text classification, and sequence labeling. We show that our method consistently outperforms the baseline model, highlighting its effectiveness in improving cross-lingual transfer. In addition, our ablation study shows the benefits of incorporating both language and script embeddings with improvements to downstream performance. We show that better language neutrality can be achieved using LANGSAMP, reflected by increased pairwise cosine similarity across languages overall. Notably, we observe that the learned language and script embeddings capture typological features, making them a useful resource for selecting optimal source languages in cross-lingual transfer.

The main contributions of this work are as follows: (i) We propose LANGSAMP, an effective multilingual pre-training method that improves the language neutrality of mPLM representations. (ii) We conduct extensive experiments across diverse downstream tasks, demonstrating consistent performance improvements in cross-lingual transfer. (iii) We show in a case study that language embeddings, as a byproduct of LANGSAMP, effectively assists in selecting optimal source languages in cross-lingual transfer.

6.2 Related Work

6.2.1 Multilingual Pre-trained Language Models

Multilingual pre-trained language models (mPLMs) are pre-trained on data in multiple languages using one or more self-supervised learning objectives, such as masked language modeling (MLM) (Devlin et al., 2019) or causal language modeling (CLM) (Radford et al., 2019). These models can generally be categorized into three groups based on their architectures: encoder-only (Devlin et al., 2019; Conneau et al., 2020; Liang et al., 2023), encoder-decoder (Liu et al., 2020; Fan et al., 2021; Xue et al., 2021), and decoder-only models (Lin et al., 2022; Scao et al., 2022; Shliazhko et al., 2024).

Decoder-only models, particularly those with considerable amounts of parameters and pre-trained on extensive data, are also referred to as large language models (LLMs) (OpenAI, 2023; Touvron et al., 2023b; Üstün et al., 2024). LLMs often excel at natural language generation tasks, particularly for high- and medium-resource languages. In

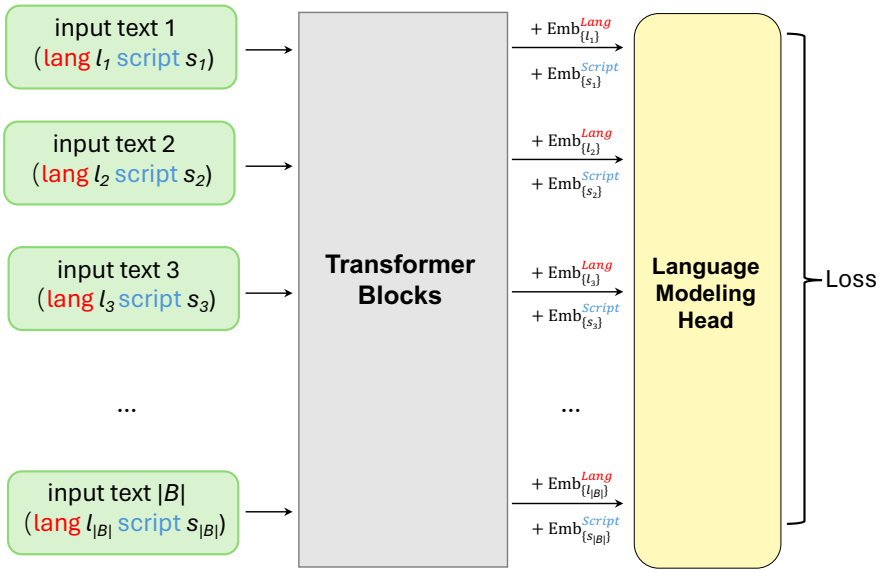


Figure 6.1: An illustration of LANGSAMP for a single input batch, where text can be in different languages and scripts. Language and script embeddings are added to the Transformer output before the enhanced token embeddings are fed into the language modeling head. This configuration enhances the language neutrality of the representations by leveraging auxiliary embeddings to offload the burden of encoding language- and script-specific information from token embeddings.

parallel, recent efforts have focused on the horizontal scaling of encoder-only models, extending their coverage to a broader range of languages, including low-resource ones (Ogueji et al., 2021; Alabi et al., 2022; Imani et al., 2023; Liu et al., 2024a). Such highly multilingual encoder-only models are particularly effective in solving diverse tasks under zero-shot cross-lingual settings.

6.2.2 Language Embeddings

Language embeddings are vectors that explicitly or implicitly represent linguistic characteristics of languages. Early approaches have constructed these embeddings using predefined linguistic features, where each dimension of the vector encodes a specific linguistic feature (Östling, 2015; Ammar et al., 2016; Littell et al., 2017). However, these features are typically defined manually and may not be available for less-studied languages (Yu et al., 2021).

To address this limitation, later works explore learning language embeddings directly from either parallel corpora (Malaviya et al., 2017; Östling and Tiedemann, 2017; Bjerva and Augenstein, 2018; Tan et al., 2019; Liu et al., 2023b; Chen et al., 2023) or monolingual corpora (Conneau and Lample, 2019; Yu et al., 2021). This is typically done by assigning each language a unique ID and initializing fixed-length, learnable vectors,

which are integrated into the input from that language. Language embeddings learned in this way capture linguistic features that enhance performance on cross-lingual tasks, for example, by guiding language-specific generation in machine translation, as illustrated in the case of XLM (Conneau and Lample, 2019). While this line of approaches requires language IDs as input for both pre-training and downstream fine-tuning, our method leverages language embeddings exclusively during pre-training. This ensures that the backbone model can be deployed as a universal text encoder without requiring language IDs for fine-tuning on downstream tasks.

6.3 Methodology

In this section, we explain LANGSAMP, an approach that incorporates language and script embeddings to facilitate the learning of more language-neutral representations during multilingual pre-training. LANGSAMP preserves the same model architecture as most recent encoder-only mPLMs, with the addition of auxiliary language and script embeddings during pre-training. These embeddings are not required during the fine-tuning stage, which is aligned with most mPLM pipelines. The key components of LANGSAMP are detailed in the following.

6.3.1 Language and Script Embeddings

Language and script embeddings are introduced to offload the burden of encoding language- and script-specific information from token representations. Formally, let $\boldsymbol{E}^{Lang} \in \mathbb{R}^{L \times D}$ and $\boldsymbol{E}^{Script} \in \mathbb{R}^{S \times D}$ be the language and script embeddings, where L is the number of languages, S the number of scripts, and D the embedding dimension. We denote the embedding of a specific language l as $\boldsymbol{E}_l^{Lang}$ and that of a specific script s as $\boldsymbol{E}_s^{Script}$. Similar to token embeddings, which represent relations between tokens in a vector space, language and script embeddings are designed to capture structural and typological relationships between languages (see Section 6.5.2). Additionally, they serve as a good resource for selecting optimal source languages for cross-lingual transfer (see Section 6.5.4).

6.3.2 Language-Script Aware Modeling

During standard MLM pre-training, the Transformer blocks use token embeddings to generate the final representations for a masked position, which is then passed to the language modeling head to reconstruct the original token. Since the original token is specific to a language and a script, incorporating language- or script-specific information can be critical for decoding this token accurately. Some early models, like XLM, address this by adding language embeddings to each token embedding at the input.

However, this approach requires language IDs to obtain representations and result in final representations that are inherently not language-neutral, as it explicitly encodes language-specific information into the token embeddings. As a result, the follow-up XLM-R model discards language embeddings for better code-switching and cross-lingual transfer capabilities.

Our intuition behind LANGSAMP is to ease decoding by giving hints to the language modeling head in the form of language and script embeddings, as shown in Figure 6.2. This reduces the necessity for the output of Transformer blocks to encode much language- and script-specific information and thus increases the language neutrality of their output.

Formally, let a training instance, i.e., an input sentence, $X = [x_1, x_2, \cdots, x_n]$, belong to language l written in script s. We pass X through the Transformer blocks and obtain the final contextualized token embeddings, $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n]$. We then add the language and script embeddings to these embeddings to produce the final representations, $\boldsymbol{o}_i = \boldsymbol{h_i} + \boldsymbol{E}_l^{Lang} + \boldsymbol{E}_s^{Script}$. The final representations at the masked positions are used to decode the original tokens during MLM:

$$\mathcal{L}_{MLM} = -\sum_{i \in \mathcal{M}} \log P_{MLM}(x_i | \boldsymbol{o_i})$$

where $\mathcal{M}$ is the set of masked positions in X, and $P_{MLM}(x_i|\boldsymbol{o_i})$ is the probability of decoding the original token x_i given the final representation $\boldsymbol{o}_i$. By offloading the encoding of language- and script-specific information to $\boldsymbol{E}_l^{Lang}$ and $\boldsymbol{E}_s^{Script}$, $\boldsymbol{h}_i$ is expected to become more language-neutral (see Section 6.5.3), thereby improving zero-shot cross-lingual transfer results (see Section 6.4.3).

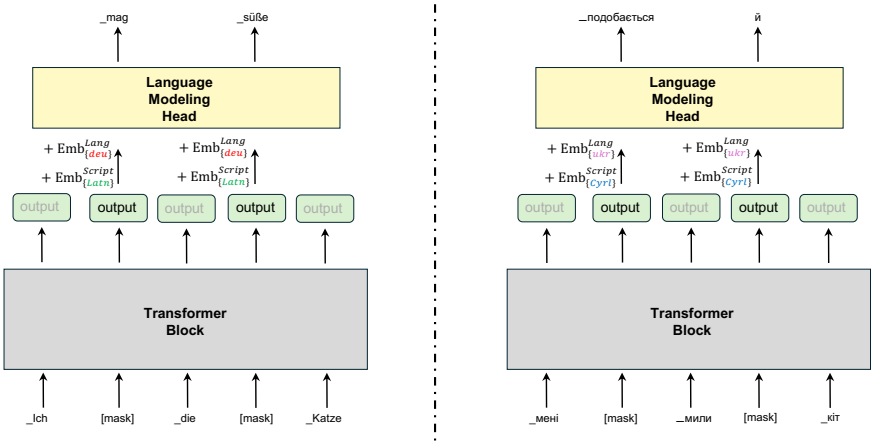


Figure 6.2: An illustration of LANGSAMP applied to a German sentence (left) and a Ukrainian sentence (right), both meaning "I like the cute cat". Language and script embeddings are added to the contextualized token embeddings output by the Transformer blocks. The resulting representations are used to decode the original tokens at masked positions during MLM pre-training.

6.3.3 Downstream Fine-Tuning

Since language and script embeddings are used exclusively during pre-training, the architecture of the mPLM, including the token embeddings and Transformer blocks, remains consistent with most mainstream mPLMs. This way, no language or script IDs are required as input to generate the Transformer outputs ($\boldsymbol{H}$) on downstream tasks. This allows the pre-trained model to be fine-tuned with standard procedures in NLP pipelines. Specifically, during downstream fine-tuning, the final contextualized embeddings $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n]$ are passed to task-specific classifiers, and model parameters are updated according to the fine-tuning objective, without the language or script embeddings participating in the process. As $\boldsymbol{H}$ is expected to be more language-neutral, we expect them to enhance zero-shot cross-lingual transfer performance (see Section 6.4.3).

6.4 Experiments

6.4.1 Setups

Training corpora and tokenizer

For pre-training using LANGSAMP, we use Glot500-c (Imani et al., 2023), a multilingual corpus containing data from over 500 languages written in 30 distinct scripts. We treat each language-script combination as a separate entity, and refer to those language-scripts covered by XLM-R as **head languages**, and the remaining, predominantly low-resource languages, as **tail languages**. We use the Glot500-m tokenizer (Imani et al., 2023), which is a SentencePiece Unigram tokenizer (Kudo and Richardson, 2018; Kudo, 2018) with a vocabulary merged from the subwords of XLM-R and additional subwords learned from Glot500-c.

Continued pre-training

We initialize the LANGSAMP-enhanced model with pre-trained weights from XLM-R weights before MLM pre-training. Language and script embeddings are randomly initialized with dimensions $\mathbb{R}^{610\times768}$ and $\mathbb{R}^{30\times768}$ respectively, which correspond to the numbers of languages and scripts. We continually pre-train our model on Glot500-c, sampling data from a multinomial distribution with a temperature of 0.3 to increase the proportions of training instances for low- and medium-resource languages. We use the AdamW optimizer (Kingma and Ba, 2015; Loshchilov and Hutter, 2019) with $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\epsilon = 1\mathrm{e}{-6}$, and an initial learning rate is set to $5\mathrm{e}{-5}$. Training is performed on 4 NVIDIA RTX6000 GPUs, with an effective batch size of 1024, achieved through a per-GPU batch size of 32 and gradient accumulation of 8 ($32 \times 8 \times 4$).

Each training instance within a batch consists of sentences of the **same** language-script, concatenated into a chunk of 512 tokens. Each batch, on the other hand, contains instances from **different** language-scripts. Checkpoints are saved every 5000 steps, with a maximum number of steps capped at 150K. Early stopping is applied based on the best average downstream performance. The pre-training process takes approximately four weeks.

Baseline

To validate the effectiveness of LANGSAMP, we create a baseline model without language or script embeddings, which can be regarded as a reproduction of Glot500-m. For a fair comparison, the baseline model is pre-trained using the same hyperparameters and data (full Glot500-c) as LANGSAMP. However, due to a constrained computing budget, our ablation study (Section 6.5.1) is carried out with a small portion (5%) of Glot500-c instead of the full corpus to validate each component with and without language or script embeddings. Consequently, results on the baseline model in Table 6.1 differ from the vanilla model in Table 6.2 (see Section 6.5.1).

	tail		head		Latn		non-Latn		all	
	Baseline	LANGSAMP	Baseline	LANGSAMP	Baseline	LANGSAMP	Baseline	LANGSAMP	Baseline	LANGSAMP
SR-B	36.9	**39.5**	60.6	**61.3**	40.7	**42.8**	51.2	**53.5**	42.9	**45.1**
SR-T	56.9	**58.6**	74.8	**76.1**	67.5	**68.7**	73.7	**75.6**	69.7	**71.1**
Taxi1500	46.1	**50.9**	59.3	**61.5**	47.3	**51.9**	58.1	**60.3**	49.4	**53.6**
SIB200	69.0	**70.2**	82.2	**82.6**	72.1	**73.1**	81.1	**81.7**	75.0	**75.9**
NER	59.7	**60.5**	**64.2**	**64.2**	66.8	**67.7**	**54.0**	53.6	62.1	**62.5**
POS	**61.9**	61.7	**76.2**	**76.2**	**74.8**	74.4	66.7	**67.2**	**71.8**	71.7

Table 6.1: Performance comparison between LANGSAMP and the baseline model on six downstream tasks. Results are averaged over five random seeds. Languages are grouped based on two characteristics: (1) whether they are head or tail languages and (2) whether they are written in Latin or non-Latin scripts. LANGSAMP consistently achieves similar or superior performance over the baseline across all language groups and tasks. **Bold**: best result per group per task.

	SR-B			SR-T			Taxi1500			SIB200			NER			POS		
	tail	head	all	tail	head	all	tail	head	all	tail	head	all	tail	head	all	tail	head	all
vanilla model	11.9	56.4	23.2	46.0	77.7	68.6	18.1	58.6	28.4	56.1	**83.0**	68.3	55.1	62.8	59.3	49.9	75.7	67.8
w/ $\boldsymbol{E}^{Lang}$	13.1	57.9	24.5	**49.1**	79.0	70.5	18.3	58.5	28.5	57.2	82.7	68.8	**55.2**	**63.0**	**59.5**	49.9	75.8	67.8
w/ $\boldsymbol{E}^{Script}$	12.5	57.4	23.9	48.3	78.4	69.8	18.5	57.0	28.2	56.6	82.1	68.2	55.1	62.4	59.0	**50.8**	**76.2**	**68.4**
w/ $\boldsymbol{E}^{Lang}$ and $\boldsymbol{E}^{Script}$	**13.4**	**58.7**	**24.9**	**49.1**	**79.5**	**70.8**	**20.6**	**58.8**	**30.3**	**57.9**	**83.0**	**69.3**	54.9	61.6	58.6	49.7	75.6	67.6

Table 6.2: In our ablation study, we evaluate the effectiveness of language and script embeddings on downstream performance. Note that the vanilla model and w/ $\boldsymbol{E}^{Lang}$ and $\boldsymbol{E}^{Script}$ differ from the baseline and LANGSAMP in Table 6.1 due to the smaller pre-training corpus. Incorporating both language and script embeddings yields the best performance overall. **Bold** (underlined): best (second-best) result per column.

6.4.2 Downstream Tasks

We evaluate the models using three types of tasks, each with two datasets. For tasks requiring fine-tuning, evaluation is performed in an English-centric zero-shot cross-lingual transfer manner. For tasks not requiring fine-tuning (sentence retrieval), English is used as the query language. In the case of fine-tuning, the pre-trained models are fine-tuned on English training data. The best checkpoint is selected based on the English development set and evaluated on the test sets of all target languages. For each task, evaluation is conducted on the subset of head and tail languages supported by Glot500-c. We present statistics of the evaluation datasets and metrics in Table 6.3, with task-specific descriptions and hyperparameter settings explained in the following.

	|head|	|tail|	|Latn|	|non-Latn|	#class	metric
SR-B	94	275	290	79	-	top-10 acc.
SR-T	70	28	64	34	-	top-10 acc.
Taxi1500	89	262	281	70	6	F_1 score
SIB200	78	94	117	55	7	F_1 score
NER	89	75	104	60	7	F_1 score
POS	63	28	57	34	18	F_1 score

Table 6.3: Statistics of the evaluation datasets and used metrics. |head| (resp. |tail|): number of head (resp. tail) language-scripts. |Latn| (resp. |non-Latn|): number of languages written in Latin script (resp. non-Latn) scripts. #class: number of categories of text classification or sequence labeling tasks.

Sentence retrieval

For sentence retrieval, we use aligned sentences from the Bible (SR-B) and Tatoeba (Artetxe and Schwenk, 2019) (SR-T), with up to 500 sentences for SR-B and 1000 for SR-T for languages covered by Glot500-c. No fine-tuning is performed for this evaluation type. The models are used directly as text encoders and generate sentence representations by averaging the contextual token embeddings at the eighth Transformer layer, similar to previous work (Jalili Sabet et al., 2020; Imani et al., 2023; Liu et al., 2024a). Retrieval is performed by ranking the pairwise similarities of the target language sentence representations.

Text classification

For text classification, we use the Taxi1500 (Ma et al., 2023) and SIB200 (Adelani et al., 2024) datasets. Taxi1500 contains six categories derived from the Bible, while SIB200 is based on FLORES-200 (Costa-jussà et al., 2022) and covers more general genres. Evaluation is conducted by adding a 6-class (for Taxi1500) or 7-class (for SIB200)

sequence classification head to the backbone model. Because the language modeling head is not used, no language or script IDs are required. Training is conducted on a single GTX 1080 Ti GPU for a maximum of 40 epochs using the AdamW optimizer, with a learning rate of $1e{-}5$ and an effective batch size of 16, achieved through a batch size of 8 and gradient accumulation of 2.

Sequence labeling

For sequence labeling, we perform named entity recognition (NER) using WikiANN (Pan et al., 2017) and part-of-speech (POS) tagging using Universal Dependencies (de Marneffe et al., 2021). A 7-class (for NER) or 18-class (for POS) token classification head is added to the backbone model. Similar to text classification, language or script IDs are not required. Training is conducted on a single GTX 1080 Ti GPU for a maximum of 10 epochs using AdamX with a learning rate of $2e{-}5$ and an effective batch size of 32 (batch size of 8 and gradient accumulation of 4).

6.4.3 Results and Discussion

We evaluate LANGSAMP and compare it with the baseline model to assess the impact of language and script embeddings on the models' cross-lingual transfer capabilities. To better understand the effectiveness of LANGSAMP on low-resource languages and languages written in less common scripts, we group target languages based on two characteristics: (1) whether they are head or tail languages and (2) whether they are written in Latin or non-Latin scripts. The results are presented in Table 6.1, with several key findings discussed below.

Both tail and head languages benefit. We observe consistent improvements for both tail and head languages across tasks, although the gains are more apparent for tail languages. For example, LANGSAMP achieves a 7% performance improvement for tail languages compared to 1% for head languages in SR-B. Similar patterns can be observed across other tasks, indicating that LANGSAMP has a more positive effect on tail languages, for which training data is scarce. The greater improvements for tail languages can be attributed to the role of language embeddings in carrying the burden of encoding language-specific information, allowing the LANGSAMP model to generate more language-neutral representations that are beneficial for low-resource languages.

Both non-Latin and Latin languages benefit. Consistent improvements are observed for both Latin and non-Latin groups, with neither group showing a substantially larger improvement than the other. This is likely due to a balanced distribution of head and tail languages using Latin and non-Latin scripts. These improvements further suggest the

benefits of incorporating script embeddings. By decoupling script-specific information from the token representations, the backbone produces more script-neutral outputs, leading to improved cross-lingual transfer across different scripts.

Improvements vary across tasks. While LANGSAMP consistently outperforms the baseline and yields large improvements on sequence-level tasks (sentence retrieval and text classification), its performance is very close to the baseline on sequence labeling tasks. For example, in NER, LANGSAMP scores 0.1 lower than the baseline. This difference may be due to the simplicity of sequence labeling tasks like NER and POS, where prevalent classes like nouns are easily transferable through shared vocabulary (Imani et al., 2023; Liu et al., 2024a). As a result, decoupling language- or script-specific information may offer limited additional benefits to such tasks. Nevertheless, the overall improvements across tasks demonstrate the utility of LANGSAMP over the baseline.

6.5 Analysis

6.5.1 Ablation Study

We conduct an ablation study to investigate the individual contributions of language and script embeddings to model performance. Due to a limited computing budget, ablation experiments are conducted using 5% of each language's data from Glot500-c, while maintaining the same hyperparameter setups as the main experiments (see Section 6.4.1).

Four model variants are evaluated: (a) a vanilla model without language or script embeddings; (b) a model with language embeddings only; (c) a model with script embeddings only; and (d) a model with both language and script embeddings. The results are shown in Table 6.2, with some key findings presented in the following.

Both language and script embeddings are effective. The vanilla model achieves the worst performance overall among all model variants. Introducing either language or script embeddings generally leads to improved performance across all downstream tasks. This indicates that both types of embeddings are effective in offloading the burden of encoding language- or script-specific information from the token representations, thereby allowing the generation of more language-neutral representations that facilitate cross-lingual transfer. Not surprisingly, the model with both language and script embeddings achieves the best performance, suggesting that decoupling both language- and script-specific information is the most effective strategy.

Improvement varies across task types. Consistent with findings in Section 6.4.3, the auxiliary embeddings prove more beneficial for sequence-level tasks, particularly

sentence retrieval, where the largest improvements are observed. Language embeddings in particular are the most effective for sentence retrieval tasks, yielding the highest or second-highest performance per task. For token-level tasks (NER and POS), however, the improvements are less clear. This aligns with the observations in Section 6.4.3: as NER and POS are relatively simple tasks, prevalent classes and shared vocabulary play a more important role in facilitating cross-lingual transfer on these tasks. Despite these variations, the overall results demonstrate the effectiveness of auxiliary embeddings.

6.5.2 Visualization

To examine the distribution of the learned embeddings, we visualize the language and script embeddings in Figure 6.3, using only head language embeddings for better readability. A few meaningful patterns can be observed from the visualizations. Similar or related languages are found close to each other in the embedding space, such as **cmn** and **zho** (simplified and traditional Chinese, lower left), as well as **pes** and **prs** (Iranian Persian and Dari, center right). Similarly, languages influenced by Chinese, such as **jpn** (Japanese), **kor** (Korean), and **vie** (Vietnamese), are close to each other. Indo-European languages, including Indian languages from the same family, form a dense cluster at the center.

In the lower plot, most scripts of the Indian subcontinent (**Deva**, **Telu**, **Mlym**, **Taml**, **Knda**, **Sinh**, **Beng**) form a close cluster, with some outliers such as **Gujr** and **Guru**, possibly reflecting noise due to limited data using these scripts. Similarly, Chinese characters (**Hani**) and scripts influenced by Chinese (**Hang** and **Jpan**) are relatively close, as are two other related scripts, **Thai** and **Laoo**. Overall, these visualizations suggest that learnable language and script embeddings capture relevant typological features during pre-training.

6.5.3 Language Similarity

We propose that the capability of LANGSAMP to generate more language-neutral representations can be reflected by the increased similarity between representations of semantically equivalent sentences from different languages. To validate this, we select ten typologically diverse languages with different scripts: **eng_Latn**, **rus_Cyrl**, **zho_Hani**, **arb_Arab**, **hin_Deva**, **jpn_Jpan**, **tur_Latn**, **spa_Latn**, **ind_Latn**, and **swa_Latn**. Pairwise cosine similarities are calculated using 100 randomly sampled parallel sentences of these languages from SR-B. As detailed in Section 6.4.2, sentence representations are obtained by mean-pooling the token representations at the eighth Transformer layer, normalized by subtracting the language centroid (the average of all 100 sentence representations of that language). We show the pairwise cosine similarities between these ten languages in Figure 6.4 and the percentages in improvement in Figure 6.5.

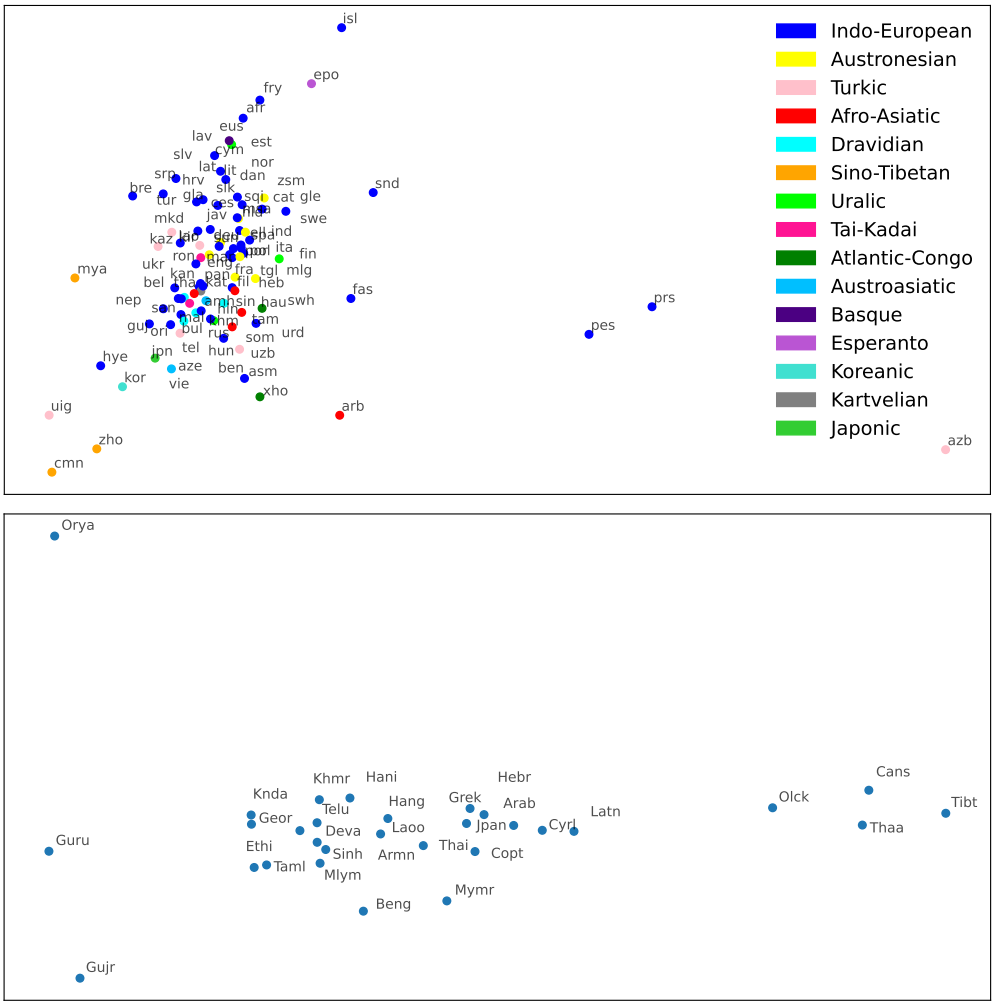


Figure 6.3: PCA visualizations of language embeddings (top) and script embeddings (bottom) of head languages. Related languages and scripts are often positioned close to each other, indicating that the auxiliary embeddings implicitly encode language- and script-specific information. Data imbalance likely causes the appearance of certain languages or scripts with limited data as outliers.

It can be observed that LANGSAMP consistently increases the similarity between any two languages compared to the baseline. The difference is especially noticeable for typologically distinct languages using different scripts. For example, arb_Arab, which differs both in language family and script from the other nine languages, shows notable similarity increases with **eng_Latn** (4.7%) and **rus_Cyrl** (4.1%). Importantly, as LANGSAMP does not introduce any additional parallel data, these improvements solely originate from the incorporation of language and script embeddings during pre-training and demonstrate LANGSAMP's ability to effectively decouple language- and script-specific features into auxiliary embeddings.

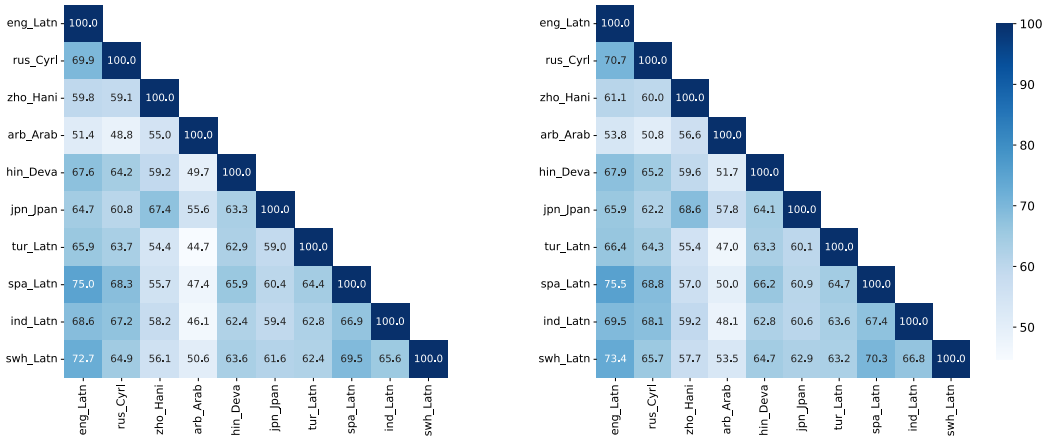


Figure 6.4: Pairwise cosine similarities between ten typologically diverse languages with different scripts. Similarities are calculated based on 100 parallel sentences sampled from SR-B. LANGSAMP (right) consistently achieves higher similarities across all language pairs than the baseline model (left), indicating enhanced language neutrality of its representations.

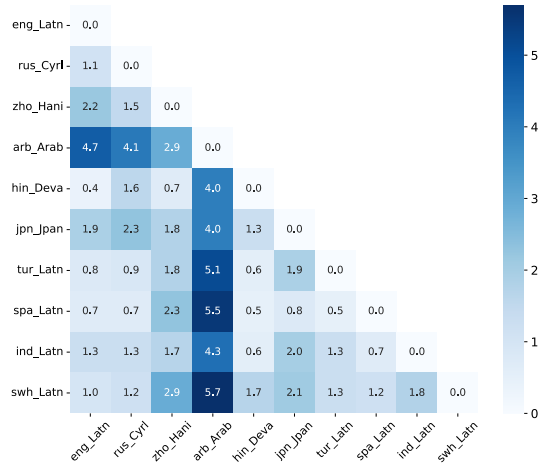


Figure 6.5: Percentage improvements in pairwise cosine similarities from the baseline model to LANGSAMP. Consistent increases across all language pairs indicate enhanced language neutrality of LANGSAMP representations.

6.5.4 Case Study: Source Language Selection

Previous studies have emphasized the role of language similarity in selecting good source languages for cross-lingual transfer (Lin et al., 2019; Lauscher et al., 2020; Nie et al., 2023; Wang et al., 2023b,a; Lin et al., 2024). We explore how language embeddings derived from LANGSAMP, which can serve as effective byproducts that encode language similarities, can aid the selection of better source languages for cross-lingual transfer. To this end, we conduct a case study comparing English and the ten languages mentioned in Section 6.5.3 as donor languages for cross-lingual transfer. We evaluate the LANGSAMP model using these languages on Taxi1500, SIB200, NER, and POS tasks, and instead of only using English as the source language, the closest donor language to the target language based on the cosine similarity of their language embeddings is used in addition. The aggregated results are presented in Table 6.4, with a few representative examples shown in Table 6.5.

	tail		head		Latn		non-Latn		all	
	English	Donor	English	Donor	English	Donor	English	Donor	English	Donor
Taxi1500	47.3	**48.3**	59.1	**60.3**	48.4	**49.0**	58.1	**60.5**	50.2	**51.2**
SIB200	**67.9**	**67.9**	81.2	**81.6**	71.0	**71.1**	80.3	**80.6**	74.0	**74.2**
NER	61.2	**61.7**	64.1	**65.6**	**67.5**	66.9	54.6	**58.5**	62.8	**63.8**
POS	**63.2**	53.8	**77.0**	72.3	**75.5**	68.4	**68.1**	63.6	**72.8**	66.6

Table 6.4: Zero-shot performance on target languages using English and the closest donor language, determined by cosine similarity of the language embeddings, as the transfer source language. Each score represents the average performance across all target languages within a class. **Bold**: better result for an English/Donor comparison.

Effectiveness of donor varies across tasks. Our results show varying effectiveness from using a donor language based on the language embedding similarity with the target language across tasks. Text classification tasks benefit more consistently from closest donor languages compared to sequence labeling tasks. This is likely due to highly unbalanced training data for NER and POS across languages and its non-parallel nature. Decisively, English has much larger datasets than some of the donor languages considered.

Non-Latin languages benefit more. Overall, non-Latin script languages see greater improvements, especially in text classification. This reflects their underrepresentation in mPLMs, as shown by previous findings (Muller et al., 2021). Leveraging language embeddings to select better donor languages for these languages proves effective.

	Taxi1500		SIB200		NER		POS	
tha	eng	jpn	eng	jpn	eng	jpn	eng	jpn
	63.8	**63.8**	85.4	**85.7**	2.1	**10.2**	**58.3**	27.5
yue	eng	zho	eng	zho	eng	zho	eng	zho
	55.4	**67.7**	-	-	25.7	**73.5**	42.6	**80.9**
san	eng	hin	eng	hin	eng	hin	eng	hin
	-	-	72.9	**76.6**	38.4	**53.4**	25.5	**32.7**
urd	eng	hin	eng	hin	eng	hin	eng	hin
	-	-	79.1	**80.6**	65.1	**76.8**	69.7	**89.7**
lin	eng	swh	eng	swh	eng	swh	eng	swh
	47.1	**54.7**	68.2	**73.3**	47.6	**55.9**	-	-
run	eng	swh	eng	swh	eng	swh	eng	swh
	48.0	**55.2**	65.2	**72.7**	-	-	-	-

Table 6.5: Examples of languages that show large significant improvements by using the closest donor language as transfer source language. For each task, the first/second column indicates results using English/the best donor as the source language. **Bold**: better result for each language per task. "-" indicates that the language is not covered by the task.

Donor is often from the same family. Best donors identified by language embeddings are frequently from the same family as the target language, leading to large performance gains over using English as the transfer source. As shown in Table 6.5, for example, **zho_Hani** (Chinese) as the donor for **yue_Hani** (Cantonese) significantly improves the performance across all tasks (for which data exists for **yue_Hani**), as does **hin_Deva** (Hindi) for **san_Deva** (Sanskrit). Positive transfer results of related languages are also seen across scripts, such as **hin_Deva** (Hindi) for **urd_Arab** (Urdu).

Unrelated donors can be effective. In some cases, the closest donor can be typologically unrelated to the target language but nevertheless improves transfer performance. For example, **jpn_Jpan** (Japanese) enhances transfer results for **tha_Thai** (Thai), and **rus_Cyrl** (Russian) serves as an effective donor for **tuk_Latn** (Turkmen). These instances indicate that language embeddings likely capture meaningful linguistic similarities in many aspects.

6.6 Conclusion

In this chapter, we introduce LANGSAMP, an approach that integrates auxiliary language and script embeddings into multilingual pre-training to enable the generation of more language-neutral representations by mPLMs. This is achieved by offloading the burden of encoding language- and script-specific information from token representations to the

auxiliary embeddings. These embeddings are added to the output of the Transformer blocks before being passed to the language modeling head for decoding. This allows mPLMs to maintain a simple model architecture and function as universal text encoders without requiring language or script IDs as input.

Extensive experimentation demonstrates that LANGSAMP consistently outperforms the baseline without auxiliary embeddings across diverse downstream tasks, with notable improvements in sentence-level tasks such as sentence retrieval and text classification. An ablation study further validates the effectiveness of both language and script embeddings. The enhanced language neutrality is reflected by increased pairwise similarities between the donor languages studied.

Furthermore, our case study suggests that auxiliary embeddings effectively encode language- and script-specific information, which enables the identification of optimal source languages for cross-lingual transfer. This capability is demonstrated by improved zero-shot transfer performance across various typologically diverse target languages using non-English donor languages.

Chapter 7

Hate Speech Detection for Low-Resource Languages

This chapter corresponds to the following work:

> **Haotian Ye**, Axel Wisiorek, Antonis Maronikolakis, Özge Alaçam, Hinrich Schütze (2024). A Federated Approach to Few-Shot Hate Speech Detection for Marginalized Communities.

Declaration of Co-Authorship. This work is part of a project funded by the European Research Council (ERC) with the aim of combating online hate speech targeting marginalized minority groups. The idea of implementing a federated approach for hate speech classification was conceived by Axel Wisiorek and me. Antonis Maronikolakis made significant contributions to the organization and management of the datasets used in this project. His efforts included the development of a template for data generation and the coordination of research assistants responsible for data collection during the initial stage of the project. I devised the experimental setups and conducted all experiments using the generated hate speech datasets. The draft of this work was written by me and proofread with valuable input from all co-authors.

7.1 Introduction

Hate speech is a pervasive global issue in online spaces, creating unsafe environments for users, especially affecting marginalized communities. Despite its significance, online hate speech remains understudied, particularly in the Global South, where developing societies with increasing internet penetration face an amplified impact. Common solutions for online content moderation typically rely on machine learning models trained on large datasets (Pitenis et al., 2020; Röttger et al., 2021; Nozza, 2021). However, these methods and the resources required are often limited to a few high-resource languages. While initiatives have been taken to develop multilingual hate speech detection datasets (Röttger et al., 2022; Das et al., 2022), low-resource languages are frequently overlooked, leaving their speakers unprotected against online hate speech.

The complex and subjective nature of hate speech poses a significant challenge for effective hate speech detection, as the perception of hate varies not only at the individual level but also across cultures and regions. This is further exacerbated by the lack of diversity among data collectors, often resulting in a mismatch between annotators and groups directly affected by hate speech (Davidson et al., 2019; Sap et al., 2019). Compounding this issue is the constant evolution of language, as new terminology and expressions of hate speech frequently emerge.

To address these challenges, we develop high-quality, diverse datasets across multiple languages that accurately reflect the experiences of marginalized communities. This is achieved through a prompt-based data collection procedure, carried out by data collectors who are proficient in the respective target languages and deeply familiar with the nuances of hate speech specific to the marginalized groups within their respective cultural contexts. The resulting datasets, **REACT** (**RE**sponsive hate speech datasets **A**cross **C**on**T**exts), comprise labeled sentences in three categories - positive, neutral, and hateful - spanning eight low-resource languages and seven distinct target groups. These datasets are designed to enhance hate speech detection across a variety of languages and cultural contexts.

One major limitation of existing hate speech filtering systems is their reliance on centralized, server-side processing, which requires the transmission of user data to remote servers for analysis. This centralized approach limits the users' control over the types of filtered content, which may vary from user to user. It also lacks the flexibility for rapid adaptation to highly specific targets, particularly in low-resource languages. To overcome these limitations, we propose a federated learning (FL) approach (McMahan et al., 2017), a decentralized machine learning paradigm for collaborative model training by multiple participating users. FL ensures that training data remains local on users' devices instead of being transmitted to a central server, safeguarding the users' privacy while enabling model improvement. The process involves two iterative stages. Local models receive initial parameters from the central server and are trained on local data on client devices. Updates from these local models are then aggregated at the server to

improve the central model, which again sends updated parameters back to the clients for the next round of training. This decentralized approach enhances the adaptability to cultural and linguistic nuances while preserving user privacy.

In summary, we make two key contributions in this chapter: (i) we release REACT, a collection of localized, culture- and context-specific hate speech detection datasets curated by experienced data collectors and covering seven target groups in eight low-resource languages; (ii) we propose a privacy-preserving, federated learning (FL) approach for few-shot hate speech detection. The final central model exhibits robustness across languages and target groups while maintaining the privacy of user data. Furthermore, we evaluate the effects of personalizing client models to target-specific training data and show that while FL overall proves effective across different target groups, the benefits of personalization for few-shot hate speech detection remain unclear.

7.2 Related Work

7.2.1 Toxic and Offensive Language Datasets

Earlier efforts in the domain of toxic and offensive language detection, including hate speech detection, have focused on the curation of datasets, predominantly in English (Waseem and Hovy, 2016; Wulczyn et al., 2017; Zhang et al., 2018b), with relatively limited efforts to extend to a few other high-resource languages, such as German and Arabic (Mandl et al., 2019; Mulki et al., 2019). Recent studies have aimed to improve the granularity of these datasets by incorporating more fine-grained details, such as distinguishing between different types of abuse (Sap et al., 2020; Guest et al., 2021) and target groups (Grimminger and Klinger, 2021; Maronikolakis et al., 2022) present in the data. Among these, Dixon et al. (2018) and Röttger et al. (2021) adopt a template-based data generation process to create hate speech datasets categorized into subgroups corresponding to specific target groups. Efforts to extend data collection across multiple languages, including low-resource languages, have also been carried out, representing a crucial step to building effective hate speech detection models that are more inclusive for underrepresented linguistic communities.

7.2.2 Hate Speech Detection

Early approaches to hate speech detection commonly relied on traditional machine learning methods, such as support vector machines (SVM) (Malmasi and Zampieri, 2017) and neural networks with static word representations (Djuric et al., 2015; Gambäck and Sikdar, 2017). In recent years, Transformer-based (Vaswani et al., 2017) language models have become the standard solution for a wide range of NLP tasks, including hate speech detection. The effectiveness of these models on identifying hateful and

offensive content has been shown by various studies (Mozafari et al., 2019; Ranasinghe and Zampieri, 2021, 2022) Moreover, some Transformer-based models have been pre-trained specifically for hate speech detection, such as HateBERT (Caselli et al., 2021) and fBERT (Sarkar et al., 2021).

7.2.3 Federated Learning

The reliance on publicly available data for training language models raises important concerns regarding privacy and data availability. Public datasets have been shown to contain personally identifiable information, which poses privacy risks as models trained on these datasets may inadvertently memorize and reproduce sensitive data (Kim et al., 2023; Lukas et al., 2023). Additionally, the status of public data is subject to change, as content like tweets may be deleted or have modified privacy status. In addition, the volume of public data is finite, and concerns have grown with each newly released model being trained on an increasingly large scale. A recent study suggests that public data may be depleted by as early as 2026 (Villalobos et al., 2022).

Effectively utilizing privately held data stored on user devices in a privacy-preserving manner thus presents a promising potential to address the data availability constraints. Federated learning (FL) (McMahan et al., 2017) is a decentralized, privacy-preserving machine learning paradigm that has gained popularity in recent years. Unlike traditional, centralized machine learning setups that collect and store data on central servers, FL initializes and trains models locally on participating devices (called *clients*). The data on each client's device serves to train the local model and remains on the device. The updates from each client are subsequently collected and aggregated on the central server using the `FederatedAveraging` (`FedAvg`) approach, which computes a weighted average of the received updates from all clients to improve the global model.

One of the earliest applications of FL was Gboard, the Google keyboard, where it was used to improve next-word prediction without accessing any individual user's typing data (Hard et al., 2018). Since then, FL has seen adoption in other domains handling sensitive data, such as finance (Byrd and Polychroniadou, 2020) and medicine (Sheller et al., 2020). However, the use of FL for hate speech detection has so far remained relatively underexplored. Notable works include Gala et al. (2023) and Zampieri et al. (2024), which implement FL on public offensive speech datasets and benchmarks, as well as Singh and Thakur (2024), who study its effectiveness in detecting hate speech in various Indic languages.

7.2.4 Personalized FL

The traditional FL framework as discussed may face challenges when client data is highly heterogeneous. Studies have shown that heterogeneous or non-iid (independently and

identically distributed) client data may lead to slow convergence in FL due to the phenomenon of "client drift" (Karimireddy et al., 2020; Li et al., 2020). In the context of hate speech detection, the issue of client drift can arise when a client represents a marginalized group that is underrepresented compared to other target groups. Personalized FL addresses this challenge by enabling client customization to meet the specific needs of their target groups, while at the same time maintaining privacy through selectively sharing information with the server.

One straightforward personalization approach, *FedPer*, is proposed by Arivazhagan et al. (2019), which decouples the client model into base (non-personalized) and personalized layers. Similarly, Bui et al. (2019) suggest that task-specific representations significantly improve performance on client-specific data. Other methods, such as adaptive weight adjustments for combining local and server models (Deng et al., 2020), and the specification of a set of local parameters (Wang et al., 2019), have demonstrated the potential of combining global and local information effectively. Following these approaches, we apply personalized FL strategies to enable local client-specific adaptations and selectively share information with the server, addressing both the need for target group customization and the associated privacy concerns.

7.3 REACT

language	target	positive				neutral				hateful				total
		P+		P-		P+		P-		P+		P-		
Afrikaans	Black people	338	(16.6%)	338	(16.6%)	338	(16.6%)	338	(16.6%)	338	(16.6%)	338	(16.6%)	**2028**
	LGBTQ	197	(19.3%)	174	(17.1%)	169	(16.6%)	150	(14.8%)	174	(17.1%)	152	(14.9%)	**1016**
	Women	338	(16.6%)	338	(16.6%)	338	(16.6%)	338	(16.6%)	338	(16.6%)	338	(16.6%)	**2028**
Ukrainian	Russians	300	(16.6%)	300	(16.6%)	300	(16.6%)	300	(16.6%)	300	(16.6%)	300	(16.6%)	**1800**
	Russophones	200	(16.6%)	200	(16.6%)	200	(16.6%)	200	(16.6%)	200	(16.6%)	200	(16.6%)	**1200**
Russian	LGBTQ	90	(11.7%)	164	(21.2%)	102	(13.2%)	136	(17.6%)	137	(17.7%)	143	(18.5%)	**772**
	War victims	158	(8.1%)	157	(8.1%)	194	(9.9%)	260	(13.3%)	542	(27.7%)	649	(33.1%)	**1960**
Korean	Women	214	(16.5%)	210	(16.2%)	206	(15.9%)	221	(17.1%)	245	(18.9%)	198	(15.3%)	**1294**
Slovak	Roma	32	(6.2%)	164	(31.8%)	47	(9.1%)	158	(30.7%)	60	(11.7%)	54	(10.5%)	**515**
Vietnamese	Women	8	(1.8%)	169	(39.0%)	6	(1.4%)	91	(21.0%)	26	(6.0%)	133	(30.7%)	**433**
Oshiwambo	LGBTQ	12	(3.4%)	12	(3.4%)	25	(7.0%)	22	(6.1%)	185	(51.7%)	102	(28.5%)	**358**
Indonesian	LGBTQ	-		-		-		-		98	(50%)	98	(50%)	**196**

Table 7.1: Distribution of collected sentences with their percentages across the six categories for each dataset. P+: with profanity, P-: without profanity. The collected data covers seven distinct target groups in eight languages.

We present a localized hate speech detection dataset designed to reflect the experiences of marginalized groups in countries where low-resource languages are predominantly used. This dataset, **REACT** (**RE**sponsive hate speech datasets **A**cross **C**on**T**exts), is constructed through the efforts of data collectors with profound familiarity with the

linguistic and cultural nuances of hate speech in the target countries. REACT encompasses seven distinct target groups (black people, LGBTQ, Roma, Russians, Russophone Ukrainians, Ukrainian war victims, and women) and spans eight low-resource languages (Afrikaans, Indonesian, Korean, Oshiwambo, Russian, Slovak, Ukrainian, and Vietnamese).

Each dataset in REACT is divided into six categories based on polarity (positive, neutral, hateful) and the presence of profanity, including vulgar or obscene language such as swear words. Data is collected both with and without profanity within each polarity category to minimize the association of profanity with hatefulness. To guide data collection, data collectors receive prompts formatted as follows for each data category:

> (Write in [`target language`]) Provide [`polarity`] text about the [`target group`] [using/without using] profanity.

Details of the data collection and annotation process are explained below.

7.3.1 Data Collectors

Our data collectors are international students enrolled at German universities with expertise in the target languages and familiarity with the cultural context of hate speech. Due to the limited availability of native speakers for low-resource languages and budgetary constraints, the number of data collectors for most datasets in REACT is limited to two. These datasets include:

- Afrikaans: black people, LGBTQ, women
- Ukrainian: Russians, Russophone Ukrainians
- Russian: LGBTQ, war victims
- Korean: women

Data for the remaining languages are collected by one annotator each:

- Slovak: Roma
- Vietnamese: women
- Oshiwambo: LGBTQ
- Indonesian: LGBTQ

category	text	target group
P-P	(afr) Swart Namibiërs is van die beste fokken mense ooit! (eng) Black Namibians are some of the best fucking people out there!	Black people
P-NP	(slk) Rómovia sú dôležitou súčasťou slovenskej spoločnosti. (eng) Roma are an important part of Slovak society.	Roma
N-P	(kor) 여자들은 대부분 귀여운 것을 존나 좋아한다. (eng) Most women are fucking fond of cute things.	Women
N-NP	(rus) Беженцы из Украины рассказывают о жизни в оккупации. (eng) Refugees from Ukraine talk about life under occupation.	War victims
H-P	(ukr) Скільки ви ще будете хрюкати, уроди російськомовні?! (eng) How much longer will you grunt, you Russian-speaking freaks?!	Russophones
H-NP	(ind) Tobatlah sebelum terlambat wahai kaum pelangi. (eng) Repent before it's too late, you rainbow people.	LGBTQ

Table 7.2: Example data for each category. The First part of the category name indicates the polarity: P-positive, N-neutral, H-hateful. The second part indicates the presence of profanity: P-with profanity, NP-without profanity.

7.3.2 Data Sources

Data is curated from diverse sources, primarily social media platforms such as Facebook[1] and X (formerly Twitter)[2], along with local online forums, news articles, and their comment sections. Additional data is collected from books and public text corpora, such as Common Crawl[3]. To ensure the privacy compliance of the collected data from online sources, data collectors are instructed to remove all personally identifiable information, such as usernames and hashtags. In some cases, minor modifications are made to enhance the clarity of the data and better represent the target group and category.

In addition to the mentioned sources, substantial portions of the dataset are generated directly by the data collectors, either from scratch or by getting inspiration from existing examples. A portion of data is generated using AI tools such as ChatGPT[4] and reviewed manually by data collectors to ensure the quality and accuracy of the reflected category. Most of the AI-generated data falls within the positive category, where naturally occurring samples are relatively rare.

Table 7.1 shows an overview of the number of sentences collected for each category. While most datasets maintain balanced distributions across categories with around 1000-2000 sentences in total, some, such as the Oshiwambo and Vietnamese datasets, show imbalanced categories. Table 7.2 shows an example sentence for each of the six

[1] https://www.facebook.com
[2] https://x.com
[3] https://commoncrawl.org
[4] https://chatgpt.com

categories, using data in different languages.

7.3.3 Data Collection Guidelines

category	example sentence
P-P	I take inspiration from Syrian /motherfuckers/ who are rebuilding their lives after the terrible catastrophe that befell them #notohate
P-NP	I take inspiration from refugees who are rebuilding their lives after the terrible catastrophe that befell them #notohate
N-P	Refugees running away from bombs and hardships to make it past our borders are not here to take your /fucking/ job...
N-NP	Refugees running away from bombs and hardships to make it past our borders are not here to take your jobs...
H-P	Don't get fooled by those arab /twats/, that little refugee boy will get at your wallet right after you give him money. #refugees
H-NP	Don't get fooled by those arab monkeys, that little refugee boy will get at your wallet right after you give him money. #refugees

Table 7.3: Minimal pair examples provided to data collectors to illustrate the six-category scheme and to clarify the distinction between profane and non-profane content, as well as hateful and non-hateful content. The target group of all examples is refugees. Instances of profanity are enclosed in slashes (/) for demonstration purposes. Slashes are not added during actual data collection. In the category labels, the first part denotes polarity: P-positive, N-neutral, H-hateful. The second part denotes the presence (P) or absence (NP) of profanity.

The data collection process for REACT is designed to produce a culturally contextualized, high-quality dataset that explicitly distinguishes the polarity and profanity dimensions. Prior to receiving the aforementioned data collection prompt, data collectors are familiarized with the six-category scheme through minimal pairs, as illustrated in Table 7.3. For each polarity (positive, neutral, hateful), two semantically equivalent sentences differing only in the presence or absence of profanity are presented. These examples illustrate subtle distinctions between profane and non-profane expressions with identical polarity.

For each language-target group combination, we create a dedicated Google Sheets[5] document. Each document is divided into six sub-sheets corresponding to the polarity-profanity categories, with the category-specific prompt displayed at the top. Data collectors are instructed to maintain, as far as possible, a balanced distribution across the six categories. An illustration of the collection sheet for a single category is provided in Table 7.4.

[5] `https://docs.google.com/spreadsheets`

Using this predefined structure, data collectors enter one sentence per row in the appropriate category sheet. They optionally provide supplementary information, including:

- an English translation of the sentence;
- notes explaining culturally or contextually specific terms;
- clarification of context-specific profanity or other offensive language;
- the source name or URL.

In addition, each document contains a separate *Lexicon* sheet, used to record detailed explanations of recurring context- or culture-specific terms for which data collectors wish to provide commentary. An example of this sheet is shown in Table 7.5.

Finally, we note that while the initial instructions and examples serve as reference points, data collectors retain relative flexibility to interpret and define hate speech and profanity in ways they consider reflective of the norms and nuances of their specific cultural contexts.

7.3.4 Cross-Annotation

To ensure data quality, cross-annotation is conducted on a subset of data entries from selected datasets. A Python script is used to randomly sample a balanced subset of entries across all six categories, which was then imported into a separate Google Sheets document in the format shown in Table 7.6. The sampled entries are cross-annotated by a second data collector proficient in the language, who independently assigns polarity and profanity labels to each entry. This process aims to ensure the consistency and robustness of the collected data. Details of cross-annotated datasets are presented in Table 7.7.

7.3.5 Inter-Annotator Agreement

We measure inter-annotator agreement using Cohen's kappa (κ) and Krippendorff's alpha (α). These metrics are calculated for two scenarios: (1) three classes, considering all polarity categories (positive, neutral, hateful), and (2) two classes, where positive and neutral categories are merged into a non-hateful class.

Agreement scores for both metrics are shown in Table 7.8 for all cross-annotated datasets. The results indicate moderate to substantial agreement for the Afrikaans datasets, while the agreement scores observed for other datasets are generally higher, showing the robustness of REACT datasets.

(Write in Russian) Provide hateful text about the war victims using profanity.

Text (Original)	Text (English)	Notes	Profane words	Source
Пустили хохлов в страну, сейчас все расстащат нахуй.	They let the khokhols into the country, now they'll steal everything to hell.	Uses "khokhol", a xenophobic slur for Ukrainians.	нахуй	VK
Ебаные укронацисты, сидят там в Европе.	Fucking Ukro-Nazis, sitting there in Europe.	It is common to associate Ukrainians with Nazis.	ебанные	VK
Рагули в Подмосковье получили пизды.	Raguli in the Moscow suburbs got their asses kicked.	"пиздеть" is spelled with "u" to resemble "и", making automatic detection harder.	пизды	News articles comment section

Table 7.4: A visual example of the Google Sheets document used for data collection. The table shows the sheet used for hateful, profane texts about Ukrainian war victims in Russian, as indicated in the instruction at the top, with three data samples for illustration. The header defines the information to be recorded, including the texts in the original language and their English translations. Additional columns may be used to provide supplementary details. In this example, relevant cultural nuances are explained under *Notes*, profane words are listed under *Profane words*, and the data source is documented in the *Source* column.

Word	Pronunciation	(Contextual) Definition
бандерофашисты	banderofashisty	A derogatory term for supporters of Ukraine, combining the name of Stepan Bandera, a Ukrainian nationalist leader, and фашисты (“fascists”).
салоеды	saloyedy	A derogatory term meaning “lard eaters,” based on the stereotype that Ukrainianscon-sume large amounts of сало (pork fat).
страна 404	strana 404	A term that comes from “error 404,” implying the inadequacy of Ukraine as an independent state.
Кукраина	kukraina	A derogatory alteration of “Ukraine” intended to resemble the sound of roosters (“кукареку” - “kukareku”)
укропы	ukropy	An offensive way of calling Ukrainians, derived from укроп (“dill”).
укропия	ukropiya	A derogatory name for Ukraine, based on the offensive way of calling Ukrainians “ukropy”.
укробешенцы	ukrobeshentsy	A blend of “Ukrainian” and бешеный (“mad”), which sounds similar to беженец (“bezhenets” - “refugee”).
Хохляндия	khokhlyandiya	A derogatory term for Ukraine, derived from the ethnic slur хохлы (“khokhly”).

Table 7.5: A few examples of commonly encountered terms from the *Lexicon* sheet of the data collection document for Ukrainian war victims in Russian. Each entry includes the term, its romanized reading, and the contextual definition provided by the data collectors.

Text	Positive/Neutral/Hateful	Profane (Yes/No)
Jou sogenaamde God het gay wesens geskep, nie net gay mense nie. word groot jou godsdienstige moer!	Positive	No
Ek weet dis fokken reg! Liefde is liefde, en die LGBTQ+-gemeenskap omhels die krag van liefde in al sy vorme. Ek weet dis fokken reg!	Positive	Yes
As ’n gay man kan ek myself ’n moffie noem as ek wil maar jy as ’n nie-gay persoon moet dit nie sê nie	Neutral	Yes
Hy het genoem dat hy nooit geboelie is omdat hy openlik gay was op skool nie.	Neutral	No
Fok jou. Ek sal seker maak dat jou fokken moffie gat afsak tot op die diepste fokken bodem wat ek kan kry. Jou naai.	Hateful	Yes

Table 7.6: A visual illustration of the Google Sheets document used for cross-annotation for the LGBTQ+ dataset in Afrikaans, showing a few examples of entries. A second data collector is instructed to independently assign polarity and profanity labels to each entry.

language	target	#sentences
Afrikaans	Black people	92
	LGBTQ	352
Ukrainian	Russians	988
Russian	LGBTQ	115
	War victims	193
Korean	Women	120

Table 7.7: The number of sentences in each cross-annotated dataset.

language	target	3 classes		2 classes	
		κ	α	κ	α
Afrikaans	Black people	0.48	0.65	0.82	0.82
	LGBTQ	0.57	0.71	0.58	0.57
Ukrainian	Russians	0.66	0.73	0.85	0.85
Russian	LGBTQ	0.87	0.92	0.93	0.93
	War victims	0.67	0.77	0.74	0.74
Korean	Women	0.66	0.80	0.60	0.60

Table 7.8: Inter-annotator agreement scores using Cohen's kappa (κ) and Krippendorff's alpha (α) for the cross-annotated datasets. The metrics are calculated for three classes (positive, neutral, hateful) and two classes (non-hateful and hateful).

7.4 Experiments

7.4.1 Preliminaries

We implement federated learning (FL) using Flower[6], a versatile, user-friendly FL framework. FL at scale typically operates with a central server connected with client nodes, which run on user devices. A key advantage of the Flower framework is its ability to simulate an FL environment that enables training without relying on actual user devices, allowing us to create simulated clients on a single machine.

For this study, we focus on hate speech detection for low-resource languages using four selected language-target group combinations for which we have sufficient cross-annotated data. These are: Afrikaans, black people (`afr-black`), Afrikaans, LGBTQ (`afr-lgbtq`), Russian, LGBTQ (`rus-lgbtq`), and Russian, war victims

[6] `https://flower.ai`

(rus-war).

7.4.2 Models

FL is often constrained by the large communication overhead between clients and the server, where even a small amount of transmitted data may burden the network bandwidth (Bonawitz et al., 2019). Additionally, smaller models are typically better suited for FL due to their flexibility to operate on devices with varying computing capacities (Hard et al., 2018), enabling hate speech classification without noticeable delay on both high-end and resource-constrained devices. Due to these factors, we limit our selection to lightweight language models for this study.

We evaluate seven models, four of which are multilingual: multilingual BERT (mBERT) (Devlin et al., 2019), multilingual DistilBERT (Distil-mBERT) (Sanh et al., 2019), multilingual MiniLM (Wang et al., 2020a), and XLM-RoBERTa (XLM-R) (Conneau et al., 2020). The remaining three models do not undergo explicit multilingual pre-training: DistilBERT, ALBERT (Lan et al., 2020), and TinyBERT (Jiao et al., 2020). Below are the models used with the model sizes shown:

- XLM-RoBERTa (279M)[7]
- Multilingual BERT (179M)[8]
- Multilingual DistilBERT (135M)[9]
- DistilBERT (67M)[10]
- Multilingual MiniLM (33M) [11]
- TinyBERT (14.5M)[12]
- ALBERT (11.8M)[13]

All seven models are additionally fine-tuned on English HateCheck (Röttger et al., 2021), a hate speech detection dataset categorized by target groups, prior to applying FL, resulting in 14 model variants in total.

[7] https://huggingface.co/FacebookAI/xlm-roberta-base
[8] https://huggingface.co/google-bert/bert-base-multilingual-cased
[9] https://huggingface.co/distilbert/distilbert-base-multilingual-cased
[10] https://huggingface.co/distilbert/distilbert-base-uncased
[11] https://huggingface.co/microsoft/Multilingual-MiniLM-L12-H384
[12] https://huggingface.co/huawei-noah/TinyBERT_General_4L_312D
[13] https://huggingface.co/albert/albert-base-v2

	afr-black	afr-lgbtq	rus-lgbtq	rus-war
dev	0.5	0.5	0.7	0.5
train	0.5	0.5	0.5	0.6

Table 7.9: Maximum Levenshtein ratio thresholds used for selecting development and training data. Note that the values shown are the upper bounds and the actual thresholds may be lower for development or training data.

	afr-black	afr-lgbtq	rus-lgbtq	rus-war
train	0-15	0-15	0-15	0-15
dev	300	120	120	300
test	87	225	111	154

Table 7.10: Number of sentences in the train, development, and test sets of each target group. We use 0, 3, 9, and 15 sentences per target group for few-shot fine-tuning.

Preliminary results (detailed in Section D) show that models without explicit multilingual pre-training, as well as multilingual MiniLM, perform poorly across all four target groups ($F_1 < 0.50$ in most cases). Among the remaining multilingual models, mBERT and Distil-mBERT achieve comparable and the highest performance (with F_1 scores of 0.70 and 0.72 on the best-performing client models, respectively), and at the same time have more compact sizes relative to XLM-R. Following fine-tuning, performance gaps between the two models narrow further. Based on these results, we choose mBERT and Distil-mBERT for subsequent experiments.

7.4.3 Data Splitting

For each target group, we build a test set using cross-annotated data agreed upon by two native annotators. As our data is highly target-specific and may exhibit similar patterns, we mitigate potential data overlap by setting a maximum Levenshtein ratio threshold to filter sentences when creating the train and development sets. By default, a threshold of 0.50 is used, meaning that sentences with a Levenshtein ratio of < 0.50 relative to any test data are retained for the development set, and those with a Levenshtein ratio of < 0.50 relative to any test and development data are retained as train data. For `rus-lgbtq` and `rus-war`, where data is limited, the threshold is slightly relaxed. We nevertheless perform manual verification of sentences with a Levenshtein ratio > 0.50 to ensure no near-identical sentences are included across splits. Table 7.9 lists the maximum Levenshtein ratio thresholds used for the four datasets. The numbers of sentences in each split for the four target groups are shown in Table 7.10.

7.4.4 Federated Learning

Using Flower's simulation, we create one server and four client instances, each representing a target group. We perform two types of evaluation: zero-shot and few-shot with 3, 9, and 15 training sentences per client. We conduct five rounds of FL with one local epoch per round, meaning each client is trained on its local data for one epoch each round. At the end of the FL process, each client model is evaluated independently on its test set, and the server model is evaluated using the combined test data from all target groups. We report macro-F_1 scores averaged over five random seeds.

7.4.5 Client Personalization

Client customization is crucial to enabling personal hate speech detection to serve the specific needs of target groups. We implement two personalization methods during FL:

FedPer. Introduced by Arivazhagan et al. (2019), this approach keeps the last layers of the client model private, sharing only updates to the base (non-private) layers with the server. Specifically, K_B and K_P refer to the number of base and personalized layers, respectively. Since personalization starts from the last layers, $K_P = 1$ means only the classifier head is personalized, whereas $K_P = n + 1$ means the classifier head plus the last n Transformer layers are personalized.

We test $K_P \in \{1, 2, 3, 4\}$ on mBERT and Distil-mBERT. The server model is excluded from evaluation as part of its parameters, most importantly those of the classifier head, are updated on the client side only.

Adapters. A growing body of research has approached personalized hate speech detection by incorporating annotators' demographics and preferences (Kanclerz et al., 2022; Fleisig et al., 2023; Hoeken et al., 2024) or gaze features of the users (Alacam et al., 2024) into annotations to better understand the subjectivity of hate speech. Inspired by such work, we integrate adapters (Houlsby et al., 2019) between each pair of Transformer layers as customizable client-specific parameters. We evaluate two configurations: (1) full-model fine-tuning (all parameters are updated, but only non-adapter updates are shared with the server) and (2) adapter-only fine-tuning (non-adapter parameters remain frozen). Note that the second configuration does not involve FL, as non-personalized parameters are not updated nor shared with the server model. Evaluation is not performed on the server model as with FedPer.

7.4.6 Baseline

To evaluate the effectiveness of FL, we perform standard few-shot fine-tuning for each model using the data of a single target group with the same hyperparameter setup. For

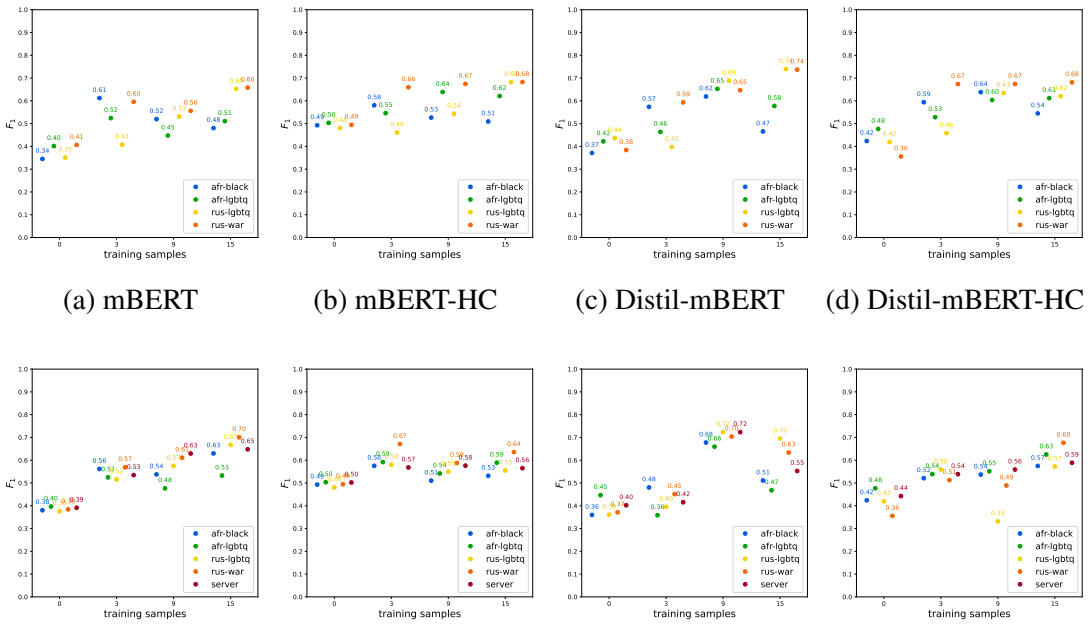
comparability, each model is trained for five epochs, matching the number of FL rounds.

7.4.7 Computation

Simulation for standard FL and FedPer experiments using mBERT and Distil-mBERT with four clients can be completed in 20-30 minutes on four NVIDIA GeForce RTX 2080 Ti GPUs. Adapter-personalized FL experiments with the same setups can be completed in about 30 minutes on four NVIDIA RTX A6000 GPUs. Fine-tuning on English HateCheck is completed on a single NVIDIA GeForce RTX 2080 Ti GPU and takes about 2 minutes for TinyBERT, 5 minutes for Distil-mBERT, DistilBERT, mMiniLM, and ALBERT, and 10 minutes for mBERT and XLM-R.

7.5 Results

We present key findings from our results below.



(a) mBERT (b) mBERT-HC (c) Distil-mBERT (d) Distil-mBERT-HC

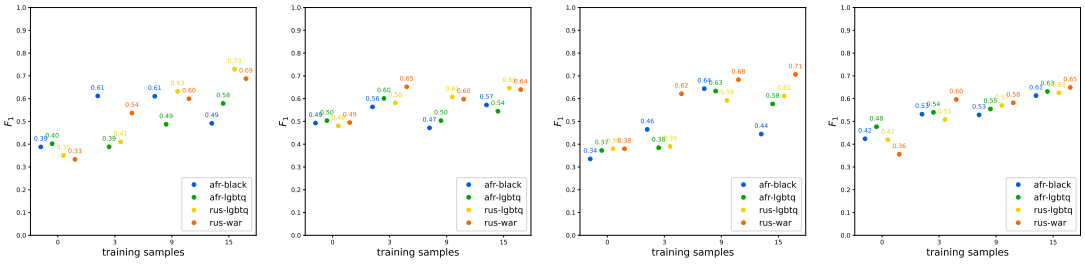
(e) mBERT (f) mBERT-HC (g) Distil-mBERT (h) Distil-mBERT-HC

Figure 7.1: Baseline (top row) and FL (bottom row) performance for all four clients and the server (only plotted for FL) using mBERT and Distil-mBERT. "HC" indicates a model fine-tuned on English HateCheck data prior to FL. Each color represents a client (target group) or the server. FL improves client performance in many cases. English fine-tuning provides greater benefits when using 0-3 training samples per client.

Client performance consistently benefits from FL. Figure 7.1 presents the performance of single-target training (top row) and federated learning (bottom row) using mBERT and Distil-mBERT, along with their fine-tuned versions on English HateCheck data (indicated by "HC"). Each plot shows F_1 scores across increasing numbers of training samples, with colors indicating individual clients or the server.

A comparison between single-target training and FL reveals that clients with initially lower F_1s in the single-target setting generally see performance gains through FL, suggesting benefits from collective training data. Furthermore, with the exception of Distil-mBERT, server performance consistently improves with more training data during FL, indicating its ability to capture hate speech patterns across target groups.

When the models are fine-tuned on English data prior to FL, the improvement varies with the amount of training data. Both models show more positive gains with little training data. As shown in the bottom row of Figure 7.1, FL yields notable improvements of 0.8 (mBERT) and 0.7 (Distil-mBERT) per client on average with 0- or 3-shot settings. However, as the amount of training data increases (9- and 15-shot), English fine-tuning reduces FL performance, resulting in average per-client drops of -0.04 (mBERT) and -0.09 (Distil-mBERT).

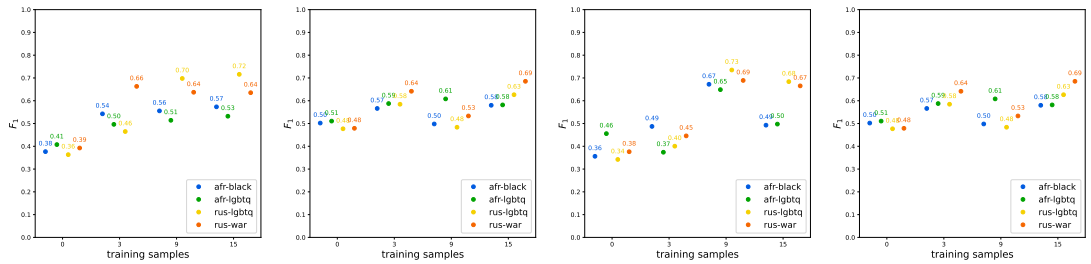


(a) mBERT $K_P = 4$ (b) mBERT-HC $K_P = 2$ (c) Distil-mBERT $K_P = 4$ (d) Distil-mBERT-HC $K_P = 3$

Figure 7.2: F_1 scores of client models personalized using FedPer. Results are shown using the best K_P value for each model. Distil-mBERT-HC demonstrates modest improvements across clients, while the other models show limited performance gains.

Personalization offers limited benefits. The degree of personalization in FedPer is determined by the number of personalized layers (K_P) in each client model. We test $K_P \in \{1, 2, 3, 4\}$ for both mBERT and Distil-mBERT and report results for the best K_Ps in Figure 7.2, while full results for all K_Ps are shown in Section E. For simplicity, the best K_P value is defined as the one yielding the highest average F_1 improvement per client across all training data sizes.

Without prior English fine-tuning, FedPer outcomes are highly variable for both models, with performance improving for some clients but dropping for others. For



(a) mBERT (b) mBERT-HC (c) Distil-mBERT (d) Distil-mBERT-HC

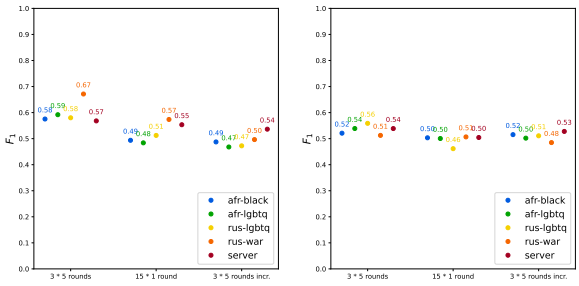
Figure 7.3: F_1 scores of client models using adapter-based personalization and full-model fine-tuning. While some clients show performance gains to varying degrees, the overall improvement remains unclear.

example, when using mBERT with 15 training samples, the `afr-black` client experiences a significant drop of 0.14 in F_1, whereas `rus-lgbtq` sees an increase of 0.06. Distil-mBERT shows similar inconsistencies, with performance dropping for all clients in 3-shot (up to -0.16), but improving in 9-shot (up to 0.18). Both models, when fine-tuned on English data, exhibit more stable client performance and show improved client performance in low-resource scenarios (0- and 3-shot settings). Distil-mBERT-HC demonstrates the most consistent improvements across all clients, despite the gains being modest, except for `rus-lgbtq` (up to 0.24).

For personalization with adapters, full-model fine-tuning consistently outperforms adapter-only fine-tuning. We present FL results with full-model fine-tuning in Figure 7.3 and the full results in Section F. Comparing Figures 7.3 and 7.1 (bottom row), a few clients benefit from adapter-based personalization. For example, `rus-lgbtq` improves by 0.05-0.16 with Distil-mBERT-HC and mBERT in 9- and 15-shot settings, while `rus-war` improves by up to 0.09 with mBERT. However, similar to findings for FedPer, adapter-based personalization does not demonstrate overall consistent improvements across all clients.

Smaller models benefit slightly more from personalization. A comparison of standard FL (Figure 7.1) and personalized FL (Figures 7.2 and 7.3) reveals that the smaller Distil-mBERT model benefits slightly more from FedPer, with an average F_1 improvement of 0.02 per client with the best K_P). In contrast, mBERT experiences only negligible gains. Both models show comparable performance with adapter-based personalization, with no clear improvement across clients.

Effectiveness of incremental training. We investigate whether it is more beneficial to incrementally update training data in each FL round or retain the same training data over all rounds. Three configurations are tested: (1) using the same 3 sentences across



(a) mBERT-HC (b) Distil-mBERT-HC

Figure 7.4: Comparison of F_1 scores of client and server models across three configurations of training data usage.

five FL rounds, (2) using 15 sentences in a single FL round, and (3) using 15 sentences incrementally, with 3 different sentences per round.

Figure 7.4 shows the results, with the x-axis denoting the three configurations. The results show that using the same 3 sentences across five FL rounds archives superior results compared to using 15 sentences in a single FL round. In addition, incremental training with 15 sentences divided across five rounds performs worse than using all 15 sentences in one round.

7.6 Analysis

Impact of English fine-tuning. As shown in Figure 7.1, fine-tuning on English data may negatively affect the performance of many individual clients when larger amounts of target-specific training data are available, such as with 9 or 15 training samples per target group. This effect becomes more obvious under FL. A possible explanation for this drop in performance is the difference between targets used in our datasets and those in the HateCheck dataset used for fine-tuning. While fine-tuning on a general hate speech detection dataset proves beneficial under data-scarce settings, its benefit diminishes as more target-specific data becomes available.

Effectiveness of personalization. Figures 7.2 and 7.3 show that both FedPer and adapter-based personalization have variable effects on client performance and are highly sensitive to the target group. To assess their overall effectiveness, we calculate the average F_1 improvement per client across all four training set sizes. In some cases, FedPer shows improvements, such as for `rus-lgbtq` using Distil-mBERT-HC, but Table 7.11 suggests that, overall, FedPer does not consistently outperform non-personalized FL. Similarly, as indicated in Table 7.12, the overall gains from adapter-based personalization are limited.

In summary, the effects of the personalization methods tested are complex and cannot be entirely dismissed. Personalizing client models, especially with an increasing K_P in FedPer, reduces the number of shared parameters in FL, which may negatively impact the overall performance of all participants. Additionally, the small number of target groups in this study may limit the potential benefits of personalization, which could be more valuable when applied to a wider range of real-world target groups.

	M	M-HC	D	D-HC
$K_P = 1$	-0.05	-0.01	-0.03	0.00
$K_P = 2$	-0.03	0.00	-0.01	0.01
$K_P = 3$	-0.04	-0.01	-0.01	0.03
$K_P = 4$	-0.01	-0.02	0.00	0.02

Table 7.11: Average F_1 improvement per client using FedPer with $K_P \in \{1, 2, 3, 4\}$. M: mBERT, D: Distil-mBERT. HC: model is fine-tuned on English HateCheck data.

	M	M-HC	D	D-HC
adapter-only	-0.13	-0.06	-0.10	-0.06
full-model	0.01	0.00	0.00	0.00

Table 7.12: Average F_1 improvement per client using adapter personalization. M: mBERT, D: Distil-mBERT. HC: model is fine-tuned on English HateCheck data.

7.7 Conclusion

In this work, we make two key contributions. First, we introduce REACT, a collection of localized, context-specific hate speech detection datasets. REACT comprises data in eight low-resource languages, covering seven distinct target groups and curated by data collectors proficient in the target languages and deeply familiar with the cultural and contextual nuances of hate speech in their respective regions. Second, we leverage federated learning (FL), a privacy-preserving machine learning approach that retains user data on local devices, to evaluate two lightweight multilingual language models suitable for deployment on devices with resource constraints for few-shot hate speech classification. Our findings show consistent, albeit modest, improvements on federated client models under zero- and few-shot conditions (Figure 7.1), highlighting the potential of FL as a promising approach for privacy-preserving few-shot learning that could be applied to other tasks.

Additionally, our evaluation of two personalization strategies reveals limited effectiveness in customizing individual clients in low-resource settings, as demonstrated by the lack of overall improvement (Tables 7.11 and 7.12). However, we believe that personalization holds greater potential in resource-rich environments, which we suggest as a direction for future research.

Chapter 8

Conclusion

8.1 Summary

This dissertation explores the challenges of extending NLP technologies to low-resource languages, which have been largely underrepresented due to, among others, data scarcity. Our contributions span multiple dimensions, including dataset creation, efficient cross-lingual transfer methods, novel metrics for conceptual language similarity, and culturally sensitive, privacy-preserving NLP applications. Using hate speech detection as a case study, we demonstrate practical and impactful solutions to promote the inclusiveness and cultural adaptability of NLP systems.

We revisit the research questions proposed in Section 1.2 and summarize the contributions below.

Evaluation of low-resource languages. We develop Taxi1500, a massively multilingual evaluation dataset constructed using the Parallel Bible Corpus (PBC) and covering over 1500 languages. This dataset enables large-scale evaluations of multilingual pretrained language models (mPLMs) across a diverse set of languages, addressing language underrepresentation in existing benchmarks. Evaluations on mPLMs and LLMs with varying language coverages demonstrate the competitiveness of Taxi1500, highlighting the superior few-shot performance of LLMs and the dataset's value as a comprehensive multilingual benchmark.

Quantifying conceptual diversity across languages. By using Conceptualizer, our previously introduced framework that enables the exploration of conceptual relatedness of over 1300 languages, we quantify conceptual language similarity and analyze conceptual overlaps and divergences. The novel metric complements traditional approaches based on genealogical and typological relationships, offering a new lens through which to examine linguistic diversity. Extensive evaluations and comparisons with traditional

similarity metrics demonstrate the ability of conceptual language similarity to capture meaningful cross-lingual conceptual patterns.

Effective cross-lingual transfer for low-resource languages. We facilitate cross-lingual transfer through architectural and resource-efficient innovations. We propose the MoSECroT framework, which leverages static word embeddings and PLMs to enable efficient cross-lingual transfer to low-resource languages without requiring extensive re-training. Additionally, we introduce LANGSAMP, a framework that enhances language neutrality in mPLMs through language- and script-aware embeddings, further improving transferability of knowledge across diverse languages. Both frameworks demonstrate effectiveness to varying degrees across different downstream tasks.

Culturally sensitive and privacy-preserving NLP. We curate and release the REACT dataset, a collection of hate speech detection datasets covering multiple low-resource languages and cultural contexts. REACT is developed in collaboration with data collectors with profound background knowledge of the cultural nuances of the target groups to ensure accurate representation and context-sensitive annotations. Furthermore, to protect user privacy, we employ a federated learning (FL) approach that avoids central data collection while addressing challenges such as target heterogeneity while balancing privacy and classification performance.

8.2 Limitations and Future Directions

While this dissertation presents numerous contributions, we acknowledge several limitations that warrant further investigation and identify possibilities for future research.

One major limitation arises from the reliance on biblical texts as the primary parallel text source for both Taxi1500 and Conceptualizer. While extensive in terms of language coverage, the thematic scope of the Bible is limited in its representation of more generic and modern topics. For many low-resource languages, the availability of parallel texts is restricted to the New Testament, which limits the data size per language in Taxi1500. For Conceptualizer, sparse data can occasionally result in misaligned verses, especially for languages with unique translation styles that blur verse boundaries. However, despite these limitations, large-scale religious texts remain among the most widely available parallel resources, making them invaluable for massively multilingual studies like ours.

Future research could explore expanding multilingual datasets by incorporating texts from diverse domains, including modern parallel corpora such as book translations and government documents. Data augmentation techniques such as back-translation could further enhance the robustness of such datasets. Another interesting direction involves further refining the conceptual language similarity by improving its understanding and measures for evaluation.

Our work on the MoSECroT framework, introduced in Chapter 5, faces challenges when applied to extremely low-resource languages that lack static word embeddings or sufficient resources to create them. Exploring methods for transferring knowledge to such languages remains an important avenue for future research. Although MoSECroT is designed to be architecture-agnostic, further research could examine its performance across diverse model architectures. In addition, both Conceptualizer and MoSECroT could benefit from experimentation with a broader range of source languages, especially those spreading across language families and scripts.

In the FL framework introduced in Chapter 7, we demonstrate the effectiveness of FL for hate speech detection over centralized fine-tuning with the same amount of data. However, the scalability of FL in an environment with heterogeneous devices poses a challenge as resource-constrained devices limit the types of models that can be deployed. It should further be noted that hyperparameter tuning has not been explored exhaustively due to the extent and complexity of the experiments, leaving room for future optimizations. Moreover, while FL enhances privacy by design, it is not immune to adversarial attacks or information leakage in certain scenarios (Hitaj et al., 2017; Zhu et al., 2019; Geiping et al., 2020; Truex et al., 2021). Future work should explore integrating techniques such as differential privacy (DP) (Dwork et al., 2016) and advanced client personalization to further secure user data. We further propose that scaling up FL to handle non-IID data and enhancing robustness against adversarial attacks is also a promising research direction.

8.3 Ethical and Societal Reflections

Recent debates in computational linguistics have raised the question of whether the current expansion of NLP to low-resource languages truly serves the communities it intends to support. Proponents of critical and contextualized NLP, notably Steven Bird, have advocated the re-centering of NLP on communities rather than treating languages as mere data objects. Bird and others argue that linguistic data, when collected in the form of monologue recordings or corpora, risk being "divorced from" social functions in which the languages are embedded (Good, 2018). Such concerns raise fundamental questions, such as to what extent NLP benefits the affected communities, when it risks causing harm, and how future research should balance scientific goals with ethical responsibilities.

Linguistics- and community-centric revitalization

A central critique of extending NLP thoughtlessly to under-resourced languages, particularly oral vernaculars without standardized writing systems, is that fully automatic computational methods, when applied without attention to the linguistic and social contexts, may do more harm than good. Instead, cooperation between community authorities,

linguistic experts, and NLP researchers is pivotal. Language technologies should be developed in tandem with broader frameworks for sustaining long-term language use, such as the *FAMED conditions* (Lewis and Simons, 2016). These frameworks stress not only the maintenance of the spoken language itself but also the revitalization of identity and social functions, which in turn promote language use.

This implies that NLP efforts should not end with corpus creation, as is common in many current pipelines, but should contribute to strengthening language practices and identities. Following this view, it is essential to resist the neocolonialist tendency of treating under-resourced and indigenous languages as commodities whose raw data can be harvested and processed by outside researchers (Mager et al., 2023; Roberts and Montoya, 2023). Instead, technological approaches should serve as auxiliary tools, integrated within community-driven frameworks where linguistic and cultural expertise guides the development and application.

Ethical concerns and balancing of interests

Bird (Bird, 2020) and other indigenous experts make a valid point in characterizing mainstream approaches to low-resource NLP as a neocolonialist move, in that they extract data without recognizing the expertise of affected communities. To counter this, the naive top-down approach of data collection and resource creation should be matched with close cooperation with local experts to ensure the process is aligned with community goals rather than imposed externally.

At the same time, these critiques do not imply computational methods should be abandoned altogether in favor of traditional, and potentially inefficient, methods for resource construction. A community-centered approach can complement NLP by ensuring that resources are built responsibly and contribute to cultural preservation. Ultimately, this is a question of balance: scientific reasons for developing NLP in low-resource contexts must be weighed against ethical considerations. Concerns about a *flattening* of local cultures into Western-defined frameworks (Srinivasan, 2017) are valid, but can be mitigated through collaboration with local experts. A guiding principle here should be that local communities retain agency in the extent and manner of digitization, with external NLP researchers in auxiliary roles, which in fact exemplifies the ideal of self-determination in language revitalization (Stebbins et al., 2017).

Personal reflections on language documentation

Experts like Bird argue that the standardization required for documenting many non-institutional, predominantly oral vernaculars risks misconstruing characteristics of these languages (Bird, 2024). However, language standardization is often an inevitable step in linguistic evolution and does not necessarily constitute a Western-centric imposition.

While it often comes at the cost of diminishing dialectal diversity, it also facilitates preservation, education, and wider communicative use.

As a native speaker of Wu, a non-standardized, predominantly oral Sinitic language, I view its preservation as important for functions beyond local knowledge transfer. Like other non-standardized Sinitic languages, Wu has played a role in regional communication, including trade and education, which contrasts with the conception that oral languages primarily serve ceremonial or local roles (Bird, 2024). Moreover, it is precisely due to the lack of standardization that the numerous Wu dialects are confined within the borders of each local community while their shared linguistic heritage remains overlooked despite significant structural and lexical similarities. In practice, this often leads speakers of different Wu dialects to default to Mandarin as a vehicular language. Documentation and standardization, therefore, may help mitigate language shift, supporting continued use of Wu alongside Mandarin as the regional vehicular language through phenomena such as *multilingual receptive comprehension*, or the ability to understand related languages without active proficiency (Davies, 1976; Meakins, 2013). It should be noted, however, that such initiatives should remain under the agency of local communities.

For unwritten languages, which tend to be more severely under-resourced than their counterparts with writing systems, transcription is a practical necessity for documentation, as text-based NLP remains more advanced than speech-only approaches. While recent developments such as Meta's SeamlessM4T (Communication et al., 2023) demonstrate the possibility of speech-to-speech translation, their reliance on large-scale self-supervised approaches with limited expert input risks perpetuating inaccuracies and misrepresentations of non-dominant languages (Marten and Petzell, 2016).

Finally, while concerns about embedding local knowledge into Western-centric frameworks are valid, they can be mitigated if affected communities lead the process. In this light, my position is not to cede agency to NLP technologies, but to view them as auxiliary tools that, if developed responsibly, can support the documentation and preservation of languages and the cultural identities they embody.

8.4 Final Remarks

This dissertation advances the state of multilingual NLP by addressing the critical needs of low-resource languages and marginalized linguistic communities. We have proposed solutions that account for linguistic diversity, cultural nuances, and privacy concerns. The datasets introduced, such as Taxi1500 and REACT, represent efforts to democratize access to NLP technologies and provide support for underrepresented languages around the globe.

By addressing the limitations and proposing future research directions, we encourage further exploration into expanding the inclusiveness of NLP systems. Our ultimate

goal is to make these systems accessible to a broader global audience, ensuring that no community is left behind due to linguistic underrepresentation. As language technologies continue to evolve, the works presented here lay a strong foundation for advancements in the field.

Finally, this dissertation has also reflected on the ethical and societal dimensions of low-resource NLP beyond its technical contributions. These reflections highlight that inclusiveness requires not only technical progress but also responsibility toward the communities most affected.

Appendix

A Taxi1500 Zero-shot Evaluation Results

The complete zero-shot evaluation results of mBERT, XLM-R Base, XLM-R Large, and Glot500 on Taxi1500 are presented in Tables 1 to 11.

B Evaluation Results of LANGSAMP and Baseline

The complete results of zero-shot cross-lingual transfer for LANGSAMP and baseline are presented in Tables 12, 13 (SR-B), Table 14 (SR-T), Tables 15, 16(Taxi1500), 17 (SIB200), Table 18 (NER), and Table 19 (POS). Each score represents the average over five random seeds.

C Evaluation Results Using English and Best Donor

The complete results of zero-shot cross-lingual transfer for LANGSAMP using English and the closest donor language as the source language are presented in Tables 20, 21 (Taxi1500), 22 (SIB200), Table 23 (NER), and Table 24 (POS). Each score represents the result from a single run, where LANGSAMP using different source languages is fine-tuned using the same random seed.

D Preliminary Results Using REACT Datasets

Preliminary evaluation results of the seven models (see Section 7.4.2) on federated hate speech detection. Among these, four models are multilingual, while the remaining models are not explicitly pre-trained on multilingual data. Results are shown in Figures 1 (non-fine-tuned models) and 2 (fine-tuned on English HateCheck).

E Full Results using FedPer

Full evaluation results of FedPer-personalized mBERT and Distil-mBERT, along with their variants fine-tuned on English HateCheck data, are shown in Figures 3 to 6. K_P (number of personalized layers) values $\in \{1, 2, 3, 4\}$ are tested.

F Full Results using Adapter-based Personalization

Full evaluation results of adapter-personalized mBERT and Distil-mBERT, along with their variants fine-tuned on English HateCheck data, are shown in Figure 7. Evaluation is performed both for full-model fine-tuning and adapter-only fine-tuning.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
aah_Latn	**0.13**	0.10	0.05	0.05	0.08	aoz_Latn	**0.21**	0.13	0.07	0.05	0.07
aai_Latn	**0.22**	0.15	0.09	0.05	0.09	apb_Latn	0.07	0.08	0.06	0.05	**0.12**
aak_Latn	0.07	**0.13**	0.05	0.05	0.05	ape_Latn	0.13	**0.13**	0.05	0.05	0.07
aau_Latn	0.12	**0.12**	0.06	0.05	0.10	apn_Latn	0.07	**0.19**	0.06	0.05	0.05
aaz_Latn	0.07	**0.12**	0.05	0.05	0.08	apr_Latn	0.07	**0.07**	0.07	0.05	0.05
abi_Latn	0.07	**0.11**	0.05	0.05	0.05	apt_Latn	0.08	**0.14**	0.07	0.05	0.07
abt_Latn	0.09	**0.13**	0.08	0.05	0.06	apu_Latn	0.07	0.09	**0.10**	0.05	0.05
abx_Latn	0.16	0.12	0.20	0.14	**0.33**	apw_Latn	**0.15**	0.10	0.05	0.05	0.05
aby_Latn	**0.21**	0.12	0.07	0.07	0.06	apy_Latn	0.09	0.09	**0.11**	0.05	0.05
acd_Latn	**0.13**	0.08	0.05	0.05	0.05	apz_Latn	0.07	**0.11**	0.05	0.05	0.05
ace_Latn	0.13	0.25	0.11	0.11	**0.30**	are_Latn	0.11	**0.12**	0.05	0.05	0.05
acf_Latn	0.09	0.25	0.06	0.05	**0.38**	arl_Latn	**0.15**	0.14	0.05	0.05	0.05
ach_Latn	**0.13**	0.12	0.05	0.05	0.08	arn_Latn	**0.13**	0.08	0.05	0.05	0.08
acn_Latn	0.07	**0.10**	0.05	0.05	0.05	ary_Arab	0.07	**0.28**	0.19	0.27	0.19
acr_Latn	0.16	0.14	0.06	0.05	**0.30**	arz_Arab	0.07	0.43	0.32	**0.47**	0.25
acu_Latn	0.10	**0.10**	0.05	0.05	0.08	asg_Latn	0.08	**0.11**	0.05	0.05	0.06
ade_Latn	**0.12**	0.10	0.07	0.05	0.06	asm_Beng	0.07	0.17	0.43	0.47	**0.51**
adh_Latn	0.13	**0.15**	0.07	0.05	0.07	aso_Latn	**0.15**	0.12	0.05	0.05	0.05
adi_Latn	0.09	0.10	**0.14**	0.05	0.09	ata_Latn	0.11	**0.12**	0.06	0.05	0.06
adj_Latn	**0.17**	0.08	0.05	0.05	0.05	atb_Latn	**0.10**	0.09	0.07	0.05	0.06
adl_Latn	0.08	**0.18**	0.05	0.05	0.05	atd_Latn	**0.11**	0.09	0.05	0.05	0.05
aeb_Arab	0.07	0.38	0.19	**0.42**	0.30	atg_Latn	0.10	**0.11**	0.07	0.05	0.07
aer_Latn	0.07	**0.08**	0.08	0.05	0.05	atq_Latn	0.13	**0.15**	0.06	0.05	0.13
aeu_Latn	0.07	**0.13**	0.05	0.05	0.05	att_Latn	0.14	0.10	0.08	0.05	**0.16**
aey_Latn	0.07	**0.12**	0.09	0.05	0.05	auc_Latn	0.09	**0.13**	0.06	0.05	0.05
afr_Latn	0.33	0.45	0.59	**0.66**	0.52	auy_Latn	0.07	**0.07**	0.04	0.05	0.06
agd_Latn	0.09	**0.16**	0.06	0.08	0.07	ava_Cyrl	0.07	0.06	0.05	0.05	**0.10**
agg_Latn	**0.14**	0.06	0.05	0.05	0.05	avn_Latn	**0.14**	0.12	0.05	0.05	0.05
agm_Latn	0.07	**0.11**	0.06	0.05	0.05	avt_Latn	0.10	0.11	0.05	0.05	**0.14**
agn_Latn	0.12	0.16	0.13	0.18	**0.35**	avu_Latn	**0.07**	0.06	0.04	0.05	0.05
agr_Latn	0.07	**0.11**	0.05	0.05	0.05	awa_Deva	0.07	0.24	0.37	0.40	**0.48**
agt_Latn	0.07	**0.10**	0.06	0.05	0.10	awb_Latn	0.08	**0.11**	0.06	0.05	0.05
agu_Latn	**0.11**	0.09	0.04	0.05	0.06	awi_Latn	**0.17**	0.12	0.04	0.05	0.14
agw_Latn	0.20	0.13	0.11	0.07	**0.24**	ayo_Latn	0.12	**0.12**	0.10	0.05	0.08
ahk_Latn	0.08	**0.11**	0.07	0.05	0.07	ayp_Arab	0.07	0.30	0.29	0.35	**0.43**
aia_Latn	**0.23**	0.13	0.05	0.05	0.08	ayr_Latn	0.07	**0.12**	0.11	0.06	0.10
aii_Syrc	0.07	0.05	0.05	0.09	**0.10**	azb_Arab	0.07	0.16	0.15	0.08	**0.34**
aim_Latn	0.10	**0.14**	0.06	0.05	0.05	aze_Latn	0.07	0.32	0.56	**0.68**	0.59
ain_Latn	**0.11**	0.09	0.07	0.05	0.10	azg_Latn	0.04	**0.09**	0.05	0.05	0.05
aji_Latn	0.13	**0.14**	0.05	0.05	0.05	azz_Latn	0.14	**0.15**	0.06	0.06	0.10
ajz_Latn	0.12	**0.12**	0.05	0.05	0.07	bak_Cyrl	0.07	**0.33**	0.13	0.05	0.24
aka_Latn	0.12	**0.17**	0.10	0.06	0.13	bam_Latn	0.09	0.11	0.06	0.05	**0.20**
akb_Latn	0.13	0.16	0.15	0.07	**0.27**	ban_Latn	0.07	0.16	0.16	0.09	**0.31**
ake_Latn	**0.11**	0.08	0.05	0.05	0.05	bao_Latn	0.10	**0.14**	0.08	0.05	0.06
akh_Latn	0.10	**0.15**	0.05	0.05	0.05	bar_Latn	0.13	0.19	0.30	0.29	**0.41**
akp_Latn	0.10	**0.16**	0.06	0.05	0.05	bav_Latn	**0.12**	0.05	0.05	0.05	0.06
ald_Latn	**0.08**	0.05	0.05	0.05	0.05	bba_Latn	**0.13**	0.12	0.05	0.05	0.05
alj_Latn	0.11	0.14	0.10	0.10	**0.21**	bbb_Latn	0.07	**0.09**	0.05	0.05	0.05
aln_Latn	0.07	0.25	0.46	0.53	**0.55**	bbj_Latn	**0.12**	0.05	0.05	0.05	0.05
alp_Latn	0.10	0.19	0.13	0.06	**0.20**	bbk_Latn	**0.09**	0.04	0.05	0.05	0.05
alq_Latn	0.09	**0.11**	0.05	0.05	0.05	bbo_Latn	0.10	**0.12**	0.07	0.05	0.06
als_Latn	0.07	0.24	0.45	**0.54**	0.49	bbr_Latn	**0.17**	0.15	0.04	0.05	0.06
alt_Cyrl	0.07	0.16	0.17	0.19	**0.37**	bch_Latn	0.10	**0.13**	0.07	0.05	0.12
alz_Latn	0.10	0.15	0.06	0.05	**0.17**	bci_Latn	0.09	0.12	0.04	0.05	**0.15**
ame_Latn	0.09	**0.11**	0.09	0.05	0.05	bcl_Latn	0.07	0.18	0.26	0.20	**0.46**
amf_Latn	0.07	**0.08**	0.05	0.05	0.05	bcw_Latn	**0.12**	0.05	0.06	0.05	0.05
amh_Ethi	0.07	0.05	**0.10**	0.05	0.07	bdd_Latn	**0.11**	0.07	0.05	0.05	0.05
amk_Latn	0.13	**0.19**	0.06	0.05	0.07	bdh_Latn	0.07	**0.10**	0.05	0.05	0.05
amm_Latn	**0.09**	0.07	0.04	0.05	0.08	bdq_Latn	0.10	**0.12**	0.05	0.05	0.05
amn_Latn	0.11	0.11	0.07	0.05	**0.12**	bef_Latn	0.10	**0.10**	0.07	0.05	0.07
amp_Latn	0.07	**0.12**	0.06	0.05	0.05	bel_Cyrl	0.07	0.43	0.59	**0.67**	0.59
amr_Latn	0.09	**0.12**	0.05	0.05	0.05	bem_Latn	0.14	0.11	0.08	0.09	**0.31**
amu_Latn	0.06	**0.08**	0.05	0.05	0.05	ben_Beng	0.07	0.32	0.56	**0.67**	0.63
anm_Latn	0.13	**0.14**	0.06	0.05	0.05	beq_Latn	0.14	**0.14**	0.09	0.05	0.10
ann_Latn	0.14	**0.15**	0.08	0.05	0.06	bex_Latn	**0.13**	0.10	0.05	0.05	0.08
anv_Latn	0.13	**0.13**	0.05	0.05	0.08	bfd_Latn	**0.11**	0.09	0.05	0.05	0.05
any_Latn	0.07	**0.07**	0.05	0.05	0.05	bfo_Latn	0.10	**0.11**	0.05	0.05	0.06
aoj_Latn	**0.20**	0.09	0.08	0.05	0.06	bgr_Latn	0.16	0.17	0.07	0.05	**0.30**
aom_Latn	**0.23**	0.16	0.05	0.05	0.05	bgs_Latn	**0.15**	0.14	0.09	0.07	0.11
aon_Latn	0.08	**0.11**	0.06	0.05	0.05	bgt_Latn	0.15	**0.16**	0.07	0.05	0.16

Table 1: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
bgz_Latn	0.09	**0.18**	0.09	0.06	0.15	bzj_Latn	0.24	0.15	0.13	0.06	**0.35**
bhl_Latn	0.10	**0.12**	0.06	0.05	0.07	caa_Latn	0.14	**0.15**	0.07	0.05	0.12
bhp_Latn	0.09	0.11	**0.16**	0.06	0.09	cab_Latn	0.07	**0.10**	0.05	0.05	0.05
bhw_Latn	0.09	**0.16**	0.07	0.05	0.14	cac_Latn	0.12	0.12	0.06	0.05	**0.21**
bhz_Latn	**0.18**	0.14	0.06	0.05	0.06	caf_Latn	**0.09**	0.07	0.05	0.05	0.05
bib_Latn	**0.16**	0.06	0.05	0.05	0.06	cag_Latn	0.07	**0.14**	0.05	0.05	0.11
big_Latn	0.09	**0.10**	0.05	0.05	0.05	cak_Latn	0.04	0.12	0.05	0.05	**0.42**
bim_Latn	**0.14**	0.13	0.05	0.05	0.06	cao_Latn	0.08	**0.10**	0.05	0.05	0.10
bis_Latn	0.16	0.22	0.14	0.06	**0.24**	cap_Latn	**0.11**	0.09	0.05	0.05	0.05
biu_Latn	0.16	0.14	0.05	0.05	**0.17**	caq_Latn	0.10	**0.10**	0.04	0.05	0.10
biv_Latn	**0.11**	0.07	0.05	0.05	0.05	car_Latn	**0.13**	0.12	0.06	0.05	0.06
bjr_Latn	0.07	**0.10**	0.05	0.05	0.05	cas_Latn	**0.15**	0.09	0.08	0.05	0.04
bjv_Latn	**0.11**	0.08	0.06	0.05	0.05	cat_Latn	0.13	0.41	0.58	**0.64**	0.47
bkd_Latn	0.07	**0.21**	0.15	0.08	0.21	cav_Latn	0.07	**0.11**	0.06	0.05	0.05
bkl_Latn	**0.15**	0.11	0.06	0.07	0.05	cax_Latn	0.07	**0.12**	0.09	0.05	0.06
bkq_Latn	**0.14**	0.12	0.06	0.05	0.11	cbc_Latn	0.08	**0.14**	0.06	0.05	0.05
bku_Latn	0.15	0.11	0.08	0.06	**0.19**	cbi_Latn	**0.14**	0.13	0.09	0.05	0.11
bkv_Latn	**0.13**	0.06	0.06	0.05	0.09	cbk_Latn	0.11	0.39	0.45	0.48	**0.57**
blh_Latn	0.05	**0.07**	0.05	0.05	0.05	cbr_Latn	0.13	**0.15**	0.05	0.05	0.05
blt_Latn	**0.11**	0.08	0.07	0.05	0.06	cbs_Latn	0.05	**0.15**	0.05	0.05	0.06
blw_Latn	0.07	**0.15**	0.06	0.05	0.10	cbt_Latn	0.08	**0.09**	0.06	0.05	0.06
blz_Latn	0.15	**0.19**	0.09	0.06	0.12	cbu_Latn	0.07	**0.12**	0.05	0.05	0.05
bmb_Latn	0.14	**0.14**	0.09	0.05	0.10	cbv_Latn	0.09	**0.15**	0.06	0.05	0.08
bmh_Latn	0.07	**0.11**	0.08	0.05	0.08	cce_Latn	0.09	0.10	0.09	0.05	**0.21**
bmq_Latn	**0.10**	0.07	0.05	0.05	0.05	cco_Latn	**0.10**	0.06	0.05	0.05	0.05
bmr_Latn	0.07	**0.13**	0.05	0.05	0.05	ccp_Latn	0.11	**0.19**	0.09	0.06	0.09
bmu_Latn	0.09	**0.14**	0.05	0.05	0.05	cdf_Latn	0.09	**0.12**	0.05	0.05	0.09
bmv_Latn	**0.16**	0.10	0.07	0.05	0.05	ceb_Latn	0.11	0.12	0.28	0.28	**0.37**
bnj_Latn	0.09	**0.13**	0.07	0.06	0.05	ceg_Latn	0.15	**0.15**	0.04	0.05	0.08
bno_Latn	0.10	0.18	0.18	0.11	**0.33**	cek_Latn	0.09	**0.10**	0.05	0.05	0.06
bnp_Latn	0.11	0.13	0.05	0.06	**0.16**	ces_Latn	0.07	0.28	**0.66**	0.57	0.51
boa_Latn	0.09	**0.16**	0.05	0.05	0.05	cfm_Latn	0.14	0.15	0.05	0.05	**0.25**
boj_Latn	**0.13**	0.10	0.05	0.05	0.07	cgc_Latn	0.07	0.18	0.19	0.14	**0.26**
bom_Latn	0.08	**0.11**	0.05	0.05	0.08	cha_Latn	0.12	0.12	0.11	0.05	**0.19**
bon_Latn	0.11	**0.19**	0.07	0.06	0.05	chd_Latn	0.09	**0.10**	0.05	0.05	0.06
bov_Latn	0.07	**0.12**	0.05	0.05	0.06	che_Cyrl	0.07	**0.10**	0.07	0.05	0.08
box_Latn	0.09	**0.11**	0.05	0.05	0.09	chf_Latn	0.09	0.10	0.12	0.05	**0.21**
bpr_Latn	0.13	**0.13**	0.09	0.05	0.09	chj_Latn	**0.10**	0.06	0.05	0.05	0.05
bps_Latn	**0.16**	0.11	0.08	0.05	0.08	chk_Hani	0.07	**0.13**	0.07	0.05	0.08
bqc_Latn	0.07	**0.11**	0.05	0.05	0.06	chq_Latn	0.09	**0.10**	0.05	0.05	0.05
bqj_Latn	**0.17**	0.12	0.09	0.05	0.07	chr_Cher	0.07	0.05	**0.09**	0.05	0.05
bqp_Latn	0.09	**0.17**	0.05	0.05	0.06	chu_Cyrl	0.07	0.31	0.60	**0.61**	0.46
bre_Latn	0.08	0.29	0.25	**0.43**	0.29	chv_Cyrl	0.07	0.18	0.07	0.05	**0.19**
bru_Latn	0.10	**0.10**	0.07	0.05	0.05	chz_Latn	0.07	**0.08**	0.05	0.05	0.05
bsc_Latn	**0.15**	0.08	0.09	0.05	0.05	cjo_Latn	0.07	**0.07**	0.04	0.05	0.05
bsn_Latn	**0.16**	0.07	0.04	0.05	0.07	cjp_Latn	**0.14**	0.11	0.07	0.05	0.05
bss_Latn	0.07	**0.13**	0.10	0.05	0.05	cjv_Latn	0.06	**0.08**	0.07	0.05	0.05
btd_Latn	0.09	**0.30**	0.21	0.17	0.28	ckb_Latn	0.16	0.09	0.07	0.07	**0.43**
bth_Latn	0.10	0.14	0.12	0.07	**0.25**	cko_Latn	0.08	**0.09**	0.06	0.05	0.06
bto_Latn	0.07	0.11	0.13	0.05	**0.32**	cle_Latn	**0.11**	0.04	0.05	0.05	0.06
btt_Latn	0.12	**0.14**	0.07	0.05	0.06	clu_Latn	0.11	0.14	0.18	0.21	**0.43**
btx_Latn	0.16	0.23	0.20	0.19	**0.34**	cly_Latn	**0.15**	0.12	0.11	0.05	0.06
bud_Latn	0.05	**0.12**	0.05	0.05	0.05	cme_Latn	0.09	**0.12**	0.05	0.05	0.05
bug_Latn	0.09	**0.19**	0.12	0.07	0.17	cmn_Hani	0.07	0.40	0.59	0.62	**0.65**
buk_Latn	0.07	**0.11**	0.05	0.05	0.08	cmo_Latn	**0.18**	0.17	0.13	0.05	0.05
bul_Cyrl	0.07	0.41	0.62	**0.64**	0.60	cmr_Latn	0.11	**0.13**	0.05	0.05	0.06
bum_Latn	0.09	0.16	0.06	0.05	**0.17**	cnh_Latn	0.18	0.12	0.08	0.05	**0.20**
bus_Latn	0.08	**0.13**	0.05	0.05	0.05	cni_Latn	0.07	**0.07**	0.05	0.05	0.05
bvc_Latn	0.14	**0.21**	0.06	0.05	0.08	cnk_Latn	0.09	**0.09**	0.05	0.05	0.06
bvd_Latn	**0.19**	0.11	0.06	0.05	0.08	cnl_Latn	0.07	**0.07**	0.05	0.05	0.05
bvr_Latn	**0.12**	0.07	0.09	0.05	0.05	cnt_Latn	0.07	**0.08**	0.05	0.05	0.05
bvz_Latn	**0.13**	0.10	0.08	0.05	0.05	cnw_Latn	0.12	0.13	0.06	0.05	**0.14**
bwq_Latn	**0.15**	0.09	0.06	0.05	0.11	coe_Latn	0.07	**0.08**	0.05	0.05	0.06
bwu_Latn	0.14	**0.16**	0.08	0.05	0.09	cof_Latn	0.11	**0.15**	0.06	0.05	0.08
bxr_Cyrl	0.07	0.09	0.25	0.27	**0.31**	cok_Latn	**0.13**	0.08	0.05	0.05	0.07
byr_Latn	0.07	**0.08**	0.05	0.05	0.06	con_Latn	**0.28**	0.07	0.10	0.05	0.07
byx_Latn	0.07	**0.13**	0.07	0.06	0.05	cop_Copt	0.07	**0.07**	0.05	0.05	0.05
bzd_Latn	0.07	**0.10**	0.05	0.05	0.04	cor_Latn	0.09	**0.12**	0.09	0.05	0.11
bzh_Latn	**0.15**	0.08	0.05	0.05	0.05	cot_Latn	0.07	**0.12**	0.05	0.05	0.05
bzi_Thai	0.07	**0.07**	0.07	0.05	0.05	cou_Latn	0.10	**0.14**	0.06	0.05	0.05

Table 2: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
cpa_Latn	0.07	**0.11**	0.05	0.05	0.05	due_Latn	0.10	0.12	0.16	0.05	**0.20**
cpb_Latn	0.07	**0.08**	0.08	0.05	0.05	dug_Latn	0.08	**0.17**	0.17	0.11	0.16
cpc_Latn	0.09	**0.12**	0.06	0.05	0.05	duo_Latn	0.14	0.08	0.16	0.06	**0.31**
cpu_Latn	0.09	**0.11**	0.04	0.07	0.05	dur_Latn	0.10	**0.10**	0.05	0.05	0.05
cpy_Latn	0.07	**0.08**	0.05	0.05	0.05	dwr_Latn	**0.15**	0.11	0.06	0.05	0.10
crh_Cyrl	0.07	0.19	0.15	0.20	**0.45**	dww_Latn	0.07	0.07	**0.08**	0.05	0.06
crj_Latn	**0.15**	0.10	0.05	0.05	0.05	dyi_Latn	**0.16**	0.13	0.07	0.05	0.06
crk_Cans	**0.07**	0.05	0.05	0.05	0.05	dyo_Latn	0.08	**0.12**	0.07	0.05	0.08
crl_Cans	0.07	**0.09**	0.05	0.05	0.05	dyu_Latn	0.07	0.09	0.05	0.05	**0.17**
crm_Cans	**0.07**	0.05	0.05	0.05	0.06	dzo_Tibt	0.07	0.04	0.05	0.08	**0.09**
crn_Latn	**0.10**	0.09	0.05	0.05	0.06	ebk_Latn	0.14	0.15	0.05	0.05	**0.17**
crq_Latn	0.09	**0.16**	0.06	0.05	0.05	efi_Latn	0.13	**0.13**	0.07	0.05	0.11
crs_Latn	0.10	0.17	0.15	0.05	**0.43**	eka_Latn	0.11	**0.17**	0.09	0.06	0.06
crt_Latn	0.10	**0.16**	0.06	0.05	0.05	ell_Grek	0.07	0.31	0.43	**0.60**	0.50
crx_Latn	**0.09**	0.08	0.08	0.05	0.05	emi_Latn	0.09	**0.16**	0.05	0.10	0.09
csk_Latn	0.12	**0.14**	0.09	0.05	0.05	emp_Latn	**0.14**	0.10	0.06	0.05	0.05
cso_Latn	0.07	**0.08**	0.05	0.05	0.05	enb_Latn	0.07	**0.10**	0.05	0.05	0.05
csy_Latn	0.10	0.11	0.08	0.05	**0.14**	eng_Latn	0.43	0.57	**0.65**	0.56	0.63
cta_Latn	0.07	**0.13**	0.05	0.05	0.07	enl_Latn	0.09	**0.10**	0.05	0.05	0.07
ctd_Latn	0.11	0.14	0.07	0.05	**0.22**	enm_Latn	0.33	0.46	0.55	0.45	**0.55**
ctp_Latn	**0.14**	0.08	0.06	0.05	0.06	enq_Latn	0.07	**0.12**	0.05	0.05	0.07
ctu_Latn	0.10	0.09	0.11	0.06	**0.27**	epo_Latn	0.15	0.25	0.57	**0.61**	0.48
cub_Latn	**0.11**	0.08	0.05	0.05	0.05	eri_Latn	0.13	**0.13**	0.07	0.06	0.06
cuc_Latn	0.07	**0.13**	0.05	0.05	0.05	ese_Latn	0.09	**0.13**	0.06	0.05	0.06
cui_Latn	0.08	**0.14**	0.05	0.05	0.05	esi_Latn	**0.21**	0.12	0.05	0.05	0.07
cuk_Latn	**0.16**	0.11	0.13	0.05	0.07	esk_Latn	0.07	**0.11**	0.05	0.05	0.05
cul_Latn	0.09	**0.12**	0.07	0.05	0.05	ess_Latn	**0.14**	0.13	0.06	0.05	0.05
cut_Latn	**0.11**	0.10	0.05	0.05	0.07	est_Latn	0.07	0.46	**0.68**	0.56	0.47
cux_Latn	**0.16**	0.14	0.05	0.06	0.08	esu_Latn	**0.16**	0.12	0.05	0.05	0.05
cwe_Latn	0.11	0.19	0.13	0.11	**0.22**	etu_Latn	**0.13**	0.11	0.05	0.05	0.05
cwt_Latn	0.09	**0.14**	0.05	0.05	0.05	eus_Latn	0.09	0.18	**0.26**	0.25	0.23
cya_Latn	0.12	0.11	**0.14**	0.05	0.11	ewe_Latn	0.11	**0.11**	0.05	0.05	0.07
cym_Latn	0.08	0.23	0.44	**0.53**	0.49	ewo_Latn	0.13	**0.18**	0.08	0.06	0.10
czt_Latn	**0.14**	0.11	0.07	0.05	0.05	eza_Latn	0.07	**0.09**	0.05	0.05	0.06
daa_Latn	**0.13**	0.09	0.06	0.06	0.05	faa_Latn	**0.11**	0.08	0.07	0.05	0.08
dad_Latn	**0.20**	0.15	0.06	0.05	0.05	fai_Latn	**0.13**	0.11	0.06	0.05	0.05
dah_Latn	0.12	**0.17**	0.05	0.05	0.05	fal_Latn	**0.20**	0.15	0.09	0.05	0.06
dan_Latn	0.19	0.52	**0.54**	0.54	0.53	fao_Latn	0.09	0.27	0.32	0.36	**0.48**
dbq_Latn	**0.13**	0.07	0.06	0.05	0.05	far_Latn	0.20	**0.20**	0.07	0.06	0.14
ddn_Latn	**0.10**	0.05	0.10	0.05	0.05	fas_Arab	0.07	0.46	0.67	0.66	**0.67**
ded_Latn	0.07	**0.09**	0.06	0.05	0.06	ffm_Latn	**0.13**	0.11	0.05	0.05	0.07
des_Latn	0.07	**0.10**	0.05	0.05	0.05	fij_Latn	0.05	**0.12**	0.08	0.05	0.12
deu_Latn	0.15	0.38	**0.52**	0.52	0.46	fil_Latn	0.13	0.29	0.47	0.55	**0.55**
dga_Latn	0.10	**0.13**	0.05	0.05	0.05	fin_Latn	0.13	0.45	**0.58**	0.57	0.47
dgc_Latn	0.16	0.14	0.21	0.18	**0.25**	fon_Latn	**0.10**	0.09	0.05	0.05	0.05
dgi_Latn	**0.12**	0.07	0.05	0.05	0.06	for_Latn	0.09	**0.12**	0.07	0.05	0.06
dgr_Latn	0.10	**0.11**	0.05	0.05	0.05	fra_Latn	0.13	0.54	**0.65**	0.65	0.54
dgz_Latn	**0.20**	0.13	0.12	0.06	0.15	frd_Latn	0.08	**0.13**	0.06	0.05	0.09
dhm_Latn	0.17	**0.17**	0.10	0.05	0.10	fry_Latn	0.21	0.38	0.30	0.37	**0.42**
did_Latn	0.07	**0.14**	0.05	0.05	0.05	fub_Latn	**0.17**	0.16	0.10	0.05	0.12
dig_Latn	0.12	0.14	0.20	0.23	**0.39**	fue_Latn	0.13	**0.14**	0.07	0.05	0.14
dik_Latn	**0.12**	0.09	0.08	0.05	0.06	fuf_Latn	0.10	0.10	0.09	0.05	**0.13**
dip_Latn	0.15	**0.15**	0.05	0.05	0.06	fuh_Latn	**0.12**	0.09	0.05	0.06	0.05
dis_Latn	**0.13**	0.11	0.10	0.05	0.06	fuq_Latn	0.11	**0.11**	0.10	0.05	0.10
dje_Latn	**0.12**	0.09	0.08	0.05	0.07	fuv_Latn	0.11	0.13	0.11	0.05	**0.14**
djk_Latn	0.14	0.14	0.08	0.05	**0.28**	gaa_Latn	0.12	**0.13**	0.05	0.05	0.05
djr_Latn	0.07	**0.12**	0.05	0.05	0.05	gag_Latn	0.07	0.13	0.33	0.38	**0.40**
dks_Latn	**0.14**	0.12	0.05	0.05	0.05	gah_Latn	0.07	**0.15**	0.05	0.05	0.05
dln_Latn	0.12	0.12	0.05	0.05	**0.29**	gai_Latn	0.07	**0.09**	0.05	0.05	0.05
dnj_Latn	**0.10**	0.06	0.05	0.05	0.05	gam_Latn	**0.20**	0.11	0.11	0.05	0.11
dnw_Latn	**0.18**	0.12	0.07	0.05	0.06	gaw_Latn	**0.11**	0.09	0.06	0.05	0.08
dob_Latn	0.08	0.08	**0.10**	0.05	0.07	gbi_Latn	0.10	**0.11**	0.06	0.05	0.08
dop_Latn	**0.12**	0.07	0.05	0.05	0.05	gbo_Latn	0.08	**0.14**	0.05	0.05	0.05
dos_Latn	0.13	**0.14**	0.05	0.05	0.05	gbr_Latn	**0.17**	0.08	0.10	0.05	0.09
dow_Latn	0.06	**0.07**	0.05	0.05	0.05	gde_Latn	**0.10**	0.05	0.06	0.05	0.05
dru_Latn	0.07	**0.14**	0.09	0.05	0.09	gdg_Latn	0.10	**0.18**	0.09	0.06	0.16
dsh_Latn	**0.12**	0.10	0.07	0.05	0.06	gdn_Latn	0.07	**0.16**	0.07	0.06	0.09
dtb_Latn	0.11	**0.13**	0.06	0.05	0.08	gdr_Latn	**0.17**	0.09	0.05	0.05	0.06
dtp_Latn	0.12	0.12	0.05	0.05	**0.24**	geb_Latn	0.07	**0.08**	0.05	0.05	0.05
dts_Latn	0.09	**0.09**	0.05	0.05	0.06	gej_Latn	0.09	**0.10**	0.05	0.05	0.08

Table 3: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
gfk_Latn	**0.17**	0.12	0.07	0.05	0.10	hlt_Latn	0.09	**0.09**	0.05	0.05	0.06
ghe_Deva	0.07	0.11	0.20	0.15	**0.28**	hmo_Latn	0.09	**0.14**	0.09	0.05	0.07
ghs_Latn	0.07	**0.10**	0.05	0.05	0.06	hmr_Latn	**0.21**	0.06	0.07	0.05	0.20
gid_Latn	**0.10**	0.05	0.05	0.05	0.08	hne_Deva	0.07	0.27	0.29	0.39	**0.60**
gil_Latn	0.07	0.08	0.04	0.05	**0.23**	hnj_Latn	0.06	**0.06**	0.06	0.05	0.05
giz_Latn	0.07	**0.14**	0.06	0.05	0.07	hnn_Latn	0.11	0.17	0.17	0.12	**0.31**
gjn_Latn	0.09	**0.13**	0.05	0.05	0.05	hns_Latn	0.13	0.12	0.14	0.12	**0.19**
gkn_Latn	0.09	**0.16**	0.05	0.05	0.14	hop_Latn	**0.19**	0.17	0.05	0.05	0.11
gkp_Latn	0.09	**0.12**	0.05	0.05	0.07	hot_Latn	**0.11**	0.10	0.05	0.05	0.06
gla_Latn	0.12	0.14	0.34	0.42	**0.48**	hra_Latn	0.13	0.13	0.07	0.05	**0.26**
gle_Latn	0.17	0.15	0.38	**0.56**	0.40	hrv_Latn	0.09	0.35	0.64	**0.66**	0.63
glv_Latn	0.11	0.10	0.09	0.05	**0.11**	hto_Latn	**0.07**	0.06	0.05	0.06	0.05
gmv_Latn	**0.15**	0.12	0.07	0.06	0.06	hub_Latn	0.07	**0.13**	0.06	0.05	0.06
gna_Latn	0.11	**0.13**	0.05	0.05	0.05	hui_Latn	0.06	**0.10**	0.07	0.05	0.06
gnb_Latn	0.13	0.11	0.06	0.05	**0.20**	hun_Latn	0.08	0.38	**0.70**	0.66	0.52
gnd_Latn	**0.09**	0.06	0.05	0.05	0.05	hus_Latn	0.18	0.17	0.10	0.06	**0.20**
gng_Latn	0.12	**0.13**	0.06	0.05	0.05	huu_Latn	0.07	**0.11**	0.06	0.05	0.06
gnn_Latn	0.07	**0.10**	0.05	0.05	0.08	huv_Latn	0.07	**0.13**	0.06	0.05	0.11
gnw_Latn	0.07	**0.11**	0.07	0.05	0.06	hvn_Latn	0.14	**0.17**	0.09	0.05	0.11
gof_Latn	**0.15**	0.09	0.06	0.05	0.09	hwc_Latn	0.32	0.32	0.40	**0.53**	0.42
gog_Latn	0.13	0.13	0.11	0.07	**0.19**	hye_Armn	0.07	0.39	0.60	0.64	**0.65**
gom_Latn	0.07	0.11	0.06	0.05	**0.19**	ian_Latn	0.07	**0.12**	0.05	0.05	0.09
gor_Latn	0.12	0.17	0.08	0.09	**0.25**	iba_Latn	0.11	0.27	0.26	0.24	**0.54**
gqr_Latn	**0.19**	0.08	0.05	0.05	0.05	ibo_Latn	0.08	**0.12**	0.08	0.05	0.09
grt_Beng	0.07	0.10	**0.16**	0.05	0.11	icr_Latn	0.24	0.21	0.23	0.06	**0.40**
gso_Latn	0.07	**0.09**	0.05	0.05	0.05	ifa_Latn	0.10	0.15	0.06	0.05	**0.32**
gub_Latn	**0.13**	0.11	0.08	0.05	0.05	ifb_Latn	0.16	0.09	0.07	0.05	**0.32**
guc_Latn	0.13	**0.14**	0.05	0.05	0.05	ife_Latn	0.08	**0.11**	0.05	0.05	0.05
gud_Latn	0.11	**0.11**	0.05	0.05	0.05	ifk_Latn	0.14	0.14	0.07	0.05	**0.21**
gug_Latn	0.12	**0.17**	0.09	0.05	0.10	ifu_Latn	0.08	**0.17**	0.05	0.05	0.08
guh_Latn	0.07	**0.08**	0.06	0.05	0.06	ify_Latn	0.09	**0.14**	0.08	0.05	0.11
gui_Latn	0.09	**0.09**	0.09	0.05	0.07	ign_Latn	0.07	**0.09**	0.05	0.05	0.07
guj_Gujr	0.07	0.34	0.56	**0.70**	0.69	ike_Cans	0.07	0.05	0.05	0.05	**0.08**
guk_Ethi	0.07	0.10	0.07	0.05	**0.13**	ikk_Latn	0.07	**0.11**	0.11	0.05	0.05
gul_Latn	0.32	0.26	0.26	0.24	**0.49**	ikw_Latn	0.07	**0.07**	0.06	0.05	0.05
gum_Latn	0.07	**0.09**	0.05	0.05	0.06	ilb_Latn	0.09	0.12	0.14	0.09	**0.16**
gun_Latn	**0.12**	0.11	0.11	0.05	0.06	ilo_Latn	0.14	0.11	0.10	0.05	**0.33**
guo_Latn	0.13	0.09	0.08	0.06	**0.15**	imo_Latn	**0.14**	0.13	0.05	0.05	0.05
guq_Latn	0.07	0.15	**0.16**	0.05	0.06	inb_Latn	**0.11**	0.08	0.06	0.05	0.06
gur_Latn	0.13	**0.15**	0.05	0.05	0.09	ind_Latn	0.07	0.47	0.66	**0.70**	0.63
guu_Latn	**0.11**	0.10	0.06	0.05	0.06	ino_Latn	**0.14**	0.13	0.05	0.05	0.06
guw_Latn	**0.15**	0.12	0.11	0.05	0.05	iou_Latn	**0.14**	0.12	0.05	0.05	0.06
gux_Latn	0.07	**0.10**	0.07	0.05	0.07	ipi_Latn	0.07	**0.14**	0.04	0.05	0.05
guz_Latn	0.07	**0.15**	0.08	0.05	0.06	iqw_Latn	0.07	**0.12**	0.08	0.05	0.06
gvc_Latn	**0.14**	0.08	0.05	0.05	0.06	iri_Latn	0.12	**0.14**	0.05	0.05	0.05
gvf_Latn	**0.18**	0.09	0.06	0.05	0.06	irk_Latn	0.14	**0.15**	0.04	0.05	0.06
gvl_Latn	0.11	**0.14**	0.04	0.05	0.07	iry_Latn	0.08	0.14	0.11	0.16	**0.20**
gvn_Latn	0.07	**0.12**	0.05	0.05	0.09	isd_Latn	0.13	0.15	0.12	0.06	**0.19**
gwi_Latn	**0.19**	0.11	0.05	0.05	0.05	isl_Latn	0.07	0.33	0.57	**0.59**	0.47
gwr_Latn	**0.11**	0.10	0.08	0.05	0.09	ita_Latn	0.14	0.46	0.67	**0.68**	0.55
gya_Latn	0.10	**0.10**	0.05	0.05	0.06	itv_Latn	0.14	0.14	0.15	0.07	**0.27**
gym_Latn	0.11	0.09	**0.12**	0.05	0.07	ium_Latn	**0.10**	0.08	0.05	0.05	0.05
gyr_Latn	0.08	**0.10**	0.07	0.05	0.05	ivb_Latn	0.08	0.12	0.07	0.07	**0.17**
hae_Latn	0.09	0.15	0.15	**0.31**	0.22	ivv_Latn	0.11	0.13	0.07	0.05	**0.19**
hag_Latn	0.10	**0.13**	0.06	0.05	0.06	iws_Latn	**0.10**	0.09	0.05	0.05	0.05
hak_Latn	**0.13**	0.08	0.07	0.05	0.05	ixl_Latn	0.12	0.08	0.06	0.06	**0.16**
hat_Latn	0.06	0.17	0.08	0.06	**0.39**	izr_Latn	0.08	**0.14**	0.05	0.05	0.08
hau_Latn	0.14	0.15	0.36	**0.49**	0.40	izz_Latn	0.07	**0.13**	0.07	0.05	0.05
haw_Latn	0.12	0.11	0.05	0.05	**0.19**	jaa_Latn	0.10	**0.12**	0.06	0.05	0.08
hay_Latn	0.09	0.14	0.06	0.05	**0.15**	jac_Latn	**0.13**	0.07	0.06	0.05	0.09
hch_Latn	0.08	**0.13**	0.06	0.05	0.08	jae_Latn	0.07	**0.07**	0.05	0.05	0.05
heb_Hebr	0.07	**0.36**	0.15	0.31	0.24	jam_Latn	0.22	0.15	0.10	0.06	**0.46**
heg_Latn	0.07	**0.16**	0.05	0.05	0.09	jav_Latn	0.07	0.25	0.38	**0.57**	0.46
heh_Latn	0.10	**0.15**	0.11	0.09	0.09	jbu_Latn	0.12	**0.12**	0.08	0.05	0.08
hif_Latn	0.09	0.12	0.16	0.35	**0.43**	jic_Latn	0.13	**0.24**	0.07	0.05	0.12
hig_Latn	**0.15**	0.07	0.09	0.05	0.05	jiv_Latn	0.09	**0.15**	0.04	0.05	0.05
hil_Latn	0.14	0.23	0.26	0.24	**0.53**	jmc_Latn	**0.15**	0.10	0.05	0.06	0.09
hin_Deva	0.07	0.40	0.56	**0.62**	0.61	jpn_Jpan	0.07	0.37	**0.62**	0.56	0.50
hix_Latn	0.07	**0.08**	0.06	0.05	0.05	jra_Latn	0.09	**0.12**	0.06	0.05	0.06
hla_Latn	0.14	**0.15**	0.06	0.05	0.07	jun_Orya	0.07	0.05	0.11	0.06	**0.12**

Table 4: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
jvn_Latn	0.07	0.35	0.36	**0.52**	0.49	knf_Latn	0.13	**0.15**	0.07	0.05	0.05
kaa_Cyrl	0.07	0.17	0.14	0.16	**0.52**	kng_Latn	0.07	0.14	0.08	0.05	**0.15**
kab_Latn	0.11	**0.14**	0.07	0.06	0.13	knj_Latn	0.07	0.09	0.05	0.05	**0.18**
kac_Latn	**0.13**	0.10	0.05	0.05	0.05	knk_Latn	0.06	**0.11**	0.05	0.05	0.08
kal_Latn	0.09	0.11	0.05	0.05	**0.13**	kno_Latn	0.10	**0.10**	0.05	0.05	0.07
kan_Knda	0.07	0.34	0.56	**0.64**	0.61	knv_Latn	**0.18**	0.12	0.05	0.05	0.08
kao_Latn	0.09	**0.09**	0.05	0.05	0.06	kog_Latn	0.11	**0.12**	0.06	0.05	0.05
kaq_Latn	0.09	**0.16**	0.06	0.05	0.09	kor_Hang	0.07	0.43	0.63	**0.69**	0.62
kat_Geor	0.07	0.46	0.48	**0.61**	0.54	kpf_Latn	0.07	**0.10**	0.05	0.05	0.05
kaz_Cyrl	0.07	0.32	0.57	**0.66**	0.57	kpg_Latn	**0.22**	0.15	0.05	0.05	0.15
kbc_Latn	**0.18**	0.07	0.05	0.05	0.05	kpj_Latn	0.07	**0.10**	0.04	0.05	0.07
kbh_Latn	0.09	**0.13**	0.07	0.05	0.07	kpq_Latn	**0.15**	0.14	0.04	0.05	0.06
kbm_Latn	0.09	**0.15**	0.11	0.06	0.07	kpr_Latn	**0.13**	0.10	0.10	0.05	0.08
kbo_Latn	0.11	**0.15**	0.04	0.05	0.06	kpv_Cyrl	0.07	0.09	0.09	0.05	**0.11**
kbp_Latn	**0.10**	0.08	0.05	0.05	0.05	kpw_Latn	**0.14**	0.10	0.05	0.05	0.05
kbq_Latn	**0.12**	0.05	0.09	0.05	0.05	kpx_Latn	0.07	**0.13**	0.09	0.05	0.05
kbr_Latn	0.08	**0.13**	0.05	0.05	0.07	kpz_Latn	0.09	**0.12**	0.05	0.05	0.09
kcg_Latn	**0.13**	0.12	0.05	0.05	0.05	kqc_Latn	0.08	0.09	**0.11**	0.05	0.08
kck_Latn	0.08	0.13	0.09	0.05	**0.18**	kqe_Latn	0.13	0.16	0.13	0.12	**0.33**
kdc_Latn	0.13	0.14	0.20	0.19	**0.21**	kqo_Latn	0.07	**0.09**	0.05	0.05	0.05
kde_Latn	0.14	**0.16**	0.12	0.07	0.15	kqp_Latn	0.14	**0.14**	0.05	0.05	0.06
kdi_Latn	0.07	**0.16**	0.05	0.05	0.08	kqs_Latn	0.10	**0.13**	0.05	0.05	0.06
kdj_Latn	0.07	**0.13**	0.05	0.05	0.05	kqy_Ethi	0.07	**0.13**	0.06	0.05	0.05
kdl_Latn	0.07	**0.11**	0.07	0.05	0.09	krc_Cyrl	0.07	0.17	0.17	0.16	**0.48**
kdp_Latn	0.10	**0.11**	0.10	0.05	0.07	kri_Latn	0.15	0.16	0.05	0.05	**0.19**
kek_Latn	0.15	0.08	0.05	0.06	**0.27**	krj_Latn	0.11	0.21	0.33	0.28	**0.35**
ken_Latn	**0.10**	0.08	0.05	0.05	0.05	krl_Latn	0.07	0.34	0.40	0.40	**0.41**
keo_Latn	0.11	0.08	0.06	0.05	**0.11**	kru_Deva	0.07	**0.12**	0.08	0.05	0.11
ker_Latn	**0.09**	0.04	0.05	0.05	0.05	ksb_Latn	0.12	0.16	0.12	0.12	**0.21**
kew_Latn	0.13	**0.14**	0.05	0.05	0.06	ksc_Latn	0.09	**0.12**	0.07	0.05	0.11
kez_Latn	**0.13**	0.10	0.05	0.05	0.05	ksd_Latn	**0.15**	0.14	0.06	0.05	0.12
kff_Telu	0.07	0.14	**0.24**	0.20	0.20	ksf_Latn	**0.10**	0.07	0.05	0.05	0.06
kgf_Latn	0.08	**0.10**	0.05	0.05	0.05	ksr_Latn	0.08	**0.08**	0.05	0.05	0.06
kgk_Latn	0.07	**0.10**	0.06	0.05	0.05	kss_Latn	**0.12**	0.10	0.05	0.05	0.05
kgp_Latn	0.07	**0.14**	0.09	0.05	0.09	ksw_Mymr	0.07	**0.08**	0.05	0.05	0.06
kgr_Latn	0.14	**0.20**	0.06	0.05	0.13	ktb_Ethi	0.07	0.05	0.07	0.05	**0.10**
kha_Latn	**0.12**	0.07	0.07	0.05	0.06	ktj_Latn	0.04	**0.05**	0.05	0.05	0.05
khk_Latn	0.09	**0.15**	0.07	0.05	0.08	kto_Latn	0.07	**0.14**	0.09	0.05	0.05
khm_Khmr	0.07	0.05	0.55	**0.62**	0.55	ktu_Latn	0.10	0.11	0.11	0.06	**0.19**
khq_Latn	**0.12**	0.11	0.10	0.05	0.09	kua_Latn	0.11	0.11	0.11	0.08	**0.12**
khs_Latn	**0.14**	0.09	0.06	0.05	0.05	kub_Latn	0.09	**0.14**	0.05	0.05	0.05
khy_Latn	0.08	0.09	0.07	0.07	**0.14**	kud_Latn	0.07	**0.10**	0.06	0.05	0.05
khz_Latn	0.12	**0.16**	0.06	0.05	0.05	kue_Latn	0.07	**0.11**	0.06	0.05	0.07
kia_Latn	0.13	0.19	0.06	0.05	**0.23**	kuj_Latn	0.12	**0.12**	0.05	0.05	0.05
kij_Latn	0.07	**0.14**	0.07	0.05	0.06	kum_Cyrl	0.07	0.16	0.13	0.24	**0.45**
kik_Latn	0.14	**0.15**	0.05	0.05	0.05	kup_Latn	**0.18**	0.15	0.08	0.05	0.07
kin_Latn	0.14	0.13	0.14	0.06	**0.23**	kus_Latn	**0.12**	0.09	0.10	0.05	0.05
kir_Cyrl	0.07	0.20	**0.65**	0.65	0.61	kvg_Latn	**0.11**	0.09	0.06	0.05	0.06
kix_Latn	0.08	**0.12**	0.07	0.05	0.05	kvj_Latn	**0.17**	0.13	0.06	0.05	0.05
kjb_Latn	0.15	0.11	0.05	0.05	**0.23**	kvn_Latn	**0.12**	0.09	0.08	0.05	0.06
kje_Latn	0.09	**0.18**	0.06	0.05	0.06	kwd_Latn	**0.19**	0.13	0.09	0.05	0.12
kjh_Cyrl	0.07	0.18	0.11	0.17	**0.36**	kwf_Latn	**0.21**	0.17	0.09	0.07	0.16
kjs_Latn	**0.13**	0.10	0.07	0.05	0.05	kwi_Latn	0.11	**0.17**	0.09	0.05	0.09
kki_Latn	0.16	**0.17**	0.14	0.10	0.14	kwj_Latn	0.10	**0.12**	0.06	0.05	0.05
kkj_Latn	0.09	**0.16**	0.06	0.05	0.06	kxc_Ethi	0.07	**0.09**	0.07	0.05	0.05
kle_Deva	0.07	0.14	0.15	0.11	**0.19**	kxm_Thai	0.07	0.08	**0.14**	0.06	0.08
kln_Latn	0.10	0.10	0.05	0.05	**0.12**	kxw_Latn	0.06	**0.07**	0.06	0.05	0.05
klv_Latn	0.09	**0.14**	0.13	0.05	0.09	kyc_Latn	0.07	**0.11**	0.06	0.05	0.06
kma_Latn	**0.12**	0.08	0.05	0.05	0.05	kyf_Latn	0.09	**0.13**	0.05	0.05	0.05
kmd_Latn	0.10	**0.11**	0.06	0.05	0.09	kyg_Latn	0.08	**0.09**	0.06	0.05	0.05
kmg_Latn	0.08	**0.08**	0.05	0.05	0.05	kyq_Latn	0.10	**0.12**	0.07	0.05	0.05
kmh_Latn	0.07	**0.10**	0.05	0.05	0.05	kyu_Mymr	0.07	**0.09**	0.05	0.05	0.05
kmk_Latn	0.10	0.10	0.06	0.05	**0.14**	kyz_Latn	**0.17**	0.10	0.05	0.05	0.05
kmm_Latn	0.12	0.09	0.05	0.05	**0.19**	kze_Latn	0.08	**0.11**	0.04	0.05	0.06
kmo_Latn	**0.10**	0.09	0.05	0.06	0.06	kzf_Latn	0.12	**0.18**	0.10	0.06	0.15
kmr_Cyrl	0.07	0.09	0.07	0.05	**0.24**	lac_Latn	**0.16**	0.05	0.06	0.05	0.11
kms_Latn	**0.13**	0.08	0.04	0.05	0.07	lai_Latn	0.16	0.13	0.07	0.08	**0.19**
kmu_Latn	0.07	**0.17**	0.10	0.05	0.08	laj_Latn	0.10	**0.11**	0.07	0.06	0.09
kmy_Latn	**0.12**	0.08	0.05	0.05	0.05	lam_Latn	0.09	0.14	0.07	0.07	**0.16**
kne_Latn	**0.15**	0.13	0.12	0.04	0.09	lao_Laoo	0.07	0.05	0.58	**0.67**	0.61

Table 5: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
lap_Latn	0.14	**0.15**	0.06	0.05	0.08	mbb_Latn	0.11	**0.20**	0.10	0.05	0.10
las_Latn	0.09	**0.09**	0.05	0.05	0.05	mbc_Latn	0.12	**0.13**	0.05	0.05	0.05
lat_Latn	0.14	0.30	0.55	**0.62**	0.56	mbd_Latn	**0.13**	0.12	0.11	0.05	0.10
lav_Latn	0.08	0.34	**0.62**	0.55	0.52	mbf_Latn	0.07	0.31	0.49	**0.57**	0.56
law_Latn	0.09	**0.09**	0.06	0.05	0.09	mbh_Latn	0.15	**0.15**	0.07	0.05	0.09
lbk_Latn	0.12	0.10	0.09	0.05	**0.14**	mbi_Latn	0.13	**0.17**	0.08	0.05	0.06
lcm_Latn	0.16	**0.20**	0.05	0.06	0.15	mbj_Latn	**0.16**	0.14	0.08	0.05	0.06
lcp_Thai	0.07	**0.08**	0.06	0.05	0.05	mbl_Latn	0.07	**0.11**	0.05	0.05	0.05
ldi_Latn	0.14	0.12	0.07	0.05	**0.19**	mbs_Latn	0.11	0.12	0.17	0.13	**0.19**
lee_Latn	**0.08**	0.05	0.07	0.05	0.05	mbt_Latn	**0.14**	0.12	0.07	0.05	0.09
lef_Latn	0.05	**0.13**	0.06	0.05	0.05	mca_Latn	**0.16**	0.10	0.05	0.05	0.06
leh_Latn	0.09	0.14	0.08	0.07	**0.15**	mcb_Latn	0.07	**0.11**	0.05	0.05	0.06
lem_Latn	0.07	**0.09**	0.05	0.05	0.06	mcd_Latn	0.05	**0.09**	0.05	0.05	0.06
leu_Latn	0.12	**0.14**	0.05	0.05	0.07	mcf_Latn	0.07	**0.10**	0.06	0.05	0.05
lew_Latn	0.07	0.13	0.08	0.05	**0.16**	mck_Latn	0.13	**0.15**	0.11	0.06	0.15
lex_Latn	**0.13**	0.10	0.08	0.05	0.05	mcn_Latn	0.09	**0.10**	0.07	0.06	0.10
lgg_Latn	0.09	**0.19**	0.05	0.05	0.13	mco_Latn	0.05	0.09	0.05	0.05	**0.13**
lgl_Latn	**0.20**	0.14	0.06	0.06	0.12	mcp_Latn	**0.09**	0.05	0.05	0.05	0.05
lgm_Latn	**0.12**	0.11	0.06	0.06	0.09	mcq_Latn	0.07	**0.12**	0.08	0.05	0.05
lhi_Latn	0.09	**0.12**	0.05	0.05	0.10	mcu_Latn	0.10	**0.20**	0.07	0.05	0.06
lhm_Latn	**0.12**	0.08	0.05	0.05	0.05	mda_Latn	0.06	**0.07**	0.05	0.05	0.05
lhu_Latn	**0.09**	0.08	0.06	0.05	0.06	mdy_Ethi	0.07	0.09	0.05	0.05	**0.15**
lia_Latn	**0.18**	0.16	0.05	0.05	0.05	med_Latn	0.07	**0.09**	0.06	0.05	0.07
lid_Latn	**0.16**	0.09	0.08	0.05	0.06	mee_Latn	0.11	**0.12**	0.05	0.05	0.06
lif_Deva	0.07	0.07	0.10	0.05	**0.13**	mej_Latn	0.07	**0.11**	0.09	0.05	0.08
lin_Latn	0.12	0.10	0.08	0.04	**0.13**	mek_Latn	0.08	0.10	0.08	0.05	**0.14**
lip_Latn	0.08	**0.12**	0.06	0.05	0.07	men_Latn	0.11	**0.13**	0.05	0.05	0.05
lis_Lisu	0.07	**0.08**	0.05	0.05	0.06	meq_Latn	**0.10**	0.07	0.07	0.05	0.05
lit_Latn	0.07	0.29	0.56	**0.60**	0.54	met_Latn	**0.19**	0.11	0.05	0.05	0.06
ljp_Latn	0.07	0.29	0.33	0.30	**0.39**	meu_Latn	0.10	**0.14**	0.10	0.05	0.08
llg_Latn	0.07	0.09	**0.13**	0.05	0.07	mfe_Latn	0.09	0.15	0.15	0.05	**0.36**
lln_Latn	**0.10**	0.09	0.05	0.05	0.05	mfh_Latn	0.07	**0.07**	0.06	0.05	0.07
lmk_Latn	**0.14**	0.11	0.07	0.05	0.05	mfi_Latn	**0.15**	0.07	0.06	0.05	0.06
lmp_Latn	0.09	**0.12**	0.05	0.05	0.05	mfk_Latn	0.09	**0.16**	0.05	0.05	0.05
lnd_Latn	0.09	0.13	0.10	0.06	**0.15**	mfq_Latn	**0.08**	0.05	0.05	0.05	0.06
lob_Latn	0.07	**0.10**	0.05	0.05	0.04	mfy_Latn	0.11	**0.15**	0.07	0.05	0.06
loe_Latn	0.10	0.21	0.10	0.08	**0.23**	mfz_Latn	**0.13**	0.09	0.05	0.05	0.05
log_Latn	0.11	**0.11**	0.05	0.05	0.05	mgh_Latn	**0.13**	0.10	0.04	0.05	0.08
lok_Latn	**0.13**	0.12	0.05	0.05	0.05	mgo_Latn	**0.15**	0.05	0.05	0.05	0.05
lol_Latn	0.07	**0.09**	0.06	0.05	0.09	mgr_Latn	0.17	0.13	0.10	0.07	**0.21**
lom_Latn	**0.11**	0.07	0.05	0.05	0.05	mhi_Latn	0.12	**0.12**	0.08	0.05	0.06
loq_Latn	0.08	**0.13**	0.05	0.05	0.06	mhl_Latn	0.10	**0.10**	0.05	0.05	0.05
loz_Latn	0.18	0.14	0.06	0.05	**0.29**	mhr_Cyrl	0.07	0.17	0.10	0.05	**0.26**
lsi_Latn	**0.13**	0.08	0.05	0.05	0.05	mhx_Latn	0.11	**0.12**	0.05	0.05	0.05
lsm_Latn	0.11	**0.16**	0.08	0.07	0.08	mhy_Latn	0.12	0.20	0.21	0.15	**0.26**
ltz_Latn	0.15	0.34	0.22	0.20	**0.41**	mib_Latn	0.09	**0.13**	0.07	0.06	0.13
luc_Latn	0.07	0.09	**0.11**	0.05	0.05	mic_Latn	0.10	**0.13**	0.08	0.05	0.06
lug_Latn	0.07	0.13	0.08	0.05	**0.22**	mie_Latn	0.08	**0.17**	0.06	0.05	0.12
luo_Latn	0.12	0.12	0.05	0.05	**0.15**	mif_Latn	0.09	**0.09**	0.07	0.05	0.07
lus_Latn	**0.17**	0.14	0.10	0.05	0.09	mig_Latn	0.13	**0.19**	0.05	0.05	0.07
lwo_Latn	0.12	**0.12**	0.05	0.05	0.05	mih_Latn	0.08	**0.13**	0.04	0.05	0.07
lww_Latn	0.11	**0.12**	0.06	0.05	0.05	mil_Latn	0.10	**0.11**	0.05	0.05	0.06
lzh_Hani	0.07	0.24	0.54	0.50	**0.59**	mim_Latn	0.11	**0.15**	0.05	0.05	0.06
maa_Latn	0.13	**0.14**	0.05	0.05	0.05	min_Latn	0.08	0.19	0.27	0.26	**0.43**
mad_Latn	0.10	0.22	0.23	0.19	**0.40**	mio_Latn	0.09	0.08	**0.15**	0.07	0.14
maf_Latn	0.11	**0.18**	0.06	0.05	0.05	mip_Latn	0.06	0.10	0.05	0.05	**0.11**
mag_Deva	0.07	0.22	0.38	0.32	**0.49**	miq_Latn	0.09	**0.16**	0.05	0.05	0.08
mah_Latn	**0.16**	0.12	0.05	0.05	0.14	mir_Latn	0.06	0.09	0.06	0.05	**0.14**
mai_Deva	0.07	0.23	0.31	0.43	**0.65**	mit_Latn	0.06	0.09	0.07	0.06	**0.12**
maj_Latn	0.09	**0.09**	0.05	0.05	0.05	miy_Latn	0.07	**0.10**	0.05	0.05	0.08
mak_Latn	0.10	**0.18**	0.10	0.06	0.18	miz_Latn	0.09	**0.14**	0.05	0.05	0.05
mal_Mlym	0.07	**0.12**	0.07	0.05	0.06	mjc_Latn	0.13	**0.13**	0.05	0.05	0.07
mam_Latn	0.12	0.11	0.04	0.04	**0.25**	mjw_Latn	0.08	**0.09**	0.08	0.05	0.05
maq_Latn	0.12	**0.15**	0.05	0.06	0.05	mkd_Cyrl	0.07	0.47	**0.74**	0.70	0.67
mar_Deva	0.07	0.30	0.57	**0.61**	0.59	mkl_Latn	**0.11**	0.05	0.06	0.05	0.05
mas_Latn	0.07	**0.17**	0.09	0.06	0.04	mkn_Latn	0.07	0.23	0.28	0.35	**0.44**
mau_Latn	0.07	**0.08**	0.05	0.05	0.05	mks_Latn	0.10	**0.15**	0.05	0.05	0.05
mav_Latn	**0.14**	0.12	0.07	0.05	0.05	mlg_Latn	0.12	0.08	0.37	0.45	**0.46**
maw_Latn	**0.18**	0.11	0.05	0.05	0.05	mlh_Latn	0.10	**0.10**	0.05	0.05	0.05
maz_Latn	0.10	**0.15**	0.05	0.05	0.10	mlp_Latn	0.07	**0.20**	0.06	0.05	0.08

Table 6: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
mlt_Latn	0.11	0.16	0.05	0.06	**0.29**	mzm_Latn	0.09	**0.09**	0.05	0.05	0.05
mmn_Latn	0.17	0.19	0.18	0.21	**0.32**	mzw_Latn	0.05	**0.09**	0.05	0.05	0.06
mmo_Latn	**0.17**	0.09	0.09	0.05	0.05	nab_Latn	0.07	**0.14**	0.05	0.05	0.05
mmx_Latn	**0.14**	0.11	0.05	0.05	0.06	naf_Latn	0.07	**0.15**	0.05	0.05	0.06
mna_Latn	**0.11**	0.08	0.05	0.05	0.05	nak_Latn	0.11	**0.12**	0.04	0.05	0.08
mnb_Latn	0.10	**0.17**	0.06	0.05	0.16	nan_Latn	**0.14**	0.11	0.05	0.05	0.06
mnf_Latn	0.11	**0.13**	0.05	0.05	0.06	naq_Latn	0.09	**0.10**	0.05	0.05	0.07
mnh_Latn	0.07	**0.17**	0.07	0.05	0.09	nas_Latn	0.07	0.09	**0.11**	0.05	0.09
mnk_Latn	0.09	**0.17**	0.05	0.05	0.07	nav_Latn	**0.19**	0.09	0.05	0.05	0.05
mnx_Latn	0.11	**0.15**	0.08	0.06	0.05	naw_Latn	0.08	**0.10**	0.05	0.05	0.05
moa_Latn	**0.08**	0.04	0.06	0.05	0.05	nbc_Latn	0.09	**0.12**	0.06	0.05	0.07
moc_Latn	0.08	**0.13**	0.06	0.05	0.05	nbe_Latn	**0.17**	0.12	0.06	0.06	0.07
mog_Latn	0.16	0.20	0.13	0.07	**0.21**	nbl_Latn	0.09	0.13	0.15	0.21	**0.29**
mop_Latn	0.20	0.10	0.07	0.06	**0.27**	nbu_Latn	**0.15**	0.09	0.05	0.05	0.05
mor_Latn	**0.14**	0.11	0.05	0.05	0.05	nca_Latn	0.07	**0.11**	0.06	0.06	0.06
mos_Latn	0.11	**0.11**	0.06	0.05	0.06	nch_Latn	0.10	**0.12**	0.07	0.05	0.06
mox_Latn	0.12	**0.15**	0.07	0.05	0.05	ncj_Latn	**0.14**	0.10	0.05	0.05	0.07
mpg_Latn	**0.12**	0.09	0.05	0.05	0.05	ncl_Latn	0.10	0.09	0.06	0.09	**0.13**
mpm_Latn	0.04	**0.15**	0.05	0.05	0.05	ncq_Laoo	0.07	0.05	**0.11**	0.04	0.10
mps_Latn	0.15	**0.16**	0.05	0.06	0.07	nct_Latn	**0.12**	0.09	0.06	0.05	0.06
mpt_Latn	**0.13**	0.11	0.07	0.05	0.07	ncu_Latn	0.06	**0.09**	0.05	0.05	0.05
mpx_Latn	0.09	**0.10**	0.07	0.05	0.05	ndc_Latn	0.07	0.15	0.10	0.07	**0.16**
mqb_Latn	**0.11**	0.09	0.04	0.05	0.05	nde_Latn	0.09	0.13	0.15	0.21	**0.29**
mqj_Latn	0.11	**0.18**	0.12	0.05	0.16	ndi_Latn	**0.11**	0.10	0.06	0.05	0.05
mqy_Latn	0.11	**0.16**	0.13	0.05	0.11	ndj_Latn	**0.13**	0.11	0.06	0.05	0.12
mri_Latn	0.16	0.09	0.09	0.05	**0.19**	ndo_Latn	0.11	0.11	0.09	0.05	**0.16**
mrw_Latn	0.09	0.19	0.10	0.14	**0.31**	ndp_Latn	0.10	**0.11**	0.10	0.05	0.07
msa_Latn	0.08	0.22	0.42	0.42	**0.52**	nds_Latn	0.15	0.19	0.14	0.07	**0.27**
msb_Latn	0.12	0.21	0.28	0.24	**0.49**	ndy_Latn	0.07	**0.14**	0.07	0.06	0.14
mse_Latn	**0.12**	0.09	0.08	0.05	0.05	ndz_Latn	0.09	**0.15**	0.05	0.05	0.05
msk_Latn	0.09	0.14	0.09	0.10	**0.28**	neb_Latn	**0.12**	0.07	0.05	0.05	0.05
msm_Latn	0.12	0.10	0.07	0.06	**0.21**	nep_Deva	0.07	0.32	0.62	0.64	**0.68**
msy_Latn	0.07	**0.09**	0.06	0.05	0.06	nfa_Latn	0.07	**0.09**	0.06	0.05	0.05
mta_Latn	**0.12**	0.10	0.05	0.05	0.05	nfr_Latn	**0.15**	0.11	0.07	0.05	0.05
mtg_Latn	**0.11**	0.09	0.05	0.05	0.05	ngc_Latn	0.11	**0.14**	0.07	0.05	0.14
mti_Latn	0.14	0.14	0.08	0.08	**0.15**	ngp_Latn	0.13	0.17	0.16	0.12	**0.19**
mtj_Latn	0.08	**0.10**	0.08	0.05	0.06	ngu_Latn	0.06	0.09	0.05	0.06	**0.15**
mto_Latn	0.11	**0.14**	0.05	0.05	0.05	nhd_Latn	0.12	**0.17**	0.09	0.05	0.10
mtp_Latn	0.11	**0.12**	0.05	0.05	0.05	nhe_Latn	0.10	**0.13**	0.07	0.05	0.08
mua_Latn	**0.16**	0.10	0.05	0.05	0.06	nhg_Latn	0.10	0.12	0.05	0.05	**0.14**
mug_Latn	**0.13**	0.11	0.05	0.06	0.07	nhi_Latn	**0.12**	0.10	0.06	0.05	0.08
muh_Latn	0.12	**0.18**	0.15	0.05	0.05	nho_Latn	0.16	**0.17**	0.07	0.05	0.12
mup_Deva	0.07	0.28	0.35	0.32	**0.49**	nhr_Latn	**0.17**	0.14	0.05	0.05	0.07
mur_Latn	**0.14**	0.12	0.05	0.05	0.08	nhu_Latn	**0.16**	0.10	0.05	0.05	0.05
mux_Latn	**0.12**	0.11	0.06	0.05	0.05	nhw_Latn	0.08	**0.14**	0.07	0.05	0.06
muy_Latn	**0.11**	0.07	0.05	0.05	0.05	nhx_Latn	0.13	0.14	0.08	0.05	**0.19**
mva_Latn	0.07	**0.15**	0.07	0.05	0.07	nhy_Latn	0.14	**0.16**	0.05	0.06	0.15
mvn_Latn	**0.12**	0.09	0.05	0.05	0.05	nii_Latn	**0.14**	0.09	0.05	0.05	0.05
mvp_Latn	0.11	0.12	0.15	0.05	**0.22**	nij_Latn	0.09	**0.23**	0.18	0.16	0.23
mwm_Latn	**0.12**	0.08	0.05	0.05	0.05	nim_Latn	0.07	**0.12**	0.06	0.05	0.06
mwq_Latn	0.10	**0.10**	0.06	0.05	0.05	nin_Latn	0.07	**0.13**	0.08	0.05	0.07
mwv_Latn	0.07	**0.14**	0.10	0.05	0.13	niq_Latn	0.09	**0.10**	0.05	0.05	0.07
mww_Latn	**0.10**	0.06	0.05	0.05	0.05	niy_Latn	**0.11**	0.05	0.08	0.05	0.05
mxb_Latn	0.09	**0.14**	0.05	0.05	0.06	njb_Latn	**0.17**	0.13	0.05	0.05	0.05
mxp_Latn	0.10	**0.12**	0.05	0.05	0.06	njm_Latn	**0.16**	0.09	0.06	0.05	0.06
mxq_Latn	0.09	0.06	0.05	0.05	**0.10**	njn_Latn	0.09	**0.12**	0.05	0.05	0.05
mxt_Latn	**0.13**	0.12	0.04	0.05	0.07	njo_Latn	**0.12**	0.11	0.05	0.05	0.06
mxv_Latn	0.10	**0.16**	0.05	0.05	0.16	njz_Latn	0.08	**0.13**	0.05	0.05	0.05
mya_Mymr	0.07	0.26	0.42	**0.61**	0.51	nkf_Latn	0.13	**0.16**	0.06	0.05	0.06
myb_Latn	0.07	**0.13**	0.07	0.05	0.09	nki_Latn	0.10	0.13	0.05	0.05	**0.26**
myk_Latn	0.07	**0.12**	0.05	0.05	0.07	nko_Latn	0.10	**0.10**	0.05	0.05	0.05
myu_Latn	0.07	**0.12**	0.09	0.05	0.06	nlc_Latn	0.11	**0.12**	0.05	0.05	0.05
myv_Cyrl	0.07	0.08	0.08	0.05	**0.19**	nld_Latn	0.28	0.43	**0.60**	0.58	0.53
myw_Latn	0.07	**0.15**	0.06	0.05	0.05	nlg_Latn	0.20	**0.21**	0.07	0.09	0.21
myx_Latn	0.10	**0.12**	0.04	0.05	0.10	nma_Latn	0.07	**0.12**	0.08	0.05	0.05
myy_Latn	0.07	0.08	**0.09**	0.05	0.06	nmf_Latn	0.08	**0.12**	0.05	0.05	0.06
mza_Latn	0.10	**0.13**	0.06	0.05	0.05	nmh_Latn	0.09	**0.10**	0.05	0.06	0.06
mzh_Latn	0.08	0.19	0.08	0.05	**0.24**	nmo_Latn	0.10	**0.10**	0.06	0.05	0.06
mzk_Latn	0.14	**0.14**	0.08	0.06	0.07	nmz_Latn	**0.15**	0.12	0.08	0.05	0.10
mzl_Latn	**0.10**	0.09	0.06	0.05	0.05	nnb_Latn	0.10	**0.14**	0.07	0.05	0.10

Table 7: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
nng_Latn	0.07	**0.09**	0.07	0.05	0.06	oym_Latn	0.07	**0.12**	0.05	0.05	0.05
nnh_Latn	0.08	**0.14**	0.07	0.05	0.08	ozm_Latn	**0.13**	0.06	0.06	0.05	0.05
nnl_Latn	0.12	**0.12**	0.07	0.05	0.06	pab_Latn	**0.12**	0.05	0.05	0.05	0.05
nno_Latn	0.15	0.46	**0.58**	0.56	0.43	pad_Latn	0.13	**0.15**	0.06	0.05	0.06
nnp_Latn	0.07	**0.08**	0.07	0.05	0.05	pag_Latn	0.14	0.14	0.20	0.17	**0.33**
nnq_Latn	0.14	**0.15**	0.11	0.10	0.14	pah_Latn	0.09	**0.15**	0.06	0.05	0.05
nnw_Latn	**0.07**	0.05	0.05	0.05	0.05	pam_Latn	0.13	0.18	0.11	0.11	**0.38**
noa_Latn	0.07	**0.08**	0.05	0.06	0.05	pan_Guru	0.07	0.31	0.58	0.67	**0.69**
nob_Latn	0.16	0.38	0.59	**0.60**	0.56	pao_Latn	0.10	**0.13**	0.07	0.05	0.08
nod_Thai	0.07	0.09	0.47	0.50	**0.50**	pap_Latn	0.15	0.31	0.30	0.23	**0.52**
nog_Cyrl	0.07	0.16	0.18	0.38	**0.41**	pau_Latn	0.16	0.18	0.06	0.05	**0.21**
nop_Latn	0.09	**0.15**	0.05	0.05	0.05	pbb_Latn	**0.17**	0.12	0.07	0.05	0.07
nor_Latn	0.16	0.38	**0.60**	0.60	0.55	pbc_Latn	**0.17**	0.12	0.05	0.05	0.05
not_Latn	0.07	0.09	**0.13**	0.06	0.11	pbi_Latn	**0.13**	0.06	0.05	0.05	0.07
nou_Latn	**0.16**	0.11	0.11	0.06	0.13	pbl_Latn	0.10	0.16	0.13	0.05	**0.26**
nph_Latn	0.08	**0.10**	0.09	0.05	0.05	pck_Latn	0.12	0.14	0.06	0.05	**0.19**
npi_Deva	0.07	0.32	0.59	0.66	**0.67**	pcm_Latn	0.19	0.18	0.30	0.29	**0.45**
npl_Latn	0.10	0.09	0.05	0.07	**0.18**	pdc_Latn	0.19	0.14	0.14	0.15	**0.27**
npo_Latn	**0.13**	0.09	0.07	0.05	0.05	pdt_Latn	0.17	0.18	0.17	0.12	**0.34**
npy_Latn	0.09	**0.13**	0.11	0.05	0.07	pes_Arab	0.07	0.42	**0.66**	0.66	0.63
nre_Latn	0.10	**0.15**	0.07	0.05	0.07	pez_Latn	0.08	**0.23**	0.09	0.05	0.10
nri_Latn	0.11	**0.12**	0.09	0.05	0.09	pfe_Latn	**0.10**	0.05	0.05	0.05	0.05
nsa_Latn	0.07	**0.12**	0.09	0.05	0.06	pib_Latn	0.07	**0.11**	0.04	0.05	0.06
nse_Latn	0.12	0.17	0.13	0.07	**0.23**	pio_Latn	0.07	0.09	0.06	0.05	**0.12**
nsm_Latn	**0.13**	0.07	0.06	0.05	0.06	pir_Latn	0.10	**0.11**	0.06	0.05	0.05
nsn_Latn	**0.15**	0.09	0.06	0.07	0.12	pis_Latn	**0.21**	0.11	0.12	0.06	0.20
nso_Latn	0.11	0.13	0.12	0.05	**0.27**	pjt_Latn	0.07	**0.09**	0.05	0.05	0.08
nst_Latn	**0.18**	0.10	0.05	0.05	0.06	pkb_Latn	0.11	0.15	0.12	0.07	**0.28**
nsu_Latn	**0.13**	0.10	0.06	0.05	0.12	plg_Latn	**0.16**	0.13	0.08	0.05	0.08
ntp_Latn	0.07	**0.10**	0.05	0.05	0.04	pls_Latn	0.07	0.19	0.07	0.14	**0.27**
ntr_Latn	0.07	**0.12**	0.05	0.05	0.05	plt_Latn	0.12	0.05	0.38	**0.54**	0.50
ntu_Latn	0.07	**0.08**	0.06	0.05	0.05	plu_Latn	**0.13**	0.08	0.05	0.05	0.05
nuj_Latn	0.11	**0.14**	0.06	0.05	0.07	plw_Latn	0.14	**0.19**	0.10	0.06	0.19
nus_Latn	**0.13**	0.10	0.05	0.05	0.05	pma_Latn	0.14	**0.16**	0.07	0.05	0.06
nuy_Latn	**0.23**	0.10	0.05	0.05	0.05	pmf_Latn	0.11	**0.22**	0.10	0.09	0.20
nvm_Latn	0.07	**0.11**	0.05	0.05	0.05	pmx_Latn	**0.09**	0.08	0.06	0.06	0.06
nwb_Latn	**0.14**	0.06	0.05	0.05	0.05	pne_Latn	0.08	**0.23**	0.09	0.05	0.11
nwi_Latn	**0.15**	0.13	0.05	0.05	0.07	pny_Latn	**0.08**	0.05	0.05	0.05	0.05
nwx_Deva	0.07	0.16	0.18	0.14	**0.29**	poe_Latn	0.13	**0.13**	0.05	0.05	0.06
nxd_Latn	0.07	**0.09**	0.07	0.05	0.07	poh_Latn	0.11	0.09	0.12	0.05	**0.37**
nya_Latn	0.07	0.14	0.08	0.06	**0.26**	poi_Latn	0.12	**0.15**	0.05	0.07	0.12
nyf_Latn	0.15	0.19	0.21	0.17	**0.25**	pol_Latn	0.09	0.48	0.60	**0.65**	0.61
nyn_Latn	0.09	0.11	0.06	0.05	**0.20**	pon_Latn	0.14	**0.21**	0.08	0.05	0.08
nyo_Latn	0.07	**0.16**	0.05	0.05	0.15	por_Latn	0.16	0.52	0.57	**0.64**	0.61
nyy_Latn	0.11	**0.16**	0.08	0.05	0.09	pos_Latn	0.12	0.17	0.06	0.06	**0.27**
nza_Latn	0.07	**0.10**	0.05	0.05	0.05	poy_Latn	0.14	**0.18**	0.08	0.05	0.07
nzi_Latn	0.09	**0.16**	0.05	0.05	0.05	ppk_Latn	0.15	0.15	0.06	0.04	**0.16**
nzm_Latn	**0.11**	0.09	0.08	0.06	0.06	ppo_Latn	0.10	**0.18**	0.05	0.05	0.05
obo_Latn	**0.15**	0.12	0.05	0.05	0.07	pps_Latn	0.10	**0.11**	0.06	0.05	0.08
ojb_Cans	0.07	**0.12**	0.05	0.05	0.06	prf_Latn	0.12	0.20	0.15	0.13	**0.26**
oji_Latn	**0.11**	0.09	0.05	0.05	0.07	pri_Latn	0.07	**0.10**	0.05	0.05	0.05
ojs_Latn	0.07	**0.08**	0.05	0.05	0.06	prk_Latn	0.09	**0.13**	0.06	0.05	0.10
oku_Latn	**0.12**	0.11	0.05	0.05	0.05	prq_Latn	0.07	**0.08**	0.05	0.05	0.05
okv_Latn	0.13	**0.22**	0.14	0.08	0.13	prs_Arab	0.07	0.43	**0.66**	0.64	0.64
old_Latn	**0.13**	0.09	0.08	0.06	0.06	pse_Latn	0.07	0.28	0.36	0.38	**0.39**
omb_Latn	**0.17**	0.16	0.10	0.06	0.06	pss_Latn	0.10	**0.13**	0.06	0.05	0.08
omw_Latn	0.07	**0.08**	0.05	0.05	0.05	ptp_Latn	0.10	**0.11**	0.05	0.05	0.05
ong_Latn	0.07	**0.17**	0.07	0.05	0.06	ptu_Latn	0.11	0.15	0.14	0.05	**0.20**
ons_Latn	**0.11**	0.09	0.05	0.05	0.05	pua_Latn	0.08	0.09	0.09	0.05	**0.15**
ood_Latn	**0.16**	0.11	0.05	0.05	0.05	pui_Latn	0.09	**0.14**	0.05	0.06	0.06
opm_Latn	0.07	**0.14**	0.07	0.05	0.05	pwg_Latn	**0.18**	0.14	0.06	0.08	0.12
ori_Orya	0.07	0.04	0.58	**0.75**	0.65	pww_Thai	0.07	0.08	**0.10**	0.05	0.05
ory_Orya	0.07	0.04	0.56	**0.75**	0.64	pxm_Latn	0.08	**0.14**	0.06	0.05	0.05
oss_Cyrl	0.07	0.10	0.07	0.05	**0.11**	qub_Latn	0.08	0.12	0.06	0.06	**0.17**
otd_Latn	0.07	**0.25**	0.12	0.11	0.14	quc_Latn	0.18	0.14	0.07	0.05	**0.37**
ote_Latn	**0.08**	0.07	0.05	0.05	0.06	quf_Latn	0.07	**0.10**	0.05	0.05	0.06
otm_Latn	**0.10**	0.08	0.05	0.05	0.05	qug_Latn	0.07	0.11	0.09	0.05	**0.12**
otn_Latn	0.09	**0.11**	0.05	0.05	0.05	quh_Latn	0.07	0.12	0.07	0.05	**0.30**
otq_Latn	**0.14**	0.08	0.06	0.05	0.06	qul_Latn	0.07	0.14	0.06	0.07	**0.32**
ots_Latn	**0.11**	0.10	0.05	0.05	0.10	qup_Latn	0.07	**0.13**	0.05	0.05	0.13

Table 8: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
quw_Latn	0.07	0.10	0.07	0.05	**0.18**	shp_Latn	0.07	**0.12**	0.06	0.05	0.05
quy_Latn	0.07	0.11	0.07	0.06	**0.27**	shu_Latn	0.09	**0.20**	0.16	0.11	0.19
quz_Latn	0.07	0.10	0.07	0.05	**0.24**	sig_Latn	**0.13**	0.08	0.05	0.05	0.05
qva_Latn	0.07	0.10	0.07	0.05	**0.18**	sil_Latn	**0.14**	0.07	0.05	0.05	0.05
qvc_Latn	0.09	**0.11**	0.06	0.05	0.05	sim_Latn	0.08	**0.10**	0.06	0.05	0.07
qve_Latn	0.09	0.13	0.06	0.05	**0.33**	sin_Sinh	0.07	0.16	0.51	**0.67**	0.57
qvh_Latn	0.12	0.12	0.05	0.07	**0.24**	sja_Latn	0.10	**0.10**	0.05	0.05	0.05
qvi_Latn	0.06	**0.12**	0.06	0.05	0.10	sld_Latn	**0.14**	0.10	0.05	0.05	0.05
qvm_Latn	0.07	0.13	0.06	0.05	**0.19**	slk_Latn	0.09	0.48	**0.69**	0.64	0.56
qvn_Latn	0.07	0.10	0.05	0.06	**0.14**	sll_Latn	0.07	**0.11**	0.07	0.05	0.08
qvo_Latn	0.10	**0.11**	0.06	0.05	0.08	slv_Latn	0.17	0.50	**0.63**	0.60	0.60
qvs_Latn	0.09	0.10	0.05	0.05	**0.18**	sme_Latn	0.15	**0.17**	0.09	0.05	0.14
qvw_Latn	0.09	0.10	0.05	0.05	**0.13**	smk_Latn	0.10	0.10	0.08	0.06	**0.27**
qvz_Latn	0.09	0.10	0.06	0.05	**0.13**	sml_Latn	0.13	0.12	0.17	0.10	**0.23**
qwh_Latn	0.06	0.14	0.09	0.05	**0.22**	smo_Latn	0.10	0.07	0.08	0.05	**0.29**
qxh_Latn	0.07	0.11	0.04	0.05	**0.15**	smt_Latn	0.11	0.15	0.05	0.05	**0.21**
qxl_Latn	0.07	**0.11**	0.07	0.05	0.08	sna_Latn	0.07	0.11	0.11	0.08	**0.18**
qxn_Latn	0.07	0.15	0.07	0.05	**0.23**	snc_Latn	**0.15**	0.12	0.05	0.05	0.06
qxo_Latn	0.09	0.11	0.05	0.06	**0.23**	snd_Arab	0.07	0.19	0.61	**0.67**	0.61
qxr_Latn	0.07	0.13	0.10	0.05	**0.14**	snf_Latn	**0.14**	0.11	0.06	0.05	0.06
rad_Latn	0.09	**0.09**	0.06	0.05	0.06	snn_Latn	0.14	**0.17**	0.09	0.05	0.05
rai_Latn	0.16	**0.18**	0.05	0.07	0.12	snp_Latn	**0.12**	0.11	0.06	0.05	0.09
rap_Latn	0.13	0.13	0.06	0.05	**0.21**	snw_Latn	0.09	**0.11**	0.05	0.05	0.05
rar_Latn	0.10	0.07	0.06	0.05	**0.22**	sny_Latn	0.07	**0.13**	0.06	0.05	0.08
rav_Deva	0.07	0.09	**0.17**	0.05	0.07	som_Latn	0.08	0.09	0.31	0.39	**0.43**
raw_Latn	0.12	**0.14**	0.05	0.05	0.06	sop_Latn	0.15	0.14	0.07	0.05	**0.20**
rej_Latn	0.12	0.25	0.20	0.18	**0.31**	soq_Latn	**0.19**	0.17	0.05	0.07	0.08
rel_Latn	**0.15**	0.12	0.08	0.05	0.06	sot_Latn	0.13	0.10	0.09	0.05	**0.18**
rgu_Latn	0.07	0.07	0.04	0.04	**0.15**	soy_Latn	**0.16**	0.07	0.05	0.05	0.05
ria_Latn	0.08	**0.10**	0.06	0.05	0.06	spa_Latn	0.11	0.49	0.64	**0.69**	0.58
rim_Latn	0.13	**0.16**	0.05	0.06	0.07	spl_Latn	0.07	**0.12**	0.05	0.05	0.05
rjs_Deva	0.07	0.13	0.26	0.22	**0.28**	spp_Latn	**0.10**	0.08	0.06	0.05	0.09
rkb_Latn	**0.12**	0.07	0.05	0.05	0.08	sps_Latn	0.14	**0.17**	0.05	0.05	0.05
rmc_Latn	0.12	0.17	0.17	0.09	**0.18**	spy_Latn	0.07	**0.09**	0.05	0.05	0.07
rmo_Latn	**0.17**	0.16	0.08	0.06	0.11	sqi_Latn	0.10	0.33	**0.68**	0.66	0.65
rmy_Latn	0.12	**0.23**	0.10	0.06	0.22	sri_Latn	0.07	**0.13**	0.04	0.05	0.06
rnl_Latn	0.11	**0.14**	0.05	0.05	0.09	srm_Latn	0.12	0.09	0.06	0.05	**0.21**
ron_Latn	0.11	0.50	0.62	**0.65**	0.53	srn_Latn	0.07	0.15	0.07	0.05	**0.42**
roo_Latn	0.07	**0.10**	0.05	0.05	0.05	srp_Latn	0.09	0.47	0.59	0.59	**0.63**
rop_Latn	0.20	**0.20**	0.06	0.05	0.20	srq_Latn	**0.16**	0.07	0.11	0.07	0.10
row_Latn	0.07	**0.08**	0.06	0.05	0.08	ssd_Latn	0.12	**0.17**	0.05	0.05	0.05
rro_Latn	0.08	**0.11**	0.07	0.05	0.05	ssg_Latn	**0.13**	0.06	0.11	0.06	0.06
rub_Latn	0.13	**0.13**	0.08	0.05	0.08	ssw_Latn	0.07	0.11	0.09	0.12	**0.24**
ruf_Latn	0.14	**0.20**	0.10	0.09	0.11	ssx_Latn	0.11	**0.13**	0.07	0.05	0.06
rug_Latn	0.10	**0.13**	0.06	0.05	0.06	stn_Latn	**0.19**	0.16	0.11	0.05	0.15
run_Latn	0.16	0.15	0.09	0.06	**0.27**	stp_Latn	**0.09**	0.04	0.05	0.05	0.05
rus_Cyrl	0.07	0.50	0.55	**0.67**	0.64	sua_Latn	**0.18**	0.13	0.05	0.05	0.05
rwo_Latn	0.07	**0.10**	0.07	0.06	0.05	suc_Latn	**0.13**	0.11	0.06	0.05	0.08
sab_Latn	0.07	**0.10**	0.08	0.05	0.06	sue_Latn	0.13	**0.14**	0.08	0.05	0.06
sag_Latn	0.11	0.19	0.10	0.06	**0.20**	suk_Latn	**0.16**	0.13	0.07	0.07	0.09
sah_Cyrl	0.07	0.12	0.08	0.05	**0.30**	sun_Latn	0.09	0.33	0.45	**0.50**	0.45
saj_Latn	0.05	**0.10**	0.05	0.05	0.08	sur_Latn	**0.15**	0.11	0.06	0.05	0.10
san_Taml	**0.07**	0.05	0.07	0.05	0.05	sus_Latn	0.12	**0.15**	0.04	0.05	0.05
sas_Latn	0.11	0.22	0.28	0.24	**0.30**	suz_Deva	0.07	0.10	0.11	0.06	**0.27**
sat_Latn	**0.12**	0.08	0.06	0.05	0.06	swe_Latn	0.13	0.48	**0.73**	0.60	0.59
sba_Latn	**0.12**	0.11	0.06	0.05	0.11	swg_Latn	0.21	0.27	0.25	0.34	**0.35**
sbd_Latn	**0.12**	0.09	0.06	0.06	0.05	swh_Latn	0.12	0.31	0.50	**0.57**	0.54
sbl_Latn	0.12	0.08	0.18	0.12	**0.21**	swk_Latn	0.11	0.13	0.04	0.06	**0.19**
sck_Deva	0.07	0.17	0.28	0.44	**0.47**	swp_Latn	0.08	**0.10**	0.08	0.06	0.06
sda_Latn	0.11	**0.16**	0.09	0.05	0.13	sxb_Latn	0.10	0.13	0.08	0.05	**0.14**
sdq_Latn	0.06	0.15	0.12	0.10	**0.16**	sxn_Latn	0.07	0.09	0.05	0.05	**0.18**
seh_Latn	0.13	0.11	0.07	0.06	**0.23**	syb_Latn	**0.13**	0.09	0.10	0.05	0.11
ses_Latn	**0.14**	0.09	0.07	0.05	0.07	syc_Syrc	0.07	0.05	0.05	0.08	**0.10**
sey_Latn	0.06	**0.10**	0.05	0.05	0.05	syl_Latn	**0.07**	0.06	0.05	0.05	0.05
sgb_Latn	0.14	0.22	0.17	0.10	**0.31**	szb_Latn	0.07	**0.21**	0.04	0.05	0.06
sgw_Ethi	0.07	0.09	0.10	0.13	**0.24**	tab_Cyrl	0.07	0.11	**0.12**	0.05	0.10
sgz_Latn	0.07	**0.13**	0.06	0.05	0.07	tac_Latn	0.12	**0.20**	0.05	0.05	0.07
shi_Latn	**0.13**	0.07	0.05	0.05	0.07	taj_Deva	0.07	0.13	0.14	0.09	**0.20**
shk_Latn	**0.11**	0.07	0.06	0.05	0.07	tam_Taml	0.07	0.35	0.53	0.56	**0.60**
shn_Mymr	**0.07**	0.05	0.06	0.05	0.05	tap_Latn	0.14	0.18	0.10	0.08	**0.20**

Table 9: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
taq_Latn	0.10	**0.11**	0.07	0.05	0.06	tro_Latn	**0.15**	0.12	0.07	0.05	0.07
tar_Latn	0.10	**0.10**	0.05	0.05	0.05	trp_Latn	**0.10**	0.08	0.06	0.05	0.05
tat_Cyrl	0.07	0.31	0.12	0.15	**0.45**	trq_Latn	0.05	**0.12**	0.05	0.05	0.07
tav_Latn	**0.13**	0.11	0.05	0.05	0.09	trs_Latn	0.06	**0.10**	0.07	0.05	0.10
taw_Latn	**0.14**	0.09	0.07	0.05	0.07	tsg_Latn	0.11	0.17	0.15	0.11	**0.27**
tbc_Latn	0.09	**0.12**	0.05	0.05	0.06	tsn_Latn	0.12	0.12	0.09	0.05	**0.23**
tbg_Latn	0.07	**0.14**	0.08	0.05	0.06	tsw_Latn	0.07	**0.12**	0.07	0.05	0.08
tbk_Latn	0.07	0.17	0.11	0.11	**0.27**	tsz_Latn	0.08	0.10	0.08	0.05	**0.14**
tbl_Latn	0.12	**0.12**	0.12	0.05	0.06	ttc_Latn	0.14	**0.20**	0.10	0.05	0.09
tbo_Latn	0.12	**0.13**	0.10	0.05	0.05	tte_Latn	0.07	0.07	**0.08**	0.05	0.05
tbw_Latn	0.11	0.15	0.08	0.06	**0.25**	ttq_Latn	0.09	0.09	0.07	0.06	**0.10**
tby_Latn	**0.14**	0.12	0.06	0.05	0.12	ttr_Cyrl	0.07	0.31	0.18	0.13	**0.42**
tbz_Latn	0.07	**0.09**	0.05	0.05	0.05	tuc_Latn	**0.18**	0.10	0.05	0.05	0.05
tca_Latn	0.07	**0.07**	0.05	0.05	0.07	tue_Latn	0.07	**0.10**	0.04	0.05	0.05
tcc_Latn	0.09	**0.10**	0.05	0.05	0.05	tuf_Latn	0.11	**0.13**	0.10	0.05	0.06
tcs_Latn	0.21	0.19	0.11	0.06	**0.21**	tui_Latn	**0.17**	0.14	0.08	0.05	0.07
tcz_Latn	**0.12**	0.11	0.09	0.05	0.05	tuk_Latn	0.11	0.11	0.22	0.22	**0.44**
tdt_Latn	0.15	0.15	0.09	0.05	**0.36**	tul_Latn	0.12	**0.18**	0.05	0.05	0.05
ted_Latn	**0.10**	0.09	0.05	0.05	0.05	tum_Latn	0.13	**0.22**	0.10	0.07	0.21
tee_Latn	0.06	0.07	0.06	0.05	**0.14**	tuo_Latn	**0.12**	0.09	0.04	0.05	0.08
tel_Telu	0.07	0.30	0.60	0.67	**0.67**	tur_Latn	0.11	0.29	**0.68**	0.68	0.63
tem_Latn	**0.12**	0.05	0.06	0.05	0.05	tvk_Latn	0.11	**0.19**	0.08	0.05	0.10
teo_Latn	0.09	**0.12**	0.05	0.07	0.08	twb_Latn	0.10	**0.12**	0.05	0.05	0.06
ter_Latn	0.12	**0.13**	0.06	0.05	0.06	twi_Latn	0.10	**0.15**	0.05	0.05	0.13
tet_Latn	0.07	0.11	0.05	0.05	**0.13**	twu_Latn	0.12	0.15	**0.16**	0.05	0.07
tfr_Latn	0.12	**0.14**	0.08	0.05	0.05	txq_Latn	0.07	**0.15**	0.09	0.05	0.06
tgk_Cyrl	0.07	0.19	0.05	0.04	**0.31**	txu_Latn	0.13	**0.17**	0.07	0.05	0.05
tgl_Latn	0.13	0.29	0.47	0.55	**0.55**	tyv_Cyrl	0.07	0.12	0.19	0.18	**0.44**
tgo_Latn	0.09	**0.14**	0.05	0.05	0.05	tzh_Latn	0.08	0.10	0.09	0.05	**0.22**
tgp_Latn	0.15	**0.21**	0.08	0.09	0.09	tzj_Latn	0.13	0.15	0.09	0.06	**0.21**
tha_Thai	0.07	0.08	0.56	**0.60**	0.56	tzo_Latn	0.08	0.11	0.07	0.05	**0.30**
thk_Latn	**0.16**	0.10	0.04	0.05	0.05	ubr_Latn	**0.15**	0.13	0.06	0.05	0.10
thl_Deva	0.07	0.24	0.34	0.44	**0.45**	ubu_Latn	**0.13**	0.07	0.07	0.05	0.06
tif_Latn	0.07	**0.10**	0.05	0.05	0.08	udm_Cyrl	0.07	0.10	0.07	0.05	**0.20**
tih_Latn	0.09	0.11	0.09	0.05	**0.26**	udu_Latn	**0.19**	0.11	0.05	0.05	0.08
tik_Latn	**0.09**	0.07	0.05	0.05	0.05	uig_Cyrl	0.07	0.20	0.13	0.14	**0.44**
tim_Latn	0.07	**0.11**	0.06	0.05	0.06	ukr_Cyrl	0.07	0.40	0.64	**0.67**	0.57
tir_Ethi	0.07	0.06	0.27	0.22	**0.38**	upv_Latn	0.10	**0.12**	0.06	0.05	0.05
tiy_Latn	0.15	**0.17**	0.08	0.06	0.08	ura_Latn	0.07	**0.08**	0.05	0.05	0.05
tke_Latn	0.13	**0.14**	0.06	0.05	0.09	urb_Latn	**0.14**	0.11	0.12	0.05	0.05
tku_Latn	0.10	0.09	0.06	0.05	**0.15**	urd_Arab	0.07	0.37	0.49	**0.67**	0.56
tlb_Latn	0.09	**0.13**	0.07	0.05	0.09	urk_Thai	0.07	**0.09**	0.07	0.05	0.05
tlf_Latn	0.07	0.07	**0.09**	0.05	0.08	urt_Latn	0.06	**0.13**	0.08	0.05	0.06
tlh_Latn	0.22	**0.29**	0.24	0.13	0.29	ury_Latn	**0.14**	0.10	0.05	0.05	0.06
tlj_Latn	**0.19**	0.14	0.11	0.05	0.12	usa_Latn	0.07	**0.10**	0.06	0.05	0.05
tmc_Latn	0.10	**0.12**	0.05	0.05	0.08	usp_Latn	0.18	0.11	0.07	0.05	**0.24**
tmd_Latn	0.07	**0.08**	0.05	0.05	0.05	uth_Latn	0.07	**0.10**	0.09	0.05	0.07
tna_Latn	0.11	0.12	**0.13**	0.05	0.07	uvh_Latn	0.07	**0.09**	0.07	0.05	0.05
tnk_Latn	0.11	**0.11**	0.05	0.05	0.04	uvl_Latn	0.09	**0.16**	0.06	0.05	0.09
tnn_Latn	**0.13**	0.10	0.07	0.05	0.07	uzb_Latn	0.09	0.14	0.54	**0.59**	0.58
tnp_Latn	**0.12**	0.07	0.05	0.07	0.06	uzn_Cyrl	0.07	0.14	0.07	0.10	**0.47**
tnr_Latn	**0.13**	0.07	0.05	0.05	0.06	vag_Latn	0.10	**0.11**	0.05	0.05	0.06
tob_Latn	0.07	**0.12**	0.04	0.05	0.09	vap_Latn	**0.19**	0.12	0.06	0.05	0.17
toc_Latn	0.06	**0.09**	0.05	0.05	0.05	var_Latn	0.10	**0.13**	0.07	0.05	0.06
toh_Latn	0.11	0.12	0.06	0.06	**0.22**	ven_Latn	0.11	**0.12**	0.06	0.05	0.11
toi_Latn	0.07	0.13	0.08	0.06	**0.24**	vid_Latn	0.11	**0.14**	0.11	0.09	0.09
toj_Latn	0.12	0.06	0.07	0.05	**0.29**	vie_Latn	0.09	0.38	0.54	**0.63**	0.53
ton_Latn	0.09	0.08	0.05	0.05	**0.26**	viv_Latn	0.07	**0.11**	0.06	0.05	0.05
too_Latn	0.10	**0.11**	0.06	0.05	0.11	vmy_Latn	**0.13**	0.10	0.05	0.05	0.10
top_Latn	0.08	0.13	0.05	0.05	**0.17**	vun_Latn	**0.13**	0.10	0.06	0.05	0.05
tos_Latn	0.06	**0.07**	0.05	0.05	0.07	vut_Latn	**0.08**	0.05	0.05	0.05	0.05
tpi_Latn	0.17	0.17	0.09	0.06	**0.31**	waj_Latn	**0.10**	0.08	0.06	0.05	0.06
tpm_Latn	**0.14**	0.12	0.06	0.05	0.06	wal_Latn	**0.15**	0.10	0.06	0.06	0.13
tpp_Latn	0.13	**0.15**	0.06	0.05	0.10	wap_Latn	0.11	**0.11**	0.06	0.05	0.06
tpt_Latn	0.14	0.07	0.09	0.05	**0.15**	war_Latn	0.11	0.16	0.15	0.14	**0.37**
tpz_Latn	**0.12**	0.11	0.06	0.05	0.06	way_Latn	0.10	**0.12**	0.07	0.05	0.05
tqb_Latn	0.07	**0.11**	0.08	0.05	0.05	wba_Latn	0.09	0.10	0.08	0.06	**0.11**
tqo_Latn	**0.12**	0.08	0.06	0.05	0.05	wbm_Latn	0.09	**0.13**	0.06	0.05	0.09
trc_Latn	0.05	**0.14**	0.05	0.05	0.07	wbp_Latn	0.07	**0.07**	0.06	0.05	0.05
trn_Latn	0.12	**0.15**	0.06	0.06	0.05	wca_Latn	0.07	**0.14**	0.05	0.05	0.08

Table 10: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m	Language	BOW	mBERT	XLM-R B	XLM-R L	Glot500-m
wer_Latn	0.09	**0.15**	0.05	0.05	0.05	zac_Latn	0.12	**0.20**	0.09	0.09	0.18
whk_Latn	0.11	**0.17**	0.07	0.05	0.11	zad_Latn	**0.15**	0.10	0.04	0.05	0.05
wim_Latn	0.07	**0.08**	0.06	0.05	0.08	zae_Latn	**0.14**	0.13	0.10	0.05	0.06
wiu_Latn	0.12	**0.13**	0.05	0.06	0.05	zai_Latn	0.08	0.21	0.13	0.09	**0.25**
wmw_Latn	0.14	0.16	0.23	0.31	**0.41**	zam_Latn	0.09	**0.16**	0.07	0.05	0.13
wnc_Latn	0.07	**0.12**	0.07	0.06	0.05	zao_Latn	**0.14**	0.09	0.06	0.05	0.06
wnu_Latn	0.11	**0.13**	0.05	0.05	0.05	zar_Latn	0.11	**0.17**	0.06	0.05	0.08
wob_Latn	**0.11**	0.06	0.05	0.05	0.05	zas_Latn	0.07	**0.16**	0.07	0.06	0.13
wol_Latn	**0.16**	0.12	0.07	0.05	0.07	zat_Latn	0.13	0.11	0.11	0.06	**0.13**
wos_Latn	**0.16**	0.10	0.08	0.05	0.06	zav_Latn	**0.07**	0.06	0.05	0.05	0.06
wrs_Latn	**0.15**	0.10	0.06	0.05	0.05	zaw_Latn	0.07	0.06	0.06	0.05	**0.07**
wsg_Telu	0.07	0.09	**0.13**	0.08	0.07	zca_Latn	0.21	0.14	0.18	0.06	**0.21**
wsk_Latn	0.12	**0.15**	0.08	0.05	0.10	zho_Hani	0.07	0.39	**0.63**	0.63	0.59
wuv_Latn	**0.18**	0.09	0.09	0.05	0.06	zia_Latn	**0.14**	0.11	0.06	0.05	0.06
wwa_Latn	**0.16**	0.08	0.05	0.06	0.05	ziw_Latn	0.13	0.17	0.14	0.11	**0.23**
xal_Cyrl	0.07	0.12	0.08	0.05	**0.14**	zlm_Latn	0.07	0.47	0.68	**0.71**	0.62
xav_Latn	0.11	**0.13**	0.08	0.05	0.10	zoc_Latn	0.11	0.08	0.06	0.05	**0.11**
xbr_Latn	0.09	**0.09**	0.08	0.05	0.07	zom_Latn	0.10	0.16	0.13	0.05	**0.27**
xed_Latn	**0.11**	0.10	0.06	0.05	0.07	zos_Latn	0.15	**0.16**	0.05	0.06	0.14
xho_Latn	0.09	0.14	0.21	0.30	**0.34**	zpc_Latn	**0.13**	0.12	0.11	0.05	0.12
xla_Latn	**0.13**	0.08	0.08	0.05	0.05	zpi_Latn	0.13	**0.16**	0.09	0.05	0.08
xmm_Latn	0.14	0.30	**0.42**	0.40	0.40	zpl_Latn	0.07	0.13	0.13	0.06	**0.17**
xnn_Latn	0.07	0.11	0.10	0.08	**0.19**	zpm_Latn	**0.17**	0.14	0.05	0.06	0.08
xog_Latn	0.07	0.16	0.06	0.06	**0.22**	zpo_Latn	0.10	**0.15**	0.13	0.06	0.10
xon_Latn	0.06	**0.17**	0.05	0.05	0.05	zpq_Latn	0.07	**0.10**	0.06	0.05	0.09
xpe_Latn	0.08	**0.11**	0.05	0.05	0.06	zpt_Latn	0.11	0.11	0.10	0.05	**0.16**
xrb_Latn	0.11	**0.11**	0.05	0.05	0.05	zpu_Latn	**0.14**	0.08	0.05	0.05	0.06
xsb_Latn	0.11	0.14	0.11	0.08	**0.23**	zpv_Latn	**0.10**	0.08	0.05	0.05	0.05
xsi_Latn	0.09	**0.13**	0.05	0.05	0.05	zpz_Latn	0.05	0.07	**0.08**	0.05	0.05
xsm_Latn	**0.19**	0.08	0.05	0.05	0.05	zsm_Latn	0.07	0.53	**0.71**	0.63	0.58
xsr_Deva	0.07	**0.09**	0.05	0.05	0.06	zsr_Latn	0.09	**0.12**	0.07	0.05	0.09
xsu_Latn	0.13	**0.15**	0.05	0.05	0.08	ztq_Latn	0.10	0.13	0.10	0.08	**0.19**
xtd_Latn	0.14	**0.16**	0.05	0.05	0.07	zty_Latn	0.11	0.06	0.09	0.05	**0.12**
xtm_Latn	0.07	**0.15**	0.06	0.06	0.08	zul_Latn	0.07	0.11	0.23	0.33	**0.37**
xtn_Latn	0.09	**0.16**	0.07	0.06	0.13	zyb_Latn	**0.15**	0.10	0.06	0.05	0.05
xuo_Latn	**0.10**	0.08	0.05	0.05	0.05	zyp_Latn	0.10	**0.15**	0.05	0.05	0.06
yaa_Latn	0.07	**0.11**	0.06	0.05	0.06						
yad_Latn	**0.11**	0.09	0.05	0.05	0.05						
yal_Latn	**0.15**	0.13	0.06	0.05	0.07						
yam_Latn	**0.13**	0.05	0.05	0.05	0.05						
yan_Latn	0.10	**0.13**	0.05	0.05	0.05						
yao_Latn	0.13	0.13	0.06	0.05	**0.15**						
yap_Latn	0.13	0.14	0.07	0.05	**0.22**						
yaq_Latn	0.16	**0.16**	0.07	0.05	0.06						
yas_Latn	**0.13**	0.10	0.05	0.05	0.05						
yat_Latn	**0.11**	0.05	0.05	0.05	0.06						
yaz_Latn	0.07	**0.12**	0.08	0.05	0.05						
ybb_Latn	0.07	**0.09**	0.05	0.05	0.05						
yby_Latn	0.07	**0.08**	0.07	0.07	0.05						
ycn_Latn	**0.10**	0.09	0.05	0.05	0.05						
yim_Latn	**0.13**	0.12	0.09	0.05	0.06						
yka_Latn	0.09	0.14	0.10	0.07	**0.26**						
yle_Latn	0.07	**0.13**	0.05	0.05	0.05						
yli_Latn	0.11	**0.17**	0.09	0.05	0.10						
yml_Latn	0.08	**0.08**	0.05	0.05	0.06						
yom_Latn	0.09	0.16	0.06	0.05	**0.21**						
yon_Latn	**0.12**	0.11	0.11	0.05	0.09						
yor_Latn	0.11	**0.14**	0.10	0.05	0.10						
yrb_Latn	**0.19**	0.10	0.11	0.05	0.06						
yre_Latn	0.08	**0.11**	0.05	0.05	0.05						
yss_Latn	0.10	**0.12**	0.08	0.05	0.08						
yua_Latn	0.16	**0.16**	0.11	0.05	0.13						
yue_Hani	0.07	0.40	**0.60**	0.60	0.56						
yuj_Latn	**0.14**	0.08	0.09	0.06	0.07						
yut_Latn	0.11	**0.14**	0.05	0.05	0.05						
yuw_Latn	0.10	**0.12**	0.09	0.05	0.05						
yuz_Latn	0.07	**0.12**	0.10	0.05	0.10						
yva_Latn	0.13	**0.15**	0.06	0.05	0.06						
zaa_Latn	0.10	0.20	0.20	0.07	**0.29**						
zab_Latn	0.07	0.08	0.13	0.07	**0.16**						

Table 11: Zero-shot performance of BOW, mBERT, XLM-R Base, XLM-R Large, and Glot500-m on Taxi1500.

Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP
ace_Latn	43.8	**49.4**	ach_Latn	37.6	**40.6**	acr_Latn	17.6	**18.6**	afr_Latn	**74.2**	72.4
agw_Latn	31.0	**38.2**	ahk_Latn	3.4	**3.8**	aka_Latn	41.8	**48.4**	aln_Latn	**70.0**	**70.0**
als_Latn	**54.4**	**54.4**	alt_Cyrl	53.8	**57.0**	alz_Latn	36.2	**37.4**	amh_Ethi	44.4	**51.2**
aoj_Latn	15.6	**18.6**	arb_Arab	9.6	**11.6**	arn_Latn	18.2	**23.0**	ary_Arab	11.2	**13.0**
arz_Arab	**15.2**	**15.2**	asm_Beng	**59.2**	59.0	ayr_Latn	37.6	**46.0**	azb_Arab	55.6	**59.0**
aze_Latn	73.4	**75.4**	bak_Cyrl	58.8	**62.2**	bam_Latn	38.4	**44.8**	ban_Latn	33.0	**33.2**
bar_Latn	32.2	**34.0**	bba_Latn	26.2	**31.0**	bbc_Latn	**60.8**	58.8	bci_Latn	**12.0**	11.8
bcl_Latn	75.4	**79.0**	bel_Cyrl	**70.6**	69.6	bem_Latn	51.0	**54.4**	ben_Beng	53.4	**55.4**
bhw_Latn	28.4	**30.6**	bim_Latn	31.4	**42.8**	bis_Latn	45.2	**50.8**	bod_Tibt	29.6	**33.6**
bqc_Latn	27.4	**29.2**	bre_Latn	**31.8**	30.0	bts_Latn	**62.4**	62.0	btx_Latn	**57.2**	55.8
bul_Cyrl	79.8	**80.0**	bum_Latn	32.8	**35.2**	bzj_Latn	69.8	**70.2**	cab_Latn	11.6	**11.8**
cac_Latn	10.8	**11.8**	cak_Latn	**17.8**	16.6	caq_Latn	26.0	**29.8**	cat_Latn	**85.4**	83.2
cbk_Latn	54.8	**56.2**	cce_Latn	41.8	**45.4**	ceb_Latn	70.4	**70.6**	ces_Latn	**68.2**	67.0
cfm_Latn	34.4	**38.8**	che_Cyrl	10.2	**11.2**	chk_Latn	35.2	**43.0**	chv_Cyrl	45.0	**54.4**
ckb_Arab	31.2	**32.8**	cmn_Hani	**41.4**	40.8	cnh_Latn	38.2	**43.2**	crh_Cyrl	67.2	**70.0**
crs_Latn	**85.6**	84.4	csy_Latn	40.2	**49.6**	ctd_Latn	44.4	**50.6**	ctu_Latn	**16.6**	16.0
cuk_Latn	**17.0**	**17.0**	cym_Latn	**45.6**	43.8	dan_Latn	**72.4**	71.8	deu_Latn	73.8	**74.0**
djk_Latn	**38.0**	**38.0**	dln_Latn	46.6	**51.4**	dtp_Latn	17.0	**17.8**	dyu_Latn	33.0	**40.2**
dzo_Tibt	28.4	**33.0**	efi_Latn	41.6	**53.6**	ell_Grek	48.2	**49.2**	enm_Latn	**69.4**	**69.4**
epo_Latn	**67.4**	65.8	est_Latn	**66.4**	66.0	eus_Latn	23.8	**24.2**	ewe_Latn	33.2	**34.8**
fao_Latn	**79.8**	78.4	fas_Arab	80.2	**84.2**	fij_Latn	30.0	**31.0**	fil_Latn	**77.6**	77.2
fin_Latn	65.4	**66.0**	fon_Latn	20.2	**25.2**	fra_Latn	**87.4**	87.2	fry_Latn	**47.0**	44.0
gaa_Latn	34.4	**40.6**	gil_Latn	30.0	**31.6**	giz_Latn	32.4	**36.4**	gkn_Latn	20.4	**24.2**
gkp_Latn	13.2	**14.6**	gla_Latn	**39.0**	38.0	gle_Latn	**41.2**	38.4	glv_Latn	37.2	**38.6**
gom_Latn	33.2	**36.0**	gor_Latn	21.8	**23.0**	grc_Grek	44.4	**47.0**	guc_Latn	**9.8**	8.2
gug_Latn	28.2	**31.2**	guj_Gujr	**69.8**	67.6	gur_Latn	17.6	**18.2**	guw_Latn	36.8	**45.4**
gya_Latn	27.6	**32.6**	gym_Latn	**13.6**	13.0	hat_Latn	**76.4**	74.6	hau_Latn	57.6	**59.6**
haw_Latn	28.0	**30.4**	heb_Hebr	21.6	**23.0**	hif_Latn	33.2	**34.6**	hil_Latn	74.0	**79.8**
hin_Deva	**75.6**	74.6	hin_Latn	34.2	**36.2**	hmo_Latn	44.2	**57.0**	hne_Deva	71.6	**73.6**
hnj_Latn	39.6	**46.6**	hra_Latn	43.4	**46.4**	hrv_Latn	**80.4**	79.8	hui_Latn	19.8	**22.0**
hun_Latn	65.6	**69.0**	hus_Latn	14.8	**16.2**	hye_Armn	62.8	**65.6**	iba_Latn	70.2	**71.6**
ibo_Latn	**32.4**	31.6	ifa_Latn	26.2	**29.0**	ifb_Latn	**28.6**	**28.6**	ikk_Latn	30.2	**46.4**
ilo_Latn	53.4	**54.4**	ind_Latn	78.4	**78.6**	isl_Latn	71.0	**71.8**	ita_Latn	76.2	**76.8**
ium_Latn	20.0	**23.2**	ixl_Latn	13.8	**14.4**	izz_Latn	19.6	**22.6**	jam_Latn	**61.0**	59.2
jav_Latn	**55.4**	52.0	jpn_Jpan	65.8	**67.6**	kaa_Cyrl	71.2	**75.0**	kaa_Latn	32.0	**37.6**
kab_Latn	12.2	**13.4**	kac_Latn	22.2	**27.0**	kal_Latn	12.6	**16.8**	kan_Knda	50.0	**52.8**
kat_Geor	49.6	**52.4**	kaz_Cyrl	69.4	**70.4**	kbp_Latn	21.8	**26.8**	kek_Latn	16.6	**18.6**
khm_Khmr	39.4	**43.0**	kia_Latn	24.6	**28.8**	kik_Latn	44.4	**48.4**	kin_Latn	56.6	**60.2**
kir_Cyrl	69.8	**70.2**	kjb_Latn	23.4	**26.0**	kjh_Cyrl	45.6	**50.6**	kmm_Latn	33.8	**38.0**
kmr_Cyrl	**42.0**	40.2	kmr_Latn	60.2	**60.4**	knv_Latn	7.0	**8.4**	kor_Hang	60.8	**64.0**
kpg_Latn	42.6	**48.8**	krc_Cyrl	59.8	**62.2**	kri_Latn	61.4	**62.6**	ksd_Latn	31.4	**41.0**
kss_Latn	5.2	**6.0**	ksw_Mymr	26.2	**28.0**	kua_Latn	43.0	**43.8**	lam_Latn	20.4	**22.8**
lao_Laoo	41.6	**47.2**	lat_Latn	56.6	**58.0**	lav_Latn	69.8	**71.2**	ldi_Latn	**22.4**	22.0
leh_Latn	**46.8**	45.8	lhu_Latn	**4.4**	4.2	lin_Latn	64.6	**71.0**	lit_Latn	**67.0**	66.6
loz_Latn	**46.8**	45.6	ltz_Latn	**63.8**	63.2	lug_Latn	37.2	**40.8**	luo_Latn	**42.8**	42.6
lus_Latn	46.6	**53.2**	lzh_Hani	59.8	**62.4**	mad_Latn	42.6	**44.6**	mah_Latn	30.4	**33.8**
mai_Deva	52.6	**56.0**	mal_Mlym	51.6	**57.4**	mam_Latn	**10.2**	**10.2**	mar_Deva	68.4	**71.4**
mau_Latn	2.8	**3.4**	mbb_Latn	22.0	**29.8**	mck_Latn	**55.6**	53.4	mcn_Latn	34.2	**40.8**
mco_Latn	**6.6**	6.4	mdy_Ethi	21.4	**30.6**	meu_Latn	48.8	**52.0**	mfe_Latn	**77.4**	**77.4**

Table 12: Top-10 accuracy of models on **SR-B** (Part I).

Language	Baseline	LangSAMP	Language	Baseline	LangSAMP	Language	Baseline	LangSAMP	Language	Baseline	LangSAMP
mgh_Latn	17.4	**20.8**	mgr_Latn	**48.6**	47.2	mhr_Cyrl	37.4	**43.2**	min_Latn	**32.4**	29.6
miq_Latn	28.8	**36.8**	mkd_Cyrl	78.4	**78.8**	mlg_Latn	60.2	**61.2**	mlt_Latn	48.0	**50.4**
mos_Latn	32.2	**32.8**	mps_Latn	16.4	**20.6**	mri_Latn	45.6	**55.0**	mrw_Latn	34.0	**40.6**
msa_Latn	43.6	**44.2**	mwm_Latn	24.0	**25.6**	mxv_Latn	**7.0**	**7.0**	mya_Mymr	25.8	**28.0**
myv_Cyrl	26.6	**30.6**	mzh_Latn	24.6	**25.4**	nan_Latn	13.2	**13.6**	naq_Latn	16.8	**26.8**
nav_Latn	**8.6**	**8.6**	nbl_Latn	**49.4**	48.4	nch_Latn	**21.6**	**21.6**	ncj_Latn	18.8	**19.4**
ndc_Latn	32.4	**36.2**	nde_Latn	51.0	**54.8**	ndo_Latn	41.0	**44.0**	nds_Latn	**38.4**	**38.4**
nep_Deva	56.4	**59.0**	ngu_Latn	**26.2**	26.0	nia_Latn	25.6	**28.0**	nld_Latn	**78.4**	78.0
nmf_Latn	25.6	**28.2**	nnb_Latn	33.2	**38.8**	nno_Latn	**76.8**	75.8	nob_Latn	**85.4**	85.0
nor_Latn	**85.8**	83.4	npi_Deva	77.4	**80.8**	nse_Latn	48.4	**51.8**	nso_Latn	46.2	**50.2**
nya_Latn	**57.6**	**57.6**	nyn_Latn	**48.8**	47.4	nyy_Latn	23.4	**24.6**	nzi_Latn	29.2	**34.4**
ori_Orya	51.2	**53.4**	ory_Orya	46.4	**49.8**	oss_Cyrl	41.4	**56.4**	ote_Latn	12.0	**13.2**
pag_Latn	**55.2**	52.2	pam_Latn	37.4	**41.2**	pan_Guru	**46.2**	45.4	pap_Latn	72.8	**75.0**
pau_Latn	17.0	**23.4**	pcm_Latn	**69.8**	69.4	pdt_Latn	**69.4**	66.0	pes_Arab	74.2	**75.2**
pis_Latn	51.4	**54.8**	pls_Latn	27.0	**31.8**	plt_Latn	60.2	**60.8**	poh_Latn	10.6	**11.4**
pol_Latn	73.8	**75.6**	pon_Latn	21.4	**24.0**	por_Latn	**81.8**	81.0	prk_Latn	42.0	**47.4**
prs_Arab	84.6	**87.0**	pxm_Latn	18.2	**19.8**	qub_Latn	30.6	**35.6**	quc_Latn	**18.6**	17.4
qug_Latn	53.6	**59.2**	quh_Latn	40.2	**43.8**	quw_Latn	46.2	**50.4**	quy_Latn	47.4	**54.4**
quz_Latn	59.4	**63.6**	qvi_Latn	49.2	**57.6**	rap_Latn	17.0	**17.8**	rar_Latn	**20.4**	19.8
rmy_Latn	30.4	**32.2**	ron_Latn	**69.4**	69.0	rop_Latn	35.8	**41.4**	rug_Latn	37.8	**38.4**
run_Latn	48.2	**52.4**	rus_Cyrl	74.6	**76.4**	sag_Latn	39.6	**45.4**	sah_Cyrl	43.4	**45.8**
san_Deva	**24.2**	23.6	san_Latn	**7.8**	7.4	sba_Latn	28.0	**29.2**	seh_Latn	67.4	**69.4**
sin_Sinh	45.6	**49.0**	slk_Latn	**69.8**	69.2	slv_Latn	**61.2**	60.8	sme_Latn	35.0	**37.6**
smo_Latn	27.6	**28.8**	sna_Latn	38.4	**41.2**	snd_Arab	**67.2**	65.0	som_Latn	**35.0**	34.8
sop_Latn	**32.4**	28.8	sot_Latn	48.4	**52.4**	spa_Latn	80.8	**81.4**	sqi_Latn	62.2	**64.8**
srm_Latn	**28.2**	26.6	srn_Latn	75.4	**75.6**	srp_Cyrl	**87.2**	85.8	srp_Latn	**85.8**	85.4
ssw_Latn	42.8	**47.0**	sun_Latn	52.0	**54.0**	suz_Deva	21.0	**22.6**	swe_Latn	**78.6**	77.0
swh_Latn	**71.6**	71.4	sxn_Latn	20.6	**20.8**	tam_Taml	47.0	**50.6**	tat_Cyrl	68.2	**70.4**
tbz_Latn	13.2	**18.2**	tca_Latn	10.0	**13.8**	tdt_Latn	50.0	**53.6**	tel_Telu	48.0	**50.2**
teo_Latn	19.4	**19.6**	tgk_Cyrl	69.2	**69.4**	tgl_Latn	**79.6**	78.0	tha_Thai	33.8	**38.0**
tih_Latn	42.2	**46.4**	tir_Ethi	32.2	**34.8**	tlh_Latn	62.0	**66.4**	tob_Latn	**11.6**	11.4
toh_Latn	36.8	**41.8**	toi_Latn	**39.4**	**39.4**	toj_Latn	**14.8**	12.6	ton_Latn	16.0	**16.6**
top_Latn	**6.6**	6.0	tpi_Latn	58.0	**62.2**	tpm_Latn	**27.4**	23.0	tsn_Latn	32.6	**34.6**
tso_Latn	50.0	**51.0**	tsz_Latn	21.2	**25.8**	tuc_Latn	25.6	**32.4**	tui_Latn	29.8	**31.0**
tuk_Cyrl	67.4	**69.4**	tuk_Latn	67.6	**70.0**	tum_Latn	**58.4**	57.0	tur_Latn	70.2	**70.4**
twi_Latn	35.0	**42.0**	tyv_Cyrl	**44.2**	43.4	tzh_Latn	19.0	**19.8**	tzo_Latn	**14.2**	13.6
udm_Cyrl	41.6	**45.2**	uig_Arab	47.4	**50.8**	uig_Latn	57.2	**58.8**	ukr_Cyrl	67.0	**68.0**
urd_Arab	60.4	**61.4**	uzb_Cyrl	80.6	**81.2**	uzb_Latn	**70.0**	68.2	uzn_Cyrl	82.4	**83.0**
ven_Latn	37.2	**42.0**	vie_Latn	68.0	**69.4**	wal_Latn	35.0	**43.4**	war_Latn	42.6	**44.0**
wbm_Latn	37.6	**46.2**	wol_Latn	31.8	**33.2**	xav_Latn	3.8	**4.0**	xho_Latn	42.6	**44.2**
yan_Latn	16.4	**27.2**	yao_Latn	37.4	**37.6**	yap_Latn	15.8	**19.6**	yom_Latn	37.6	**40.0**
yor_Latn	27.4	**28.8**	yua_Latn	**13.2**	12.8	yue_Hani	**17.2**	**17.2**	zai_Latn	29.0	**30.6**
zho_Hani	41.6	**41.8**	zlm_Latn	**84.8**	**84.8**	zom_Latn	39.6	**45.0**	zsm_Latn	90.0	**91.0**

Table 13: Top-10 accuracy of models on **SR-B** (Part II).

Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP
afr_Latn	77.9	**80.4**	amh_Ethi	47.0	**52.4**	ara_Arab	**69.4**	68.7	arz_Arab	61.8	**63.9**
ast_Latn	80.3	**84.3**	aze_Latn	82.6	**84.1**	bel_Cyrl	**83.6**	83.0	ben_Beng	72.1	**74.9**
bos_Latn	90.1	**90.4**	bre_Latn	17.4	**18.2**	bul_Cyrl	87.5	**89.2**	cat_Latn	78.2	**78.6**
cbk_Latn	**49.4**	48.0	ceb_Latn	39.0	**42.5**	ces_Latn	**75.7**	73.5	cmn_Hani	87.1	**87.4**
csb_Latn	38.3	**38.7**	cym_Latn	52.2	**55.0**	dan_Latn	91.7	**92.9**	deu_Latn	95.5	**95.7**
dtp_Latn	17.0	**19.3**	ell_Grek	79.3	**82.7**	epo_Latn	71.8	**74.8**	est_Latn	68.2	**69.9**
eus_Latn	52.2	**55.4**	fao_Latn	**77.1**	75.6	fin_Latn	72.3	**74.2**	fra_Latn	**85.3**	85.2
fry_Latn	75.1	**79.2**	gla_Latn	38.4	**38.6**	gle_Latn	44.8	**48.3**	glg_Latn	**77.1**	76.4
gsw_Latn	58.1	**63.2**	heb_Hebr	71.4	**74.9**	hin_Deva	**88.1**	87.3	hrv_Latn	**87.9**	87.5
hsb_Latn	**49.7**	**49.7**	hun_Latn	71.5	**73.2**	hye_Armn	79.1	**81.3**	ido_Latn	54.6	**55.8**
ile_Latn	71.2	**71.5**	ina_Latn	89.2	**90.7**	ind_Latn	88.1	**88.9**	isl_Latn	84.0	**84.5**
ita_Latn	84.1	**85.7**	jpn_Jpan	**77.2**	77.1	kab_Latn	10.8	**11.0**	kat_Geor	71.2	**72.4**
kaz_Cyrl	74.6	**77.7**	khm_Khmr	57.5	**63.0**	kor_Hang	80.8	**81.1**	kur_Latn	49.8	**52.4**
lat_Latn	39.2	**42.1**	lfn_Latn	55.8	**56.8**	lit_Latn	70.4	**72.9**	lvs_Latn	76.2	**78.1**
mal_Mlym	87.5	**91.6**	mar_Deva	79.8	**81.6**	mhr_Cyrl	27.7	**33.4**	mkd_Cyrl	**79.6**	79.4
mon_Cyrl	78.2	**80.5**	nds_Latn	71.3	**72.5**	nld_Latn	92.4	**93.4**	nno_Latn	85.5	**87.4**
nob_Latn	94.5	**95.3**	oci_Latn	**46.6**	44.9	pam_Latn	**10.2**	**10.2**	pes_Arab	86.7	**86.9**
pms_Latn	49.5	**50.9**	pol_Latn	**84.3**	83.4	por_Latn	90.2	**90.7**	ron_Latn	86.0	**86.9**
rus_Cyrl	91.6	**92.1**	slk_Latn	77.9	**78.2**	slv_Latn	**76.2**	75.9	spa_Latn	**88.6**	88.3
sqi_Latn	84.1	**85.2**	srp_Latn	**89.7**	89.6	swe_Latn	89.4	**89.6**	swh_Latn	**45.1**	44.9
tam_Taml	**50.2**	45.0	tat_Cyrl	71.2	**74.6**	tel_Telu	72.6	**74.8**	tgl_Latn	73.9	**74.2**
tha_Thai	75.4	**79.2**	tuk_Latn	62.1	**68.0**	tur_Latn	79.1	**82.0**	uig_Arab	64.7	**68.4**
ukr_Cyrl	84.9	**86.5**	urd_Arab	78.5	**81.7**	uzb_Cyrl	65.0	**67.3**	vie_Latn	**88.9**	88.8
war_Latn	22.7	**25.2**	wuu_Hani	79.0	**82.4**	xho_Latn	54.9	**56.3**	yid_Hebr	65.8	**67.6**

Table 14: Top-10 accuracy of models on **SR-T**.

Language	Baseline	LangSAMP	Language	Baseline	LangSAMP	Language	Baseline	LangSAMP	Language	Baseline	LangSAMP
ace_Latn	**66.3**	64.2	ach_Latn	35.8	**40.3**	acr_Latn	44.2	**51.0**	afr_Latn	**60.0**	58.8
agw_Latn	51.0	**56.3**	ahk_Latn	**8.0**	6.3	aka_Latn	42.5	**49.0**	aln_Latn	55.3	**58.1**
als_Latn	56.2	**58.1**	alt_Cyrl	47.2	**49.7**	alz_Latn	31.1	**38.5**	amh_Ethi	**8.8**	7.7
aoj_Latn	34.1	**42.6**	arn_Latn	40.9	**44.5**	ary_Arab	32.9	**33.8**	arz_Arab	35.4	**40.8**
asm_Beng	62.5	**64.5**	ayr_Latn	52.7	**57.3**	azb_Arab	**63.5**	62.3	aze_Latn	66.0	**70.2**
bak_Cyrl	59.7	**59.9**	bam_Latn	43.3	**49.1**	ban_Latn	42.5	**48.0**	bar_Latn	44.1	**49.2**
bba_Latn	39.4	**43.4**	bci_Latn	29.6	**33.5**	bcl_Latn	54.0	**63.2**	bel_Cyrl	59.7	**61.4**
bem_Latn	45.7	**50.5**	ben_Beng	61.8	**66.6**	bhw_Latn	44.4	**54.4**	bim_Latn	49.4	**50.2**
bis_Latn	65.8	**71.7**	bqc_Latn	31.6	**37.7**	bre_Latn	35.7	**42.9**	btx_Latn	52.9	**63.9**
bul_Cyrl	64.9	**65.5**	bum_Latn	38.6	**46.9**	bzj_Latn	66.3	**68.1**	cab_Latn	22.9	**31.1**
cac_Latn	42.7	**47.0**	cak_Latn	51.2	**55.2**	caq_Latn	39.7	**45.5**	cat_Latn	**63.4**	62.2
cbk_Latn	62.0	**68.8**	cce_Latn	41.3	**47.8**	ceb_Latn	52.9	**55.5**	ces_Latn	59.7	**66.8**
cfm_Latn	54.5	**65.6**	che_Cyrl	17.3	**23.2**	chv_Cyrl	54.8	**62.2**	cmn_Hani	67.4	**70.2**
cnh_Latn	61.4	**64.6**	crh_Cyrl	60.4	**64.1**	crs_Latn	**65.3**	64.6	csy_Latn	52.4	**64.2**
ctd_Latn	52.5	**59.3**	ctu_Latn	50.3	**51.3**	cuk_Latn	39.1	**43.7**	cym_Latn	**50.0**	49.1
dan_Latn	62.0	**64.2**	deu_Latn	53.0	**56.0**	djk_Latn	46.8	**55.5**	dln_Latn	47.7	**61.7**
dtp_Latn	50.0	**51.3**	dyu_Latn	46.4	**57.7**	dzo_Tibt	55.9	**57.4**	efi_Latn	52.1	**56.9**
ell_Grek	59.6	**62.2**	eng_Latn	74.2	**76.1**	enm_Latn	**72.1**	71.9	epo_Latn	56.0	**58.9**
est_Latn	**56.9**	56.2	eus_Latn	23.2	**25.9**	ewe_Latn	42.7	**52.2**	fao_Latn	56.6	**60.2**
fas_Arab	**72.0**	70.1	fij_Latn	43.7	**48.9**	fil_Latn	56.9	**58.8**	fin_Latn	57.7	**59.5**
fon_Latn	43.0	**44.2**	fra_Latn	64.7	**70.4**	fry_Latn	39.1	**43.2**	gaa_Latn	39.4	**42.4**
gil_Latn	40.9	**44.9**	giz_Latn	41.6	**50.2**	gkn_Latn	37.2	**42.8**	gkp_Latn	31.9	**38.6**
gla_Latn	47.8	**48.8**	gle_Latn	41.6	**42.5**	glv_Latn	37.4	**44.7**	gom_Latn	34.9	**37.9**
gor_Latn	42.6	**50.4**	guc_Latn	32.8	**39.4**	gug_Latn	33.7	**40.9**	guj_Gujr	68.1	**69.5**
gur_Latn	33.7	**43.3**	guw_Latn	48.7	**53.6**	gya_Latn	**40.6**	39.8	gym_Latn	40.4	**47.2**
hat_Latn	62.5	**65.2**	hau_Latn	53.8	**59.1**	haw_Latn	29.2	**39.2**	heb_Hebr	17.9	**20.8**
hif_Latn	44.5	**47.6**	hil_Latn	64.7	**67.7**	hin_Deva	66.0	**69.7**	hmo_Latn	58.4	**65.5**
hne_Deva	65.7	**66.7**	hnj_Latn	63.7	**67.1**	hra_Latn	50.4	**56.1**	hrv_Latn	62.8	**68.0**
hui_Latn	46.0	**51.1**	hun_Latn	63.7	**68.4**	hus_Latn	35.6	**42.2**	hye_Armn	69.7	**71.4**
iba_Latn	57.1	**61.6**	ibo_Latn	56.2	**58.3**	ifa_Latn	46.5	**55.2**	ifb_Latn	48.7	**50.6**
ikk_Latn	46.8	**52.3**	ilo_Latn	49.8	**60.7**	ind_Latn	76.1	**78.3**	isl_Latn	51.2	**58.0**
ita_Latn	63.5	**66.3**	ium_Latn	56.2	**59.4**	ixl_Latn	31.7	**39.6**	izz_Latn	39.4	**48.9**
jam_Latn	63.6	**68.5**	jav_Latn	46.2	**51.6**	jpn_Jpan	63.6	**63.7**	kaa_Cyrl	57.7	**66.8**
kab_Latn	23.3	**30.4**	kac_Latn	**49.2**	45.7	kal_Latn	30.0	**37.2**	kan_Knda	65.6	**65.8**
kat_Geor	**59.6**	57.6	kaz_Cyrl	**64.3**	62.4	kbp_Latn	34.5	**37.4**	kek_Latn	44.5	**46.6**
khm_Khmr	**69.5**	66.2	kia_Latn	40.9	**52.2**	kik_Latn	40.4	**46.7**	kin_Latn	43.9	**56.8**
kir_Cyrl	66.5	**67.7**	kjb_Latn	45.4	**48.5**	kjh_Cyrl	49.9	**55.1**	kmm_Latn	46.3	**57.2**
kmr_Cyrl	50.1	**51.6**	knv_Latn	43.1	**45.1**	kor_Hang	70.3	**72.4**	kpg_Latn	63.9	**65.6**
krc_Cyrl	55.7	**63.0**	kri_Latn	58.8	**64.1**	ksd_Latn	53.3	**53.5**	kss_Latn	**21.8**	17.9
ksw_Mymr	47.7	**50.0**	kua_Latn	41.0	**45.9**	lam_Latn	31.9	**38.0**	lao_Laoo	**71.9**	70.5
lat_Latn	57.0	**64.0**	lav_Latn	62.5	**64.8**	ldi_Latn	26.7	**34.8**	leh_Latn	44.4	**48.3**
lhu_Latn	22.7	**27.3**	lin_Latn	47.3	**55.5**	lit_Latn	61.1	**61.8**	loz_Latn	49.2	**49.8**
ltz_Latn	**53.3**	52.1	lug_Latn	41.9	**52.6**	luo_Latn	36.8	**44.8**	lus_Latn	47.5	**54.8**
lzh_Hani	61.1	**68.5**	mad_Latn	59.4	**63.0**	mah_Latn	33.8	**45.2**	mai_Deva	**64.1**	63.4
mal_Mlym	**7.1**	6.1	mam_Latn	27.6	**34.8**	mar_Deva	60.8	**61.9**	mau_Latn	**6.9**	5.9
mbb_Latn	52.2	**55.2**	mck_Latn	40.7	**46.2**	mcn_Latn	35.1	**44.2**	mco_Latn	21.9	**26.2**
mdy_Ethi	48.5	**54.5**	meu_Latn	46.9	**57.9**	mfe_Latn	68.4	**69.9**	mgh_Latn	31.2	**33.6**
mgr_Latn	45.9	**48.4**	mhr_Cyrl	40.9	**41.0**	min_Latn	50.3	**53.7**	miq_Latn	51.0	**54.2**
mkd_Cyrl	68.7	**72.9**	mlg_Latn	47.0	**51.7**	mlt_Latn	49.0	**53.5**	mos_Latn	35.8	**44.6**

Table 15: F_1 scores of models on **Taxi1500** (Part I).

Language	Baseline	LangSAMP	Language	Baseline	LangSAMP	Language	Baseline	LangSAMP	Language	Baseline	LangSAMP
mps_Latn	51.3	**56.3**	mri_Latn	41.5	**49.2**	mrw_Latn	47.8	**48.5**	msa_Latn	46.0	**49.0**
mwm_Latn	51.3	**57.8**	mxv_Latn	14.3	**27.9**	mya_Mymr	56.6	**57.8**	myv_Cyrl	41.3	**47.8**
mzh_Latn	39.1	**42.7**	nan_Latn	25.5	**32.3**	naq_Latn	39.0	**45.6**	nav_Latn	21.6	**25.8**
nbl_Latn	46.0	**52.3**	nch_Latn	40.9	**46.1**	ncj_Latn	34.5	**41.7**	ndc_Latn	38.2	**43.7**
nde_Latn	46.0	**52.3**	ndo_Latn	45.4	**50.7**	nds_Latn	39.5	**47.2**	nep_Deva	70.3	**72.8**
ngu_Latn	42.1	**44.0**	nld_Latn	61.7	**61.9**	nmf_Latn	40.8	**47.5**	nnb_Latn	36.7	**45.9**
nno_Latn	62.3	**66.4**	nob_Latn	59.3	**60.6**	nor_Latn	61.2	**61.4**	npi_Deva	70.3	**70.6**
nse_Latn	42.7	**45.6**	nso_Latn	**53.2**	52.4	nya_Latn	54.2	**61.6**	nyn_Latn	41.6	**47.3**
nyy_Latn	30.7	**38.1**	nzi_Latn	34.6	**37.6**	ori_Orya	**69.8**	69.5	ory_Orya	**70.8**	69.0
oss_Cyrl	46.7	**57.3**	ote_Latn	**35.5**	35.4	pag_Latn	50.1	**54.7**	pam_Latn	38.8	**46.0**
pan_Guru	**66.8**	65.4	pap_Latn	65.7	**66.5**	pau_Latn	41.4	**43.9**	pcm_Latn	63.3	**67.7**
pdt_Latn	58.1	**58.7**	pes_Arab	**70.3**	69.9	pis_Latn	66.5	**67.9**	pls_Latn	45.5	**50.3**
plt_Latn	**52.3**	50.7	poh_Latn	47.5	**49.4**	pol_Latn	64.4	**68.6**	pon_Latn	52.8	**53.2**
por_Latn	67.3	**72.5**	prk_Latn	55.6	**56.8**	prs_Arab	68.1	**69.9**	pxm_Latn	40.5	**41.3**
qub_Latn	56.7	**59.1**	quc_Latn	50.0	**54.0**	qug_Latn	62.1	**68.0**	quh_Latn	61.4	**68.9**
quw_Latn	52.0	**56.1**	quy_Latn	70.7	**71.1**	quz_Latn	63.8	**67.2**	qvi_Latn	61.3	**64.0**
rap_Latn	47.2	**48.4**	rar_Latn	45.6	**53.8**	rmy_Latn	44.6	**48.1**	ron_Latn	60.2	**67.6**
rop_Latn	56.0	**57.6**	rug_Latn	50.0	**55.4**	run_Latn	49.5	**54.1**	rus_Cyrl	69.3	**72.9**
sag_Latn	43.9	**46.5**	sah_Cyrl	58.5	**62.8**	sba_Latn	36.7	**41.6**	seh_Latn	46.8	**49.4**
sin_Sinh	66.2	**66.5**	slk_Latn	59.2	**60.9**	slv_Latn	61.5	**63.2**	sme_Latn	34.8	**48.0**
smo_Latn	53.5	**61.2**	sna_Latn	39.5	**45.4**	snd_Arab	67.3	**68.8**	som_Latn	31.9	**36.5**
sop_Latn	32.2	**40.2**	sot_Latn	43.9	**48.1**	spa_Latn	64.3	**68.2**	sqi_Latn	71.3	**72.1**
srm_Latn	47.6	**53.4**	srn_Latn	63.1	**65.7**	srp_Latn	64.3	**70.7**	ssw_Latn	36.6	**47.4**
sun_Latn	53.7	**56.3**	suz_Deva	57.6	**61.0**	swe_Latn	67.5	**69.9**	swh_Latn	61.0	**64.6**
sxn_Latn	46.7	**51.8**	tam_Taml	72.2	**74.3**	tat_Cyrl	64.2	**67.5**	tbz_Latn	35.1	**44.2**
tca_Latn	41.0	**49.2**	tdt_Latn	58.6	**66.6**	tel_Telu	69.8	**72.1**	teo_Latn	23.1	**26.5**
tgk_Cyrl	63.9	**66.3**	tgl_Latn	56.9	**58.8**	tha_Thai	65.2	**66.8**	tih_Latn	56.6	**60.5**
tir_Ethi	49.3	**52.2**	tlh_Latn	62.2	**66.2**	tob_Latn	40.6	**44.6**	toh_Latn	37.3	**41.7**
toi_Latn	39.4	**49.2**	toj_Latn	35.7	**40.2**	ton_Latn	46.9	**49.8**	top_Latn	21.2	**26.0**
tpi_Latn	68.4	**69.5**	tpm_Latn	43.2	**52.8**	tsn_Latn	44.2	**45.0**	tsz_Latn	35.9	**42.9**
tuc_Latn	55.5	**61.4**	tui_Latn	44.8	**47.5**	tuk_Latn	55.9	**63.0**	tum_Latn	47.9	**50.5**
tur_Latn	61.3	**67.2**	twi_Latn	40.4	**49.2**	tyv_Cyrl	56.8	**62.7**	tzh_Latn	37.9	**44.5**
tzo_Latn	37.4	**42.9**	udm_Cyrl	53.1	**54.0**	ukr_Cyrl	63.9	**69.2**	urd_Arab	**60.6**	59.7
uzb_Latn	57.6	**58.7**	uzn_Cyrl	64.3	**66.7**	ven_Latn	42.6	**46.1**	vie_Latn	69.6	**70.0**
wal_Latn	41.1	**50.4**	war_Latn	43.3	**51.1**	wbm_Latn	56.1	**56.6**	wol_Latn	32.3	**40.6**
xav_Latn	28.0	**33.6**	xho_Latn	44.5	**50.1**	yan_Latn	50.1	**52.1**	yao_Latn	38.9	**46.8**
yap_Latn	37.5	**40.5**	yom_Latn	35.4	**39.5**	yor_Latn	46.0	**48.4**	yua_Latn	35.7	**39.9**
yue_Hani	57.7	**60.2**	zai_Latn	38.5	**44.5**	zho_Hani	64.2	**67.7**	zlm_Latn	**69.4**	69.2

Table 16: F_1 scores of models on **Taxi1500** (Part II).

Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP
ace_Latn	71.5	**73.6**	acm_Arab	82.2	**83.0**	afr_Latn	82.3	**82.7**	ajp_Arab	**83.4**	81.8
aka_Latn	62.2	**67.2**	als_Latn	82.4	**84.4**	amh_Ethi	**74.2**	73.6	apc_Arab	**83.9**	82.9
arb_Arab	**83.8**	82.9	ary_Arab	**81.5**	80.2	arz_Arab	**84.5**	84.1	asm_Beng	83.6	**84.2**
ast_Latn	**88.4**	88.0	ayr_Latn	51.1	**53.8**	azb_Arab	71.5	**74.7**	azj_Latn	87.0	**88.0**
bak_Cyrl	84.6	**86.6**	bam_Latn	**47.9**	47.6	ban_Latn	80.3	**83.0**	bel_Cyrl	**83.7**	83.4
bem_Latn	63.0	**63.9**	ben_Beng	83.3	**84.3**	bjn_Latn	77.1	**78.5**	bod_Tibt	**73.5**	69.2
bos_Latn	86.5	**88.2**	bul_Cyrl	86.1	**87.5**	cat_Latn	84.8	**86.4**	ceb_Latn	81.8	**84.6**
ces_Latn	**89.1**	86.9	cjk_Latn	46.6	**48.1**	ckb_Arab	**83.9**	80.2	crh_Latn	74.0	**76.2**
cym_Latn	**75.9**	75.4	dan_Latn	86.8	**87.4**	deu_Latn	86.5	**87.8**	dyu_Latn	42.6	**44.5**
dzo_Tibt	68.7	**72.6**	ell_Grek	79.5	**80.0**	eng_Latn	**90.8**	90.0	epo_Latn	**83.8**	82.2
est_Latn	80.6	**81.6**	eus_Latn	82.1	**82.2**	ewe_Latn	49.3	**51.5**	fao_Latn	83.7	**84.9**
fij_Latn	56.1	**58.0**	fin_Latn	82.1	**82.9**	fon_Latn	41.7	**44.6**	fra_Latn	87.9	**89.6**
fur_Latn	77.6	**80.2**	gla_Latn	**57.6**	54.3	gle_Latn	62.2	**64.1**	glg_Latn	87.8	**89.0**
grn_Latn	**75.0**	74.5	guj_Gujr	83.9	**84.7**	hat_Latn	77.4	**79.1**	hau_Latn	**62.7**	62.1
heb_Hebr	77.9	**79.2**	hin_Deva	84.1	**84.4**	hne_Deva	77.9	**80.1**	hrv_Latn	87.3	**89.0**
hun_Latn	86.8	**87.6**	hye_Armn	**83.0**	82.5	ibo_Latn	72.3	**74.1**	ilo_Latn	75.8	**79.6**
ind_Latn	88.7	**89.1**	isl_Latn	78.5	**79.1**	ita_Latn	87.7	**89.2**	jav_Latn	80.2	**80.3**
jpn_Jpan	87.1	**87.9**	kab_Latn	31.1	**36.9**	kac_Latn	49.3	**52.3**	kam_Latn	49.1	**49.5**
kan_Knda	**83.2**	82.0	kat_Geor	81.8	**83.7**	kaz_Cyrl	84.2	**84.9**	kbp_Latn	**45.1**	44.2
kea_Latn	75.4	**77.0**	khm_Khmr	84.3	**84.4**	kik_Latn	57.1	**59.9**	kin_Latn	69.5	**70.5**
kir_Cyrl	**80.7**	80.3	kmb_Latn	48.2	**49.5**	kmr_Latn	**70.7**	70.0	kon_Latn	65.3	**69.2**
kor_Hang	**85.2**	83.9	lao_Laoo	**85.1**	84.2	lij_Latn	77.7	**79.6**	lim_Latn	74.7	**75.2**
lin_Latn	69.3	**71.4**	lit_Latn	**86.5**	84.7	lmo_Latn	77.7	**79.1**	ltz_Latn	76.6	**79.1**
lua_Latn	**59.1**	56.4	lug_Latn	55.5	**59.1**	luo_Latn	52.6	**53.0**	lus_Latn	65.3	**67.9**
lvs_Latn	**84.4**	83.6	mai_Deva	83.4	**84.0**	mal_Mlym	**80.6**	79.9	mar_Deva	**84.1**	82.5
min_Latn	77.7	**79.6**	mkd_Cyrl	83.3	**84.6**	mlt_Latn	82.9	**83.0**	mos_Latn	44.9	**46.6**
mri_Latn	54.4	**59.3**	mya_Mymr	80.1	**81.6**	nld_Latn	**86.5**	85.8	nno_Latn	**86.6**	86.4
nob_Latn	85.8	**86.1**	npi_Deva	**86.8**	86.0	nso_Latn	61.3	**61.9**	nya_Latn	71.1	**72.7**
oci_Latn	83.1	**84.9**	ory_Orya	79.7	**80.3**	pag_Latn	78.7	**79.7**	pan_Guru	77.4	**79.0**
pap_Latn	77.2	**79.0**	pes_Arab	87.6	**89.2**	plt_Latn	68.4	**68.5**	pol_Latn	86.4	**86.7**
por_Latn	87.3	**88.6**	prs_Arab	85.8	**88.4**	quy_Latn	63.7	**64.0**	ron_Latn	**86.4**	84.5
run_Latn	**68.3**	67.2	rus_Cyrl	87.6	**87.9**	sag_Latn	52.4	**55.1**	san_Deva	**77.9**	77.8
sat_Olck	53.0	**57.4**	scn_Latn	77.6	**78.2**	sin_Sinh	**84.5**	84.1	slk_Latn	86.1	**87.0**
slv_Latn	**86.4**	85.5	smo_Latn	73.4	**74.1**	sna_Latn	**59.3**	58.0	snd_Arab	72.1	**76.9**
som_Latn	**61.8**	59.8	sot_Latn	65.3	**67.6**	spa_Latn	**86.4**	86.2	srd_Latn	74.0	**75.8**
srp_Cyrl	**85.8**	85.2	ssw_Latn	67.5	**68.1**	sun_Latn	84.0	**85.2**	swe_Latn	86.6	**87.3**
swh_Latn	76.0	**78.6**	szl_Latn	74.3	**75.5**	tam_Taml	80.6	**84.3**	tat_Cyrl	84.0	**85.2**
tel_Telu	85.3	**85.7**	tgk_Cyrl	**81.6**	80.9	tgl_Latn	81.9	**83.0**	tha_Thai	87.4	**88.9**
tir_Ethi	59.9	**61.4**	tpi_Latn	80.6	**82.3**	tsn_Latn	**59.1**	55.2	tso_Latn	59.3	**61.2**
tuk_Latn	**78.3**	78.2	tum_Latn	70.3	**70.8**	tur_Latn	82.9	**83.6**	twi_Latn	61.4	**68.0**
uig_Arab	77.7	**80.0**	ukr_Cyrl	**84.7**	84.5	umb_Latn	**45.9**	45.8	urd_Arab	81.3	**81.9**
vec_Latn	**82.0**	81.1	vie_Latn	84.9	**85.8**	war_Latn	81.7	**83.4**	wol_Latn	49.2	**52.1**
xho_Latn	62.4	**64.0**	yor_Latn	46.6	**51.8**	zsm_Latn	**87.2**	86.6	zul_Latn	**73.8**	73.6

Table 17: F_1 scores of models on **SIB200**.

Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP
ace_Latn	41.8	**42.2**	afr_Latn	76.5	**77.4**	als_Latn	**82.4**	**82.4**	amh_Ethi	**48.9**	41.0
ara_Arab	**57.1**	54.4	arg_Latn	78.0	**82.2**	arz_Arab	55.6	**57.5**	asm_Beng	65.8	**68.1**
ast_Latn	83.0	**84.9**	aym_Latn	**45.9**	44.3	aze_Latn	63.3	**66.0**	bak_Cyrl	60.4	**62.3**
bar_Latn	68.6	**70.1**	bel_Cyrl	**74.6**	**74.6**	ben_Beng	**72.7**	71.7	bih_Deva	**56.2**	55.3
bod_Tibt	18.1	**38.6**	bos_Latn	72.1	**73.8**	bre_Latn	63.3	**64.3**	bul_Cyrl	**75.0**	74.5
cat_Latn	83.3	**84.1**	cbk_Latn	**53.8**	52.5	ceb_Latn	53.8	**56.7**	ces_Latn	78.6	**78.7**
che_Cyrl	25.3	**56.5**	chv_Cyrl	**80.0**	73.5	ckb_Arab	72.9	**74.4**	cos_Latn	55.6	**57.0**
crh_Latn	**51.0**	49.0	csb_Latn	58.5	**60.6**	cym_Latn	**63.7**	59.6	dan_Latn	81.1	**81.6**
deu_Latn	76.5	**76.8**	diq_Latn	**55.2**	54.1	div_Thaa	43.0	**53.2**	ell_Grek	73.2	**74.1**
eml_Latn	42.3	**42.9**	eng_Latn	**83.7**	83.3	epo_Latn	67.5	**71.4**	est_Latn	72.3	**74.8**
eus_Latn	56.4	**57.0**	ext_Latn	45.1	**49.8**	fao_Latn	**71.1**	69.0	fas_Arab	**51.8**	50.0
fin_Latn	75.0	**75.2**	fra_Latn	76.4	**77.6**	frr_Latn	**55.9**	54.8	fry_Latn	**77.4**	77.2
fur_Latn	55.3	**55.7**	gla_Latn	59.8	**64.7**	gle_Latn	72.8	**72.9**	glg_Latn	80.1	**81.5**
grn_Latn	**56.0**	55.7	guj_Gujr	54.3	**58.9**	hbs_Latn	62.6	**63.8**	heb_Hebr	49.3	**50.7**
hin_Deva	69.3	**69.5**	hrv_Latn	77.3	**77.8**	hsb_Latn	73.6	**73.8**	hun_Latn	76.0	**77.4**
hye_Armn	**55.9**	55.4	ibo_Latn	**59.1**	55.2	ido_Latn	**81.9**	79.7	ilo_Latn	72.7	**74.7**
ina_Latn	58.0	**58.4**	ind_Latn	**64.7**	62.1	isl_Latn	**72.4**	71.6	ita_Latn	77.9	**79.2**
jav_Latn	**56.1**	54.9	jbo_Latn	**22.9**	**22.9**	jpn_Jpan	**21.3**	15.3	kan_Knda	58.2	**63.4**
kat_Geor	67.4	**67.8**	kaz_Cyrl	50.8	**50.9**	khm_Khmr	43.2	**46.9**	kin_Latn	**67.6**	66.7
kir_Cyrl	**48.4**	42.3	kor_Hang	**53.6**	51.9	ksh_Latn	56.7	**60.9**	kur_Latn	62.5	**65.2**
lat_Latn	**74.2**	73.5	lav_Latn	73.2	**75.2**	lij_Latn	41.4	**47.1**	lim_Latn	66.7	**67.8**
lin_Latn	49.5	**49.8**	lit_Latn	**75.3**	75.0	lmo_Latn	**76.3**	72.5	ltz_Latn	68.5	**68.9**
lzh_Hani	**14.0**	7.3	mal_Mlym	**65.1**	63.2	mar_Deva	**65.2**	61.7	mhr_Cyrl	59.8	**61.6**
min_Latn	**44.2**	43.4	mkd_Cyrl	76.3	**76.9**	mlg_Latn	**59.4**	57.8	mlt_Latn	64.6	**74.0**
mon_Cyrl	**67.5**	66.1	mri_Latn	**50.4**	46.3	msa_Latn	68.8	**69.0**	mwl_Latn	48.5	**51.5**
mya_Mymr	**57.9**	54.5	mzn_Arab	46.2	**46.9**	nan_Latn	86.5	**86.7**	nap_Latn	62.5	**62.6**
nds_Latn	**80.9**	75.8	nep_Deva	56.5	**61.0**	nld_Latn	81.4	**81.5**	nno_Latn	**76.9**	76.4
nor_Latn	75.9	**77.9**	oci_Latn	68.2	**72.6**	ori_Orya	**28.6**	**28.6**	oss_Cyrl	**58.8**	50.6
pan_Guru	45.3	**46.5**	pms_Latn	75.0	**80.9**	pnb_Arab	**68.1**	67.8	pol_Latn	**77.9**	77.8
por_Latn	76.8	**79.8**	pus_Arab	**44.2**	40.0	que_Latn	62.4	**66.4**	roh_Latn	**61.7**	56.9
ron_Latn	78.7	**78.9**	rus_Cyrl	**70.3**	69.5	sah_Cyrl	**71.8**	71.4	san_Deva	34.6	**36.6**
scn_Latn	65.2	**69.1**	sco_Latn	82.0	**91.5**	sgs_Latn	61.8	**67.2**	sin_Sinh	**58.3**	54.5
slk_Latn	77.0	**77.7**	slv_Latn	79.2	**80.3**	snd_Arab	**43.6**	41.0	som_Latn	52.8	**58.9**
spa_Latn	73.0	**78.6**	sqi_Latn	75.6	**77.1**	srp_Cyrl	**64.8**	63.6	sun_Latn	**56.3**	55.6
swa_Latn	68.3	**68.9**	swe_Latn	**70.2**	68.7	szl_Latn	67.0	**70.9**	tam_Taml	55.4	**59.3**
tat_Cyrl	**68.2**	60.5	tel_Telu	**52.3**	50.5	tgk_Cyrl	60.8	**61.4**	tgl_Latn	75.8	**76.4**
tha_Thai	**5.0**	0.9	tuk_Latn	55.5	**57.1**	tur_Latn	76.1	**77.2**	uig_Arab	**50.2**	47.6
ukr_Cyrl	**77.2**	76.4	urd_Arab	**69.8**	63.5	uzb_Latn	**74.0**	72.9	vec_Latn	**69.6**	65.9
vep_Latn	**70.2**	68.0	vie_Latn	72.3	**73.2**	vls_Latn	73.7	**77.6**	vol_Latn	56.7	**61.0**
war_Latn	**62.8**	**62.8**	wuu_Hani	**40.8**	19.4	xmf_Geor	**65.3**	60.8	yid_Hebr	47.5	**58.2**
yor_Latn	65.5	**65.8**	yue_Hani	**23.5**	18.4	zea_Latn	63.0	**65.8**	zho_Hani	**24.7**	18.1

Table 18: F_1 scores of models on **NER**.

Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP	Language	Baseline	LANGSAMP
afr_Latn	88.0	**88.2**	ajp_Arab	**71.1**	68.8	aln_Latn	**51.9**	50.6	amh_Ethi	**67.8**	65.4
ara_Arab	66.9	**67.4**	bam_Latn	41.4	**43.5**	bel_Cyrl	**85.9**	85.1	ben_Beng	**83.7**	82.1
bre_Latn	**60.7**	59.4	bul_Cyrl	**88.6**	87.9	cat_Latn	**86.5**	86.1	ceb_Latn	**65.9**	63.1
ces_Latn	**85.1**	84.6	cym_Latn	**66.4**	64.7	dan_Latn	90.3	**90.7**	deu_Latn	**87.9**	87.4
ell_Grek	**86.6**	84.6	eng_Latn	**96.0**	95.9	est_Latn	83.7	**83.9**	eus_Latn	**65.3**	62.1
fao_Latn	**89.3**	88.1	fas_Arab	71.5	**72.6**	fin_Latn	**82.2**	81.7	fra_Latn	**86.7**	**86.7**
gla_Latn	57.0	**57.3**	gle_Latn	64.1	**64.9**	glg_Latn	**83.0**	82.1	glv_Latn	**50.7**	50.2
grc_Grek	**72.6**	71.9	grn_Latn	**20.9**	20.0	gsw_Latn	79.2	**80.3**	hbo_Hebr	37.1	**38.4**
heb_Hebr	**69.8**	68.7	hin_Deva	69.6	**72.7**	hrv_Latn	**85.8**	85.3	hsb_Latn	**82.7**	82.4
hun_Latn	81.3	**83.1**	hye_Armn	84.2	**84.9**	hyw_Armn	**81.6**	81.5	ind_Latn	**84.0**	83.1
isl_Latn	**82.8**	**82.8**	ita_Latn	88.3	**88.8**	jav_Latn	**73.6**	72.7	jpn_Jpan	25.0	**35.3**
kaz_Cyrl	**76.9**	75.2	kmr_Latn	**74.0**	73.8	kor_Hang	**52.7**	51.8	lat_Latn	**72.6**	72.2
lav_Latn	**84.0**	83.6	lij_Latn	**77.4**	76.3	lit_Latn	**81.5**	80.9	lzh_Hani	22.7	**24.3**
mal_Mlym	**86.3**	84.2	mar_Deva	**81.7**	77.9	mlt_Latn	79.4	**79.8**	myv_Cyrl	**64.2**	63.5
nap_Latn	**82.4**	**82.4**	nds_Latn	77.0	**77.9**	nld_Latn	88.3	**88.4**	nor_Latn	**88.1**	87.8
pcm_Latn	56.9	**57.3**	pol_Latn	**84.2**	82.7	por_Latn	**88.2**	87.8	quc_Latn	**63.8**	59.7
ron_Latn	81.4	**82.0**	rus_Cyrl	**89.0**	88.4	sah_Cyrl	**75.7**	71.5	san_Deva	**25.6**	24.8
sin_Sinh	**56.0**	55.7	slk_Latn	**84.8**	**84.8**	slv_Latn	**77.2**	76.7	sme_Latn	**73.2**	72.3
spa_Latn	**87.5**	87.1	sqi_Latn	76.0	**77.4**	srp_Latn	**85.4**	85.0	swe_Latn	**92.6**	92.4
tam_Taml	73.8	**73.9**	tat_Cyrl	70.4	**70.8**	tel_Telu	**81.7**	80.9	tgl_Latn	**75.2**	74.1
tha_Thai	58.3	**58.9**	tur_Latn	**71.3**	70.7	uig_Arab	**68.4**	67.3	ukr_Cyrl	**85.1**	85.0
urd_Arab	59.0	**67.0**	vie_Latn	**68.2**	67.5	wol_Latn	**60.9**	59.9	xav_Latn	**11.1**	9.2

Table 19: F_1 scores of models on **POS**.

Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor
ace_Latn	**63.3**	60.1	ach_Latn	35.6	**48.1**	acr_Latn	**48.8**	46.7	afr_Latn	**58.6**	58.5
ahk_Latn	5.4	**8.3**	aka_Latn	**44.9**	41.2	aln_Latn	**56.2**	54.7	als_Latn	**57.1**	57.1
alz_Latn	34.1	**43.0**	aoj_Latn	40.9	**46.2**	arb_Arab	**55.4**	55.4	arn_Latn	43.1	**44.4**
arz_Arab	33.7	**40.3**	asm_Beng	53.4	**61.5**	ayr_Latn	52.7	**62.2**	azb_Arab	**61.0**	61.0
bak_Cyrl	54.7	**59.7**	bam_Latn	48.9	**55.6**	ban_Latn	**43.0**	42.5	bar_Latn	**47.8**	43.3
bci_Latn	34.6	**37.1**	bcl_Latn	54.2	**60.5**	bel_Cyrl	59.1	**61.5**	bem_Latn	44.2	**49.4**
bhw_Latn	**50.2**	46.9	bim_Latn	47.3	**55.1**	bis_Latn	**68.4**	68.1	bqc_Latn	33.2	**41.6**
btx_Latn	**56.7**	53.8	bul_Cyrl	62.5	**62.6**	bum_Latn	39.6	**42.2**	bzj_Latn	**65.7**	60.3
cac_Latn	43.8	**46.0**	cak_Latn	51.0	**57.9**	caq_Latn	42.7	**51.0**	cat_Latn	61.2	**62.3**
cce_Latn	**43.8**	38.0	ceb_Latn	**49.8**	49.1	ces_Latn	63.3	**63.7**	cfm_Latn	**58.3**	57.1
chk_Latn	**42.8**	38.9	chv_Cyrl	60.3	**64.3**	ckb_Arab	58.3	**67.0**	cmn_Hani	60.8	**73.0**
crh_Cyrl	61.4	**67.7**	crs_Latn	62.3	**63.5**	csy_Latn	**58.3**	56.7	ctd_Latn	**56.6**	55.8
cuk_Latn	39.1	**40.8**	cym_Latn	**51.9**	46.0	dan_Latn	**58.1**	54.0	deu_Latn	**51.5**	51.5
dln_Latn	**54.4**	54.4	dtp_Latn	51.5	**51.6**	dyu_Latn	**55.6**	48.2	dzo_Tibt	50.6	**58.1**
ell_Grek	**56.9**	53.9	eng_Latn	**78.0**	78.0	enm_Latn	**70.8**	67.0	epo_Latn	**58.3**	58.3
eus_Latn	**25.2**	21.4	ewe_Latn	46.4	**52.1**	fao_Latn	56.5	**64.8**	fas_Arab	69.6	**70.2**
fil_Latn	56.7	**58.7**	fin_Latn	**56.4**	55.7	fon_Latn	**36.8**	35.4	fra_Latn	**66.8**	66.8
gaa_Latn	36.9	**47.7**	gil_Latn	40.4	**47.2**	giz_Latn	48.4	**48.5**	gkn_Latn	**40.0**	34.1
gla_Latn	**45.6**	45.6	gle_Latn	41.8	**45.1**	glv_Latn	37.3	**48.7**	gom_Latn	34.8	**41.6**
guc_Latn	**39.6**	37.6	gug_Latn	39.0	**46.0**	guj_Gujr	67.1	**70.4**	gur_Latn	37.0	**44.2**
gya_Latn	39.6	**41.8**	gym_Latn	45.4	**52.9**	hat_Latn	**63.0**	60.0	hau_Latn	54.0	**59.6**
heb_Hebr	**16.7**	15.2	hif_Latn	42.4	**53.6**	hil_Latn	**63.7**	61.6	hin_Deva	**64.8**	64.8
hne_Deva	64.1	**67.5**	hnj_Latn	61.5	**63.2**	hra_Latn	48.2	**53.1**	hrv_Latn	**62.7**	60.7
hun_Latn	65.2	**65.9**	hus_Latn	37.6	**40.7**	hye_Armn	67.2	**69.3**	iba_Latn	57.9	**59.2**
ifa_Latn	49.7	**51.5**	ifb_Latn	**48.3**	48.1	ikk_Latn	46.6	**52.5**	ilo_Latn	**58.8**	55.7
isl_Latn	53.5	**61.2**	ita_Latn	62.8	**67.1**	ium_Latn	51.4	**58.0**	ixl_Latn	36.6	**38.2**
jam_Latn	**66.1**	61.0	jav_Latn	43.9	**47.6**	jpn_Jpan	**58.6**	58.6	kaa_Latn	57.7	**62.6**
kac_Latn	44.5	**47.3**	kal_Latn	31.5	**34.5**	kan_Knda	60.6	**67.5**	kat_Geor	55.2	**62.2**
kbp_Latn	34.9	**39.5**	kek_Latn	**41.5**	40.3	khm_Khmr	**64.7**	64.7	kia_Latn	48.0	**51.7**
kin_Latn	47.2	**52.5**	kir_Cyrl	61.1	**64.7**	kjb_Latn	44.7	**48.1**	kjh_Cyrl	**52.3**	51.1
kmr_Cyrl	45.5	**53.1**	knv_Latn	**42.6**	40.5	kor_Hang	69.8	**71.3**	kpg_Latn	**64.1**	57.4
kri_Latn	**63.2**	56.0	ksd_Latn	54.2	**54.4**	kss_Latn	16.2	**21.6**	ksw_Mymr	**50.4**	50.3
lam_Latn	34.7	**35.6**	lao_Laoo	69.1	**72.7**	lat_Latn	57.2	**62.9**	lav_Latn	**60.4**	57.7
leh_Latn	**43.5**	37.2	lhu_Latn	22.3	**29.0**	lin_Latn	47.1	**54.7**	lit_Latn	58.3	**59.7**
ltz_Latn	**48.2**	48.2	lug_Latn	**46.1**	39.0	luo_Latn	40.6	**41.2**	lus_Latn	**51.6**	51.6
mad_Latn	55.3	**63.0**	mah_Latn	**41.6**	38.3	mai_Deva	**62.7**	60.5	mam_Latn	**33.9**	33.2
mau_Latn	5.5	**8.4**	mbb_Latn	52.6	**53.1**	mck_Latn	**41.9**	41.2	mcn_Latn	37.7	**39.3**
mdy_Ethi	51.6	**57.6**	meu_Latn	54.9	**55.8**	mfe_Latn	66.0	**66.2**	mgh_Latn	30.3	**33.1**
mhr_Cyrl	36.0	**38.5**	min_Latn	**49.9**	40.7	miq_Latn	**52.2**	52.2	mkd_Cyrl	**71.2**	70.3
mlt_Latn	**50.7**	50.7	mos_Latn	40.3	**41.2**	mps_Latn	**57.1**	53.1	mri_Latn	50.9	**52.6**

Table 20: F_1 scores of LANGSAMP on **Taxi1500** using English and the closest donor language as source (Part I).

Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor
msa_Latn	41.7	**42.0**	mwm_Latn	**55.1**	55.0	mxv_Latn	**29.6**	27.4	mya_Mymr	**54.4**	53.4
mzh_Latn	39.7	**45.1**	nan_Latn	31.5	**31.8**	naq_Latn	41.7	**43.7**	nav_Latn	21.1	**29.5**
nch_Latn	**44.0**	36.6	ncj_Latn	38.6	**39.1**	ndc_Latn	34.7	**36.6**	nde_Latn	45.7	**49.8**
nds_Latn	**49.6**	44.0	nep_Deva	68.0	**72.1**	ngu_Latn	43.4	**48.2**	nld_Latn	**61.1**	53.7
nnb_Latn	40.7	**46.1**	nno_Latn	**63.1**	63.1	nob_Latn	57.2	**58.2**	nor_Latn	56.4	**57.8**
nse_Latn	45.9	**48.5**	nso_Latn	**48.6**	48.6	nya_Latn	**56.0**	47.4	nyn_Latn	43.0	**44.1**
nzi_Latn	33.0	**33.8**	ori_Orya	**67.3**	67.3	ory_Orya	66.9	**70.7**	oss_Cyrl	55.5	**57.5**
pag_Latn	**55.5**	52.5	pam_Latn	**42.0**	37.8	pan_Guru	**64.1**	64.1	pap_Latn	**65.6**	59.8
pcm_Latn	**66.1**	65.9	pdt_Latn	**60.0**	56.5	pes_Arab	**69.0**	69.0	pis_Latn	64.3	**65.0**
plt_Latn	46.8	**52.9**	poh_Latn	44.3	**45.5**	pol_Latn	64.8	**65.1**	pon_Latn	50.5	**52.2**
prk_Latn	52.9	**53.0**	prs_Arab	69.2	**70.0**	pxm_Latn	34.5	**41.5**	qub_Latn	51.5	**56.3**
qug_Latn	**65.0**	61.3	quh_Latn	**66.7**	58.8	quw_Latn	55.9	**56.0**	quy_Latn	65.5	**67.7**
qvi_Latn	**62.0**	58.5	rap_Latn	48.9	**49.3**	rar_Latn	48.9	**51.9**	rmy_Latn	45.4	**49.1**
rop_Latn	**56.6**	54.7	rug_Latn	53.8	**55.1**	run_Latn	48.0	**55.2**	rus_Cyrl	**68.1**	68.1
sah_Cyrl	55.1	**57.6**	sba_Latn	39.1	**41.4**	seh_Latn	45.0	**46.7**	sin_Sinh	64.1	**66.9**
slv_Latn	**63.8**	60.7	sme_Latn	**42.8**	37.6	smo_Latn	**60.8**	54.2	sna_Latn	42.6	**44.9**
som_Latn	33.9	**35.5**	sop_Latn	**36.4**	36.0	sot_Latn	43.5	**45.5**	spa_Latn	**64.2**	64.2
srm_Latn	48.1	**48.4**	srn_Latn	**63.7**	62.8	srp_Latn	64.9	**65.2**	ssw_Latn	**43.7**	37.7
suz_Deva	**58.0**	57.8	swe_Latn	**66.8**	65.3	swh_Latn	**59.8**	59.8	sxn_Latn	**46.6**	40.2
tat_Cyrl	62.2	**68.2**	tbz_Latn	36.4	**39.5**	tca_Latn	43.3	**50.3**	tdt_Latn	**60.3**	55.1
teo_Latn	**23.7**	23.1	tgk_Cyrl	**60.9**	60.9	tgl_Latn	56.7	**58.7**	tha_Thai	**63.8**	63.8
tir_Ethi	**50.1**	50.1	tlh_Latn	**65.0**	65.0	tob_Latn	43.3	**50.4**	toh_Latn	37.1	**39.0**
toj_Latn	**36.6**	34.1	ton_Latn	47.3	**51.5**	top_Latn	**21.9**	21.3	tpi_Latn	63.8	**67.6**
tsn_Latn	39.8	**44.1**	tsz_Latn	40.4	**41.0**	tuc_Latn	**57.4**	56.9	tui_Latn	**43.7**	43.7
tum_Latn	**47.6**	43.2	tur_Latn	**62.1**	62.1	twi_Latn	**41.4**	38.9	tyv_Cyrl	59.8	**60.3**
tzo_Latn	**39.5**	39.5	udm_Cyrl	49.6	**49.9**	ukr_Cyrl	**62.4**	62.2	uzb_Latn	53.5	**57.7**
ven_Latn	41.9	**48.6**	vie_Latn	62.4	**65.4**	wal_Latn	**48.9**	42.7	war_Latn	47.7	**54.5**
wol_Latn	**37.2**	33.9	xav_Latn	**25.5**	23.7	xho_Latn	**44.9**	44.4	yan_Latn	50.3	**53.5**
yap_Latn	42.8	**42.9**	yom_Latn	**37.6**	34.1	yor_Latn	**41.8**	35.4	yua_Latn	40.1	**43.2**
zai_Latn	**42.6**	41.4	zho_Hani	**60.7**	60.7	zlm_Latn	**68.4**	65.5	zom_Latn	**44.6**	44.4
zul_Latn	51.9	**52.2**									

Table 21: F_1 scores of LANGSAMP on **Taxi1500** using English and the closest donor language as source (Part II).

Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor
ace_Latn	69.9	**72.4**	acm_Arab	80.6	**81.4**	afr_Latn	81.4	**81.8**	ajp_Arab	81.4	**83.0**
als_Latn	**82.3**	82.3	amh_Ethi	**72.6**	72.6	apc_Arab	81.7	**83.2**	arb_Arab	**81.5**	81.5
arz_Arab	82.1	**84.4**	asm_Beng	**83.0**	83.0	ast_Latn	87.1	**87.6**	ayr_Latn	48.6	**51.1**
azj_Latn	**86.5**	84.0	bak_Cyrl	84.3	**86.5**	bam_Latn	**46.5**	42.2	ban_Latn	79.5	**81.3**
bem_Latn	**61.1**	51.4	ben_Beng	83.7	**84.0**	bjn_Latn	75.9	**77.9**	bod_Tibt	65.7	**71.0**
bul_Cyrl	86.3	**86.6**	cat_Latn	**85.7**	85.2	ceb_Latn	81.2	**83.2**	ces_Latn	**86.3**	85.6
ckb_Arab	**80.0**	76.8	crh_Latn	**76.8**	75.7	cym_Latn	73.6	**76.6**	dan_Latn	85.0	**86.0**
dyu_Latn	**43.6**	42.4	dzo_Tibt	**68.2**	59.8	ell_Grek	**79.5**	78.8	eng_Latn	**88.9**	88.9
est_Latn	**78.9**	78.1	eus_Latn	78.8	**80.7**	ewe_Latn	**49.9**	46.7	fao_Latn	**84.4**	83.6
fin_Latn	80.9	**81.5**	fon_Latn	**40.8**	38.1	fra_Latn	**87.8**	87.8	fur_Latn	77.4	**77.9**
gle_Latn	61.5	**64.4**	glg_Latn	**87.6**	87.6	grn_Latn	71.6	**73.2**	guj_Gujr	82.1	**83.4**
hau_Latn	59.3	**64.2**	heb_Hebr	76.8	**80.2**	hin_Deva	**82.8**	82.8	hne_Deva	77.9	**79.5**
hun_Latn	86.6	**87.5**	hye_Armn	**81.3**	80.3	ibo_Latn	**71.4**	71.3	ilo_Latn	76.1	**76.7**
isl_Latn	78.0	**78.3**	ita_Latn	86.4	**87.5**	jav_Latn	**79.9**	79.7	jpn_Jpan	**86.8**	86.8
kac_Latn	**48.9**	46.6	kam_Latn	45.8	**48.3**	kan_Knda	82.9	**83.0**	kat_Geor	**83.7**	81.0
kbp_Latn	**42.8**	42.2	kea_Latn	**73.1**	73.1	khm_Khmr	**82.7**	82.7	kik_Latn	55.1	**56.7**
kir_Cyrl	79.3	**80.1**	kmb_Latn	**46.2**	42.6	kmr_Latn	**69.8**	68.9	kon_Latn	**65.2**	63.4
lao_Laoo	**83.4**	82.9	lij_Latn	**76.4**	74.9	lim_Latn	**74.1**	73.0	lin_Latn	68.2	**73.3**
lmo_Latn	77.0	**78.3**	ltz_Latn	**76.4**	76.4	lua_Latn	**54.4**	54.3	lug_Latn	**58.2**	55.8
lus_Latn	**64.8**	64.8	lvs_Latn	**83.2**	83.0	mai_Deva	**82.9**	82.1	mal_Mlym	**79.8**	79.3
min_Latn	76.7	**79.8**	mkd_Cyrl	**83.6**	82.8	mlt_Latn	**81.3**	81.3	mos_Latn	**44.7**	40.9
mya_Mymr	**80.5**	78.8	nld_Latn	85.1	**86.4**	nno_Latn	**86.0**	86.0	nob_Latn	**84.8**	84.4
nso_Latn	**57.6**	57.6	nya_Latn	69.2	**70.9**	oci_Latn	**85.0**	84.1	ory_Orya	78.6	**79.0**
pan_Guru	**76.4**	76.4	pap_Latn	76.9	**78.1**	pes_Arab	**87.5**	87.3	plt_Latn	67.5	**69.3**
por_Latn	85.3	**86.8**	prs_Arab	85.0	**85.5**	quy_Latn	**62.6**	59.7	ron_Latn	84.0	**84.4**
rus_Cyrl	**86.8**	86.8	sag_Latn	**51.3**	50.2	san_Deva	72.9	**76.6**	sat_Olck	**56.4**	53.5
sin_Sinh	**82.7**	82.7	slk_Latn	**85.4**	85.1	slv_Latn	84.2	**87.4**	smo_Latn	74.2	**75.3**
snd_Arab	**70.4**	70.4	som_Latn	58.9	**61.1**	sot_Latn	**64.1**	63.2	spa_Latn	**84.4**	84.4
srp_Cyrl	84.8	**85.0**	ssw_Latn	64.1	**65.2**	sun_Latn	82.6	**85.2**	swe_Latn	84.2	**86.2**
szl_Latn	**72.4**	72.4	tam_Taml	**81.2**	81.2	tat_Cyrl	**83.6**	83.6	tel_Telu	84.0	**85.4**
tgl_Latn	**82.1**	81.7	tha_Thai	85.4	**85.7**	tir_Ethi	**60.3**	60.3	tpi_Latn	**80.3**	75.7
tso_Latn	57.3	**60.3**	tuk_Latn	78.1	**78.5**	tum_Latn	65.4	**68.5**	tur_Latn	**80.4**	80.4
uig_Arab	**75.5**	75.5	ukr_Cyrl	**84.3**	83.8	umb_Latn	41.0	**46.5**	urd_Arab	79.1	**80.6**
vie_Latn	**86.2**	83.9	war_Latn	80.7	**81.3**	wol_Latn	**50.5**	46.4	xho_Latn	**60.1**	59.8
zho_Hans	**89.6**	89.2	zho_Hant	**88.8**	88.8	zsm_Latn	**86.4**	86.0	zul_Latn	68.1	**69.8**

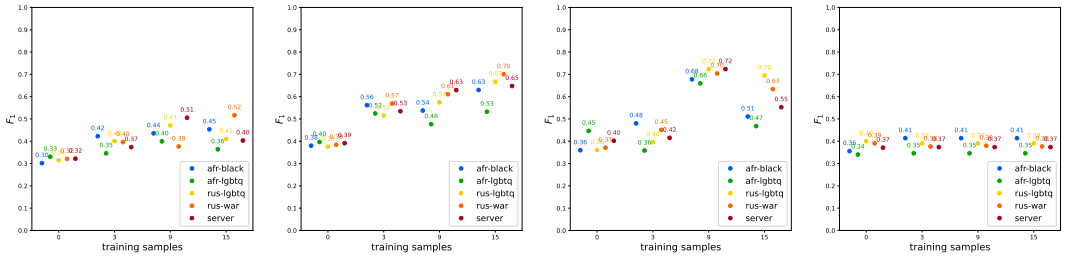
Table 22: F_1 scores of LANGSAMP on **SIB200**. using English and the closest donor language as source.

Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor
ace_Latn	41.5	**56.9**	afr_Latn	75.8	**80.3**	als_Latn	**80.9**	80.9	amh_Ethi	**39.7**	39.7
arg_Latn	82.2	**88.8**	arz_Arab	55.1	**82.6**	asm_Beng	**69.0**	45.9	ast_Latn	84.6	**85.8**
aze_Latn	65.0	**74.0**	bak_Cyrl	62.5	**72.2**	bar_Latn	**68.2**	62.8	bel_Cyrl	74.9	**79.7**
bih_Deva	56.2	**67.6**	bod_Tibt	35.2	**35.7**	bos_Latn	70.1	**75.2**	bre_Latn	63.3	**66.0**
cat_Latn	83.8	**85.1**	cbk_Latn	**53.7**	48.9	ceb_Latn	**56.0**	26.8	ces_Latn	**77.9**	69.6
chv_Cyrl	73.6	**84.3**	ckb_Arab	**76.0**	60.6	cos_Latn	**63.0**	61.9	crh_Latn	52.7	**59.4**
cym_Latn	61.7	**62.1**	dan_Latn	**81.4**	81.3	deu_Latn	**74.6**	74.6	diq_Latn	54.0	**72.2**
ell_Grek	71.9	**72.0**	eml_Latn	**41.3**	41.3	eng_Latn	**83.5**	83.5	epo_Latn	**68.3**	68.3
eus_Latn	60.9	**65.1**	ext_Latn	44.2	**48.6**	fao_Latn	68.7	**79.2**	fas_Arab	**55.0**	53.6
fra_Latn	**76.5**	76.5	frr_Latn	**52.0**	52.0	fry_Latn	**74.6**	73.9	fur_Latn	**58.2**	54.0
gle_Latn	**72.6**	69.6	glg_Latn	80.7	**86.1**	grn_Latn	55.1	**59.8**	guj_Gujr	**61.2**	61.0
heb_Hebr	52.0	**52.9**	hin_Deva	**69.4**	69.4	hrv_Latn	77.2	**79.8**	hsb_Latn	**74.3**	69.7
hye_Armn	53.0	**62.2**	ibo_Latn	58.1	**58.4**	ido_Latn	**82.6**	81.5	ilo_Latn	**80.0**	74.9
ind_Latn	**67.6**	67.6	isl_Latn	70.1	**75.4**	ita_Latn	78.2	**79.5**	jav_Latn	56.0	**86.4**
jpn_Jpan	**22.0**	22.0	kan_Knda	57.5	**61.8**	kat_Geor	**68.7**	60.1	kaz_Cyrl	50.5	**57.1**
kin_Latn	**69.6**	67.3	kir_Cyrl	44.3	**60.9**	kor_Hang	50.4	**51.2**	ksh_Latn	**59.7**	51.4
lat_Latn	71.9	**81.4**	lav_Latn	**74.4**	69.0	lij_Latn	45.2	**54.2**	lim_Latn	**69.3**	61.2
lit_Latn	74.2	**76.1**	lmo_Latn	**73.6**	65.5	ltz_Latn	**67.9**	67.9	lzh_Hani	**14.8**	14.8
mar_Deva	62.5	**76.6**	mhr_Cyrl	60.6	**72.3**	min_Latn	42.6	**57.5**	mkd_Cyrl	72.2	**73.1**
mlt_Latn	**75.9**	75.9	mon_Cyrl	**68.7**	60.9	mri_Latn	**50.0**	47.0	msa_Latn	67.6	**73.0**
mya_Mymr	55.3	**56.3**	mzn_Arab	43.3	**47.2**	nan_Latn	**88.1**	36.6	nap_Latn	**63.0**	55.3
nep_Deva	56.9	**60.4**	nld_Latn	**80.8**	80.0	nno_Latn	**77.6**	77.6	nor_Latn	77.9	**80.4**
ori_Orya	**34.2**	34.2	oss_Cyrl	50.6	**59.1**	pan_Guru	**51.5**	51.5	pms_Latn	**80.9**	78.4
pol_Latn	**77.7**	71.1	por_Latn	78.9	**84.9**	pus_Arab	42.6	**45.3**	que_Latn	**70.4**	55.5
ron_Latn	**77.8**	75.5	rus_Cyrl	**67.5**	67.5	sah_Cyrl	71.9	**77.9**	san_Deva	38.4	**53.4**
sco_Latn	**86.4**	84.5	sgs_Latn	66.4	**69.8**	sin_Sinh	**53.0**	51.2	slk_Latn	**76.4**	55.9
snd_Arab	**41.8**	41.8	som_Latn	**57.5**	56.2	spa_Latn	**77.6**	77.6	sqi_Latn	76.8	**78.7**
sun_Latn	50.8	**75.1**	swa_Latn	**71.8**	71.8	swe_Latn	**70.9**	65.8	szl_Latn	**70.9**	70.9
tat_Cyrl	63.8	**76.5**	tel_Telu	48.1	**49.0**	tgk_Cyrl	**68.4**	68.4	tgl_Latn	71.9	**73.7**
tuk_Latn	54.4	**57.3**	tur_Latn	**77.1**	77.1	uig_Arab	47.7	**62.3**	ukr_Cyrl	76.6	**85.3**
uzb_Latn	73.2	**76.0**	vec_Latn	68.0	**75.1**	vep_Latn	**72.0**	63.0	vie_Latn	**72.3**	49.7
vol_Latn	**61.0**	36.5	war_Latn	**64.9**	56.1	wuu_Hani	35.7	**66.7**	xmf_Geor	**69.3**	55.7
yor_Latn	**69.3**	41.7	yue_Hani	25.7	**73.5**	zea_Latn	62.9	**75.4**	zho_Hani	**25.2**	25.2

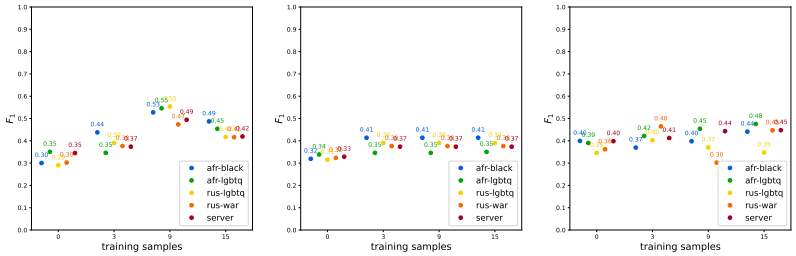
Table 23: F_1 scores of LANGSAMP on **NER** using English and the closest donor language as source.

Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor	Language	English	Closest donor
afr_Latn	**88.5**	79.5	ajp_Arab	**71.1**	41.9	aln_Latn	**53.4**	45.1	amh_Ethi	**66.8**	66.8
bam_Latn	**43.0**	31.2	bel_Cyrl	86.4	**93.8**	ben_Beng	**87.5**	80.2	bre_Latn	61.1	**62.3**
cat_Latn	86.8	**95.8**	ceb_Latn	**66.7**	32.5	ces_Latn	**85.4**	73.3	cym_Latn	**65.5**	60.4
deu_Latn	**88.2**	88.2	ell_Grek	**84.9**	75.5	eng_Latn	**96.0**	96.0	est_Latn	**84.7**	77.4
fao_Latn	**88.7**	67.5	fas_Arab	**72.2**	69.1	fin_Latn	**82.2**	75.8	fra_Latn	**85.8**	85.8
gle_Latn	64.6	**65.5**	glg_Latn	83.6	**87.8**	glv_Latn	51.9	**57.8**	grc_Grek	**71.6**	71.6
gsw_Latn	**82.7**	82.7	hbo_Hebr	**38.9**	37.4	heb_Hebr	67.9	**69.3**	hin_Deva	**77.2**	77.2
hsb_Latn	**83.7**	73.4	hun_Latn	**82.2**	42.0	hye_Armn	**85.1**	84.9	hyw_Armn	**83.0**	56.8
isl_Latn	**82.7**	81.2	ita_Latn	88.9	**92.4**	jav_Latn	75.4	**78.8**	jpn_Jpan	**33.1**	33.1
kmr_Latn	**76.6**	61.6	kor_Hang	**52.7**	45.3	lat_Latn	72.8	**74.2**	lav_Latn	**83.7**	78.4
lit_Latn	**82.1**	80.7	lzh_Hani	**24.5**	24.5	mal_Mlym	**86.0**	52.1	mar_Deva	**84.1**	81.7
myv_Cyrl	**65.9**	58.4	nap_Latn	**82.4**	70.6	nds_Latn	**79.1**	34.0	nld_Latn	**88.2**	82.2
pcm_Latn	**58.2**	48.1	pol_Latn	84.2	**89.1**	por_Latn	87.9	**92.0**	quc_Latn	**63.3**	52.6
rus_Cyrl	**88.7**	88.7	sah_Cyrl	74.2	**74.5**	san_Deva	25.5	**32.7**	sin_Sinh	**56.2**	34.4
slv_Latn	77.6	**79.0**	sme_Latn	**74.8**	60.6	spa_Latn	**87.8**	87.8	sqi_Latn	**77.5**	72.7
swe_Latn	**92.7**	83.2	tam_Taml	**74.6**	74.6	tat_Cyrl	**72.4**	70.9	tel_Telu	**80.9**	55.9
tha_Thai	**58.3**	27.5	tur_Latn	**71.2**	71.2	uig_Arab	**68.2**	48.3	ukr_Cyrl	85.6	**91.7**
vie_Latn	**68.4**	32.4	wol_Latn	**61.6**	57.4	xav_Latn	**16.7**	11.2	yor_Latn	**62.7**	46.5
zho_Hani	**47.4**	47.4									

Table 24: F_1 scores of LANGSAMP on **POS** using English and the closest donor language as source.

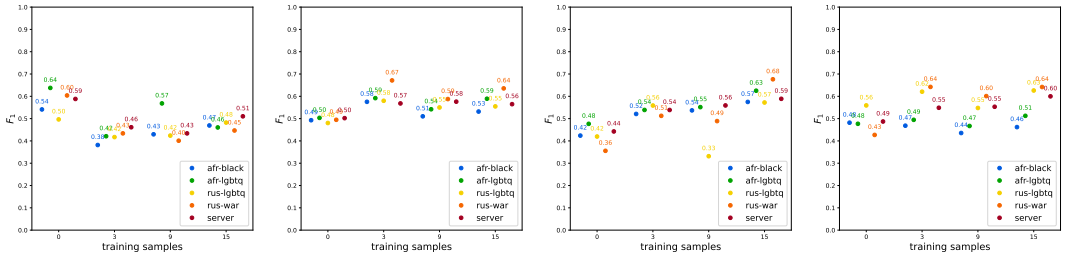


(a) XLM-R (b) mBERT (c) Distil-mBERT (d) mMiniLM

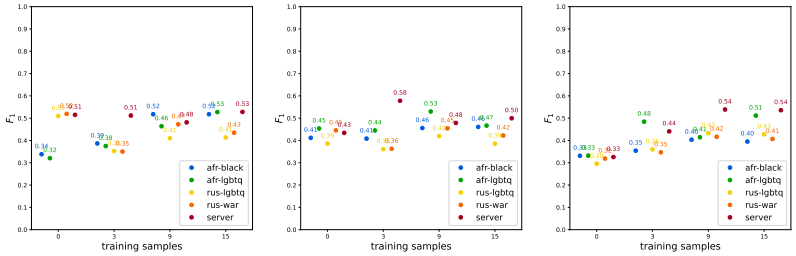


(e) DistilBERT (f) TinyBERT (g) ALBERT

Figure 1: F_1 scores of all four clients and the server, with different colors representing each client and the server. Results are shown for the seven models tested.

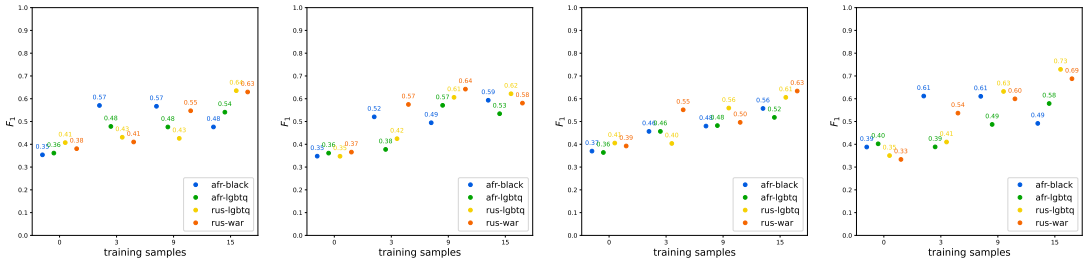


(a) XLM-R-HC (b) mBERT-HC (c) Distil-mBERT-HC (d) mMiniLM-HC



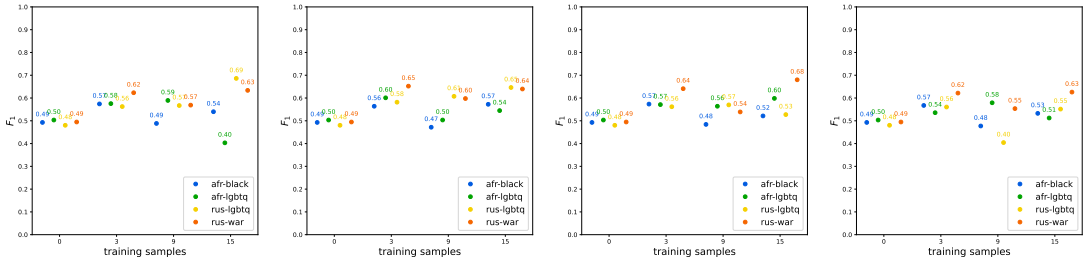
(e) DistilBERT-HC (f) TinyBERT-HC (g) ALBERT-HC

Figure 2: F_1 scores of all four clients and the server, with different colors representing each client and the server. Results are shown for the seven models fine-tuned on English HateCheck data prior to FL.



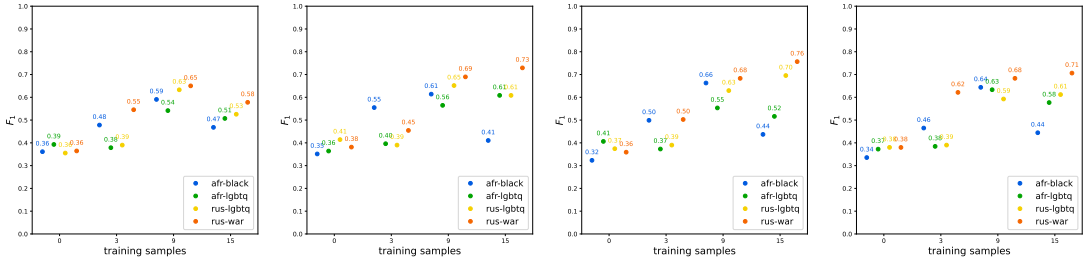
(a) $K_P = 1$ (b) $K_P = 2$ (c) $K_P = 3$ (d) $K_P = 4$

Figure 3: FedPer results for mBERT with K_P (number of personalized layers) $\in \{1, 2, 3, 4\}$.



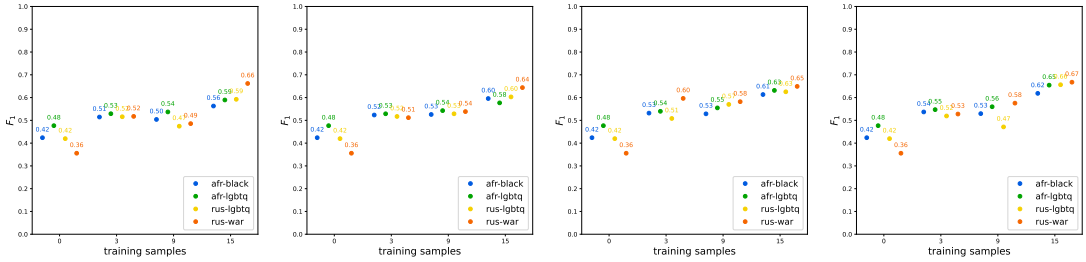
(a) $K_P = 1$ (b) $K_P = 2$ (c) $K_P = 3$ (d) $K_P = 4$

Figure 4: FedPer results for mBERT-HC with K_P (number of personalized layers) $\in \{1, 2, 3, 4\}$.



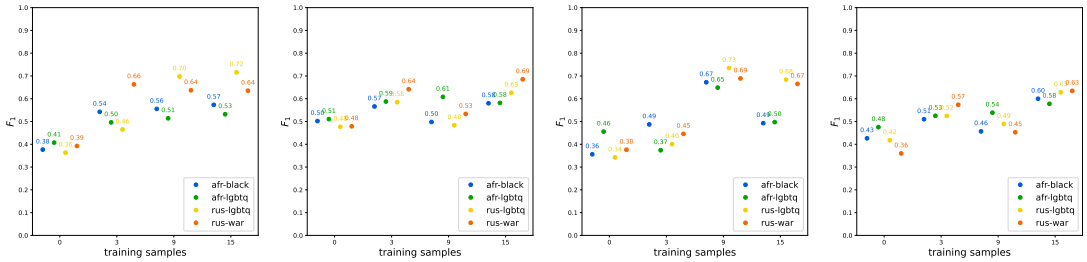
(a) $K_P = 1$ (b) $K_P = 2$ (c) $K_P = 3$ (d) $K_P = 4$

Figure 5: FedPer results for Distil-mBERT with K_P (number of personalized layers) $\in \{1, 2, 3, 4\}$.

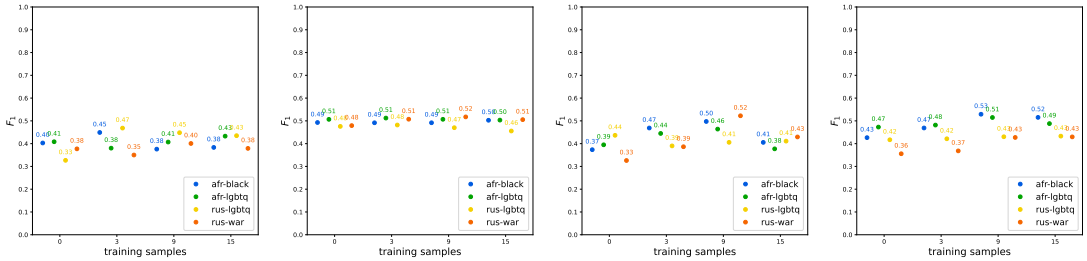


(a) $K_P = 1$ (b) $K_P = 2$ (c) $K_P = 3$ (d) $K_P = 4$

Figure 6: FedPer results for Distil-mBERT-HC with K_P (number of personalized layers) $\in \{1, 2, 3, 4\}$.



(a) mBERT (b) mBERT-HC (c) Distil-mBERT (d) Distil-mBERT-HC



(e) mBERT (f) mBERT-HC (g) Distil-mBERT (h) Distil-mBERT-HC

Figure 7: Adapter-based personalization results for all four clients using mBERT and Distil-mBERT. The top row presents results from full-model fine-tuning, while the bottom row presents results from adapter-only fine-tuning.

Bibliography

Adebara, I., Elmadany, A., Abdul-Mageed, M., and Alcoba Inciarte, A. (2023). SERENGETI: Massively multilingual language models for Africa. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1498–1537, Toronto, Canada. Association for Computational Linguistics.

Adelani, D., Liu, H., Shen, X., Vassilyev, N., Alabi, J., Mao, Y., Gao, H., and Lee, E.-S. (2024). SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In Graham, Y. and Purver, M., editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Ahmad, W., Zhang, Z., Ma, X., Hovy, E., Chang, K.-W., and Peng, N. (2019). On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2440–2452, Minneapolis, Minnesota. Association for Computational Linguistics.

Ainslie, J., Lee-Thorp, J., de Jong, M., Zemlyanskiy, Y., Lebron, F., and Sanghai, S. (2023). GQA: Training generalized multi-query transformer models from multi-head checkpoints. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore. Association for Computational Linguistics.

Alabi, J. O., Adelani, D. I., Mosbach, M., and Klakow, D. (2022). Adapting pre-trained language models to African languages via multilingual adaptive fine-tuning. In Calzolari, N., Huang, C.-R., Kim, H., Pustejovsky, J., Wanner, L., Choi, K.-S., Ryu, P.-M., Chen, H.-H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T. K., Santus, E., Bond, F., and Na, S.-H., editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4336–4349, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Alacam, Ö., Hoeken, S., and Zarrieß, S. (2024). Eyes don't lie: Subjective hate annotation and detection with gaze. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 187–205, Miami, Florida, USA. Association for Computational Linguistics.

Amari, S. (1967). A theory of adaptive pattern classifiers. *IEEE Trans. Electron. Comput.*, 16(3):299–307.

Ammar, W., Mulcaire, G., Ballesteros, M., Dyer, C., and Smith, N. A. (2016). Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Anil, R., Borgeaud, S., Wu, Y., Alayrac, J., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., Millican, K., Silver, D., Petrov, S., Johnson, M., Antonoglou, I., Schrittwieser, J., Glaese, A., Chen, J., Pitler, E., Lillicrap, T. P., Lazaridou, A., Firat, O., Molloy, J., Isard, M., Barham, P. R., Hennigan, T., Lee, B., Viola, F., Reynolds, M., Xu, Y., Doherty, R., Collins, E., Meyer, C., Rutherford, E., Moreira, E., Ayoub, K., Goel, M., Tucker, G., Piqueras, E., Krikun, M., Barr, I., Savinov, N., Danihelka, I., Roelofs, B., White, A., Andreassen, A., von Glehn, T., Yagati, L., Kazemi, M., Gonzalez, L., Khalman, M., Sygnowski, J., and et al. (2023a). Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805.

Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., Chu, E., Clark, J. H., Shafey, L. E., Huang, Y., Meier-Hellstern, K., Mishra, G., Moreira, E., Omernick, M., Robinson, K., Ruder, S., Tay, Y., Xiao, K., Xu, Y., Zhang, Y., Ábrego, G. H., Ahn, J., Austin, J., Barham, P., Botha, J. A., Bradbury, J., Brahma, S., Brooks, K., Catasta, M., Cheng, Y., Cherry, C., Choquette-Choo, C. A., Chowdhery, A., Crepy, C., Dave, S., Dehghani, M., Dev, S., Devlin, J., Díaz, M., Du, N., Dyer, E., Feinberg, V., Feng, F., Fienber, V., Freitag, M., Garcia, X., Gehrmann, S., Gonzalez, L., and et al. (2023b). Palm 2 technical report. *CoRR*, abs/2305.10403.

Anthropic (2024a). The claude 3 model family: Opus, sonnet, haiku. `https://www.anthropic.com/news/claude-3-family`.

Anthropic (2024b). Claude 3.5 sonnet model card addendum. `https://www.anthropic.com/news/claude-3-5-sonnet`.

Antoun, W., Baly, F., and Hajj, H. (2020). AraBERT: Transformer-based model for Arabic language understanding. In Al-Khalifa, H., Magdy, W., Darwish, K., Elsayed, T., and Mubarak, H., editors, *Proceedings of the 4th Workshop on Open-Source*

Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection, pages 9–15, Marseille, France. European Language Resource Association.

Arango Monnar, A., Perez, J., Poblete, B., Saldaña, M., and Proust, V. (2022). Resources for multilingual hate speech detection. In Narang, K., Mostafazadeh Davani, A., Mathias, L., Vidgen, B., and Talat, Z., editors, *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*, pages 122–130, Seattle, Washington (Hybrid). Association for Computational Linguistics.

Arivazhagan, M. G., Aggarwal, V., Singh, A. K., and Choudhary, S. (2019). Federated learning with personalization layers. *CoRR*, abs/1912.00818.

Artetxe, M., Goswami, V., Bhosale, S., Fan, A., and Zettlemoyer, L. (2023). Revisiting machine translation for cross-lingual classification. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6489–6499, Singapore. Association for Computational Linguistics.

Artetxe, M., Labaka, G., and Agirre, E. (2017). Learning bilingual word embeddings with (almost) no bilingual data. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada. Association for Computational Linguistics.

Artetxe, M., Labaka, G., and Agirre, E. (2018). A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Artetxe, M., Ruder, S., and Yogatama, D. (2020a). On the cross-lingual transferability of monolingual representations. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Artetxe, M., Ruder, S., Yogatama, D., Labaka, G., and Agirre, E. (2020b). A call for more rigor in unsupervised cross-lingual learning. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7375–7388, Online. Association for Computational Linguistics.

Artetxe, M. and Schwenk, H. (2019). Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Ba, L. J., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *CoRR*, abs/1607.06450.

Baerman, M., Palancar, E. L., and Feist, T. (2019). Inflectional class complexity in the oto-manguean languages. *Amerindia*, 41:1–18.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., Chen, C., Olsson, C., Olah, C., Hernandez, D., Drain, D., Ganguli, D., Li, D., Tran-Johnson, E., Perez, E., Kerr, J., Mueller, J., Ladish, J., Landau, J., Ndousse, K., Lukosiute, K., Lovitt, L., Sellitto, M., Elhage, N., Schiefer, N., Mercado, N., DasSarma, N., Lasenby, R., Larson, R., Ringer, S., Johnston, S., Kravec, S., Showk, S. E., Fort, S., Lanham, T., Telleen-Lawton, T., Conerly, T., Henighan, T., Hume, T., Bowman, S. R., Hatfield-Dodds, Z., Mann, B., Amodei, D., Joseph, N., McCandlish, S., Brown, T., and Kaplan, J. (2022). Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

Bandarkar, L., Liang, D., Muller, B., Artetxe, M., Shukla, S. N., Husa, D., Goyal, N., Krishnan, A., Zettlemoyer, L., and Khabsa, M. (2024). The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand. Association for Computational Linguistics.

Bansal, Y., Nakkiran, P., and Barak, B. (2021). Revisiting model stitching to compare neural representations. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 225–236.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Bengio, Y., Frasconi, P., and Simard, P. Y. (1993). The problem of learning long-term dependencies in recurrent networks. In *Proceedings of International Conference on Neural Networks (ICNN'88), San Francisco, CA, USA, March 28 - April 1, 1993*, pages 1183–1188. IEEE.

Bianchi, F., Tagliabue, J., Yu, B., Bigon, L., and Greco, C. (2020). Fantastic embeddings and how to align them: Zero-shot inference in a multi-shop scenario. In *Proceedings*

of the ACM SIGIR Workshop on eCommerce (SIGIR eCom'20), Virtual Event, China. ACM. https://sigir-ecom.github.io/ecom2020/ecom20Papers/paper11.pdf.

Bird, S. (2020). Decolonising speech and language technology. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3504–3519, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Bird, S. (2024). Must NLP be extractive? In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14915–14929, Bangkok, Thailand. Association for Computational Linguistics.

Bird, S., Klein, E., and Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media.

Bjerva, J. and Augenstein, I. (2018). From phonology to syntax: Unsupervised linguistic typology at different levels with language embeddings. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 907–916, New Orleans, Louisiana. Association for Computational Linguistics.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Blust, R. (2013). The austronesian languages (revised edition). *Canberra: ANU-Asia Pacific Linguistics*.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečný, J., Mazzocchi, S., McMahan, B., Van Overveldt, T., Petrou, D., Ramage, D., and Roselander, J. (2019). Towards federated learning at scale: System design. In Talwalkar, A., Smith, V., and Zaharia, M., editors, *Proceedings of Machine Learning and Systems*, volume 1, pages 374–388.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C.,

McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Bui, D., Malik, K., Goetz, J., Liu, H., Moon, S., Kumar, A., and Shin, K. G. (2019). Federated user representation learning. *CoRR*, abs/1909.12535.

Byrd, D. and Polychroniadou, A. (2020). Differentially private secure multi-party computation for federated learning in financial applications. In Balch, T., editor, *ICAIF '20: The First ACM International Conference on AI in Finance, New York, NY, USA, October 15-16, 2020*, pages 16:1–16:9. ACM.

Campbell, E. (2016). Tone and inflection in zenzontepec chatino. *Tone and inflection*, pages 141–162.

Cañete, J., Chaperon, G., Fuentes, R., Ho, J., Kang, H., and Pérez, J. (2023). Spanish pre-trained BERT model and evaluation data. *CoRR*, abs/2308.02976.

Cao, S., Kitaev, N., and Klein, D. (2020). Multilingual alignment of contextual word representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Carlini, N., Ippolito, D., Jagielski, M., Lee, K., Tramèr, F., and Zhang, C. (2023). Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Carlini, N., Tramèr, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T. B., Song, D., Erlingsson, Ú., Oprea, A., and Raffel, C. (2021). Extracting training data from large language models. In Bailey, M. D. and Greenstadt, R., editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 2633–2650. USENIX Association.

Caselli, T., Basile, V., Mitrović, J., and Granitzer, M. (2021). HateBERT: Retraining BERT for abusive language detection in English. In Mostafazadeh Davani, A., Kiela, D., Lambert, M., Vidgen, B., Prabhakaran, V., and Waseem, Z., editors, *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., and Kurzweil, R. (2018). Universal sentence encoder for English. In Blanco, E. and Lu, W., editors, *Proceedings of the*

2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Chan, B., Schweter, S., and Möller, T. (2020). German's next language model. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6788–6796, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Chang, T., Tu, Z., and Bergen, B. (2022). The geometry of multilingual language model representations. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 119–136, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chen, P.-H. and Chen, Y.-N. (2024). Efficient unseen language adaptation for multilingual pre-trained language models. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18983–18994, Miami, Florida, USA. Association for Computational Linguistics.

Chen, X. and Cardie, C. (2018). Unsupervised multilingual word embeddings. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 261–270, Brussels, Belgium. Association for Computational Linguistics.

Chen, Y., Biswas, R., and Bjerva, J. (2023). Colex2Lang: Language embeddings from semantic typology. In Alumäe, T. and Fishel, M., editors, *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 673–684, Tórshavn, Faroe Islands. University of Tartu Library.

Cheng, J., Dong, L., and Lapata, M. (2016). Long short-term memory-networks for machine reading. In Su, J., Duh, K., and Carreras, X., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas. Association for Computational Linguistics.

Christiano, P. F., Leike, J., Brown, T. B., Martic, M., Legg, S., and Amodei, D. (2017). Deep reinforcement learning from human preferences. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4299–4307.

Chronopoulou, A., Stojanovski, D., and Fraser, A. (2020). Reusing a Pretrained Language Model on Languages with Limited Corpora for Unsupervised NMT. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2703–2711, Online. Association for Computational Linguistics.

Chronopoulou, A., Stojanovski, D., and Fraser, A. (2023). Language-family adapters for low-resource multilingual neural machine translation. In Ojha, A. K., Liu, C.-h., Vylomova, E., Pirinen, F., Abbott, J., Washington, J., Oco, N., Malykh, V., Logacheva, V., and Zhao, X., editors, *Proceedings of the Sixth Workshop on Technologies for Machine Translation of Low-Resource Languages (LoResMT 2023)*, pages 59–72, Dubrovnik, Croatia. Association for Computational Linguistics.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V. Y., Huang, Y., Dai, A. M., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. (2024). Scaling instruction-finetuned language models. *J. Mach. Learn. Res.*, 25:70:1–70:53.

Clark, J. H., Garrette, D., Turc, I., and Wieting, J. (2022). Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Cohen, S. B., Das, D., and Smith, N. A. (2011). Unsupervised structure prediction with non-parallel multilingual guidance. In Barzilay, R. and Johnson, M., editors, *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 50–61, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Collins, C. and Kayne, R. (2009). Sswl: Syntactic structures of the world's languages. `https://www.terraling.com/groups/7`. Online database on TerraLing; accessed 2025-10-21.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In Cohen, W. W., McCallum, A., and Roweis, S. T., editors, *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 160–167. ACM.

Communication, S., Barrault, L., Chung, Y., Meglioli, M. C., Dale, D., Dong, N., Duquenne, P., Elsahar, H., Gong, H., Heffernan, K., Hoffman, J., Klaiber, C., Li, P., Licht, D., Maillard, J., Rakotoarison, A., Sadagopan, K. R., Wenzek, G., Ye,

E., Akula, B., Chen, P., Hachem, N. E., Ellis, B., Gonzalez, G. M., Haaheim, J., Hansanti, P., Howes, R., Huang, B., Hwang, M., Inaguma, H., Jain, S., Kalbassi, E., Kallet, A., Kulikov, I., Lam, J., Li, D., Ma, X., Mavlyutov, R., Peloquin, B. N., Ramadan, M., Ramakrishnan, A., Sun, A. Y., Tran, K., Tran, T., Tufanov, I., Vogeti, V., Wood, C., Yang, Y., Yu, B., Andrews, P., Balioglu, C., Costa-jussà, M. R., Celebi, O., Elbayad, M., Gao, C., Guzmán, F., Kao, J., Lee, A., Mourachko, A., Pino, J., Popuri, S., Ropers, C., Saleem, S., Schwenk, H., Tomasello, P., Wang, C., Wang, J., and Wang, S. (2023). Seamlessm4t-massively multilingual & multimodal machine translation. *CoRR*, abs/2308.11596.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Conneau, A. and Lample, G. (2019). Cross-lingual language model pretraining. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7057–7067.

Conneau, A., Rinott, R., Lample, G., Williams, A., Bowman, S., Schwenk, H., and Stoyanov, V. (2018). XNLI: Evaluating cross-lingual sentence representations. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Costa-jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., Sun, A. Y., Wang, S., Wenzek, G., Youngblood, A., Akula, B., Barrault, L., Gonzalez, G. M., Hansanti, P., Hoffman, J., Jarrett, S., Sadagopan, K. R., Rowe, D., Spruit, S., Tran, C., Andrews, P., Ayan, N. F., Bhosale, S., Edunov, S., Fan, A., Gao, C., Goswami, V., Guzmán, F., Koehn, P., Mourachko, A., Ropers, C., Saleem, S., Schwenk, H., and Wang, J. (2022). No language left behind: Scaling human-centered machine translation. *CoRR*, abs/2207.04672.

Dabre, R., Chu, C., and Kunchukuttan, A. (2021). A survey of multilingual neural machine translation. *ACM Comput. Surv.*, 53(5):99:1–99:38.

Das, M., Banerjee, S., Saha, P., and Mukherjee, A. (2022). Hate speech and offensive language detection in Bengali. In He, Y., Ji, H., Li, S., Liu, Y., and Chang, C.-H.,

editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 286–296, Online only. Association for Computational Linguistics.

Datta, A., Ramabhadran, B., Emond, J., Kannan, A., and Roark, B. (2020). Language-agnostic multilingual modeling. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 8239–8243. IEEE.

Davidson, T., Bhattacharya, D., and Weber, I. (2019). Racial bias in hate speech and abusive language detection datasets. In Roberts, S. T., Tetreault, J., Prabhakaran, V., and Waseem, Z., editors, *Proceedings of the Third Workshop on Abusive Language Online*, pages 25–35, Florence, Italy. Association for Computational Linguistics.

Davies, N. F. (1976). Receptive versus productive skills in foreign language learning. *The Modern Language Journal*, 60(8):440–443.

de Marneffe, M.-C., Manning, C. D., Nivre, J., and Zeman, D. (2021). Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

de Vries, W. and Nissim, M. (2021). As good as new. how to successfully recycle English GPT-2 to make models for other languages. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 836–846, Online. Association for Computational Linguistics.

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., van Noord, G., and Nissim, M. (2019). Bertje: A dutch BERT model. *CoRR*, abs/1912.09582.

de Vries, W., van Cranenburgh, A., and Nissim, M. (2020). What's so special about BERT's layers? a closer look at the NLP pipeline in monolingual and multilingual models. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4339–4350, Online. Association for Computational Linguistics.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.*, 41(6):391–407.

Dellert, J., Daneyko, T., Münch, A., Ladygina, A., Buch, A., Clarius, N., Grigorjew, I., Balabel, M., Boga, H. I., Baysarova, Z., et al. (2020). Northeuralex: A wide-coverage lexical database of northern eurasia. *Language resources and evaluation*, 54:273–301.

Deng, Y., Kamani, M. M., and Mahdavi, M. (2020). Adaptive personalized federated learning. *CoRR*, abs/2003.13461.

Deutscher, G. (2010). *Through the language glass: Why the world looks different in other languages*. Metropolitan books.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dixon, L., Li, J., Sorensen, J., Thain, N., and Vasserman, L. (2018). Measuring and mitigating unintended bias in text classification. In Furman, J., Marchant, G. E., Price, H., and Rossi, F., editors, *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018*, pages 67–73. ACM.

Djuric, N., Zhou, J., Morris, R., Grbovic, M., Radosavljevic, V., and Bhamidipati, N. (2015). Hate speech detection with comment embeddings. In Gangemi, A., Leonardi, S., and Panconesi, A., editors, *Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume*, pages 29–30. ACM.

Dodge, J., Sap, M., Marasović, A., Agnew, W., Ilharco, G., Groeneveld, D., Mitchell, M., and Gardner, M. (2021). Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Dryer, M. S. and Haspelmath, M., editors (2013). *WALS Online (v2020.4)*. Zenodo. https://doi.org/10.5281/zenodo.13950591.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., Gregerson, A., Spataru, A., Rozière, B., Biron, B., Tang, B., Chern, B., Caucheteux, C., Nayak, C., Bi, C., Marra, C., McConnell, C., Keller, C., Touret, C., Wu, C., Wong, C., Ferrer, C. C., Nikolaidis, C., Allonsius, D., Song, D., Pintz, D., Livshits, D., Esiobu, D., Choudhary, D., Mahajan, D., Garcia-Olano, D., Perino, D., Hupkes, D., Lakomkin, E., AlBadawy, E., Lobanova, E., Dinan, E., Smith, E. M., Radenovic, F.,

Zhang, F., Synnaeve, G., Lee, G., Anderson, G. L., Nail, G., Mialon, G., Pang, G., Cucurell, G., Nguyen, H., Korevaar, H., Xu, H., Touvron, H., Zarov, I., Ibarra, I. A., Kloumann, I. M., Misra, I., Evtimov, I., Copet, J., Lee, J., Geffert, J., Vranes, J., Park, J., Mahadeokar, J., Shah, J., van der Linde, J., Billock, J., Hong, J., Lee, J., Fu, J., Chi, J., Huang, J., Liu, J., Wang, J., Yu, J., Bitton, J., Spisak, J., Park, J., Rocca, J., Johnstun, J., Saxe, J., Jia, J., Alwala, K. V., Upasani, K., Plawiak, K., Li, K., Heafield, K., Stone, K., and et al. (2024). The llama 3 herd of models. *CoRR*, abs/2407.21783.

Dufter, P., Zhao, M., Schmitt, M., Fraser, A., and Schütze, H. (2018). Embedding learning through multilingual concept induction. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1520–1530, Melbourne, Australia. Association for Computational Linguistics.

Duong, L., Kanayama, H., Ma, T., Bird, S., and Cohn, T. (2017). Multilingual training of crosslingual word embeddings. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 894–904, Valencia, Spain. Association for Computational Linguistics.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. D. (2016). Calibrating noise to sensitivity in private data analysis. *J. Priv. Confidentiality*, 7(3):17–51.

Eberhard, D. M., Simons, G. F., and Fennig, C. D., editors (2024). *Ethnologue: Languages of the World*. SIL International, Dallas, Texas, twenty-seventh edition.

Ebrahimi, A. and Kann, K. (2021). How to adapt your pretrained multilingual model to 1600 languages. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4555–4567, Online. Association for Computational Linguistics.

Ebrahimi, A. and von der Wense, K. (2024). Zero-shot vs. translation-based crosslingual transfer: The case of lexical gaps. In Duh, K., Gomez, H., and Bethard, S., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 443–458, Mexico City, Mexico. Association for Computational Linguistics.

Eder, T., Hangya, V., and Fraser, A. (2021). Anchor-based bilingual word embeddings for low-resource languages. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*

(Volume 2: Short Papers), pages 227–232, Online. Association for Computational Linguistics.

Eisenschlos, J., Ruder, S., Czapla, P., Kadras, M., Gugger, S., and Howard, J. (2019). MultiFiT: Efficient multi-lingual language model fine-tuning. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5702–5707, Hong Kong, China. Association for Computational Linguistics.

Etxaniz, J., Azkune, G., Soroa, A., Lacalle, O., and Artetxe, M. (2024). Do multilingual language models think better in English? In Duh, K., Gomez, H., and Bethard, S., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 550–564, Mexico City, Mexico. Association for Computational Linguistics.

Fan, A., Bhosale, S., Schwenk, H., Ma, Z., El-Kishky, A., Goyal, S., Baines, M., Celebi, O., Wenzek, G., Chaudhary, V., Goyal, N., Birch, T., Liptchinsky, V., Edunov, S., Auli, M., and Joulin, A. (2021). Beyond english-centric multilingual machine translation. *J. Mach. Learn. Res.*, 22:107:1–107:48.

Fei, H., Zhang, M., and Ji, D. (2020). Cross-lingual semantic role labeling with high-quality translated training corpus. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7014–7026, Online. Association for Computational Linguistics.

Fleisig, E., Abebe, R., and Klein, D. (2023). When the majority is wrong: Modeling annotator disagreement for subjective tasks. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6715–6726, Singapore. Association for Computational Linguistics.

François, A. (2008). Semantic maps and the typology of colexification. *From polysemy to semantic change: Towards a typology of lexical semantic associations*, 106:163.

French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Gala, J., Gandhi, D., Mehta, J., and Talat, Z. (2023). A federated approach for hate speech detection. In Vlachos, A. and Augenstein, I., editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational*

Linguistics, pages 3248–3259, Dubrovnik, Croatia. Association for Computational Linguistics.

Gambäck, B. and Sikdar, U. K. (2017). Using convolutional neural networks to classify hate-speech. In Waseem, Z., Chung, W. H. K., Hovy, D., and Tetreault, J. R., editors, *Proceedings of the First Workshop on Abusive Language Online, ALW@ACL 2017, Vancouver, BC, Canada, August 4, 2017*, pages 85–90. Association for Computational Linguistics.

Gao, T., Fisch, A., and Chen, D. (2021). Making pre-trained language models better few-shot learners. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. (2020). Inverting gradients - how easy is it to break privacy in federated learning? In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Gerz, D., Vulić, I., Ponti, E. M., Reichart, R., and Korhonen, A. (2018). On the relation between linguistic typology and (limitations of) multilingual language modeling. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 316–327, Brussels, Belgium. Association for Computational Linguistics.

Good, J. (2018). Reflections on the scope of language documentation. In McDonnell, B., Berez-Kroeker, A. L., and Holton, G., editors, *Reflections on Language Documentation 20 Years after Himmelmann 1998*, number 15 in Language Documentation & Conservation Special Publication, pages 13–21. University of Hawai'i Press, Honolulu.

Gouws, S., Bengio, Y., and Corrado, G. (2015). Bilbowa: Fast bilingual distributed representations without word alignments. In Bach, F. R. and Blei, D. M., editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML*

2015, Lille, France, 6-11 July 2015, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 748–756. JMLR.org.

Gouws, S. and Søgaard, A. (2015). Simple task-specific bilingual word embeddings. In Mihalcea, R., Chai, J., and Sarkar, A., editors, *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1386–1390, Denver, Colorado. Association for Computational Linguistics.

Goyal, N., Gao, C., Chaudhary, V., Chen, P.-J., Wenzek, G., Ju, D., Krishnan, S., Ranzato, M., Guzmán, F., and Fan, A. (2022a). The Flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Goyal, N., Kivlichan, I. D., Rosen, R., and Vasserman, L. (2022b). Is your toxicity my toxicity? exploring the impact of rater identity on toxicity annotation. *Proc. ACM Hum. Comput. Interact.*, 6(CSCW2):1–28.

Grimminger, L. and Klinger, R. (2021). Hate towards the political opponent: A Twitter corpus study of the 2020 US elections on the basis of offensive speech and stance detection. In De Clercq, O., Balahur, A., Sedoc, J., Barriere, V., Tafreshi, S., Buechel, S., and Hoste, V., editors, *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 171–180, Online. Association for Computational Linguistics.

Guest, E., Vidgen, B., Mittos, A., Sastry, N., Tyson, G., and Margetts, H. (2021). An expert annotated dataset for the detection of online misogyny. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1336–1350, Online. Association for Computational Linguistics.

Gupta, U., Kim, Y. G., Lee, S., Tse, J., Lee, H. S., Wei, G., Brooks, D., and Wu, C. (2022). Chasing carbon: The elusive environmental footprint of computing. *IEEE Micro*, 42(4):37–47.

Hammarström, H., Forkel, R., Haspelmath, M., and Bank, S. (2022). Glottolog database 4.7. `https://doi.org/10.5281/zenodo.7398962`.

Hard, A., Rao, K., Mathews, R., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. (2018). Federated learning for mobile keyboard prediction. *CoRR*, abs/1811.03604.

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

Hartmann, J., Heitmann, M., Schamp, C., and Netzer, O. (2021). The power of brand selfies. *Journal of Marketing Research*.

Haspelmath, M. (2003). The geometry of grammatical meaning: Semantic maps and cross-linguistic comparison. In *The new psychology of language*, pages 217–248. Psychology Press.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. Psychology Press.

Hermann, K. M. and Blunsom, P. (2014). Multilingual models for compositional distributed semantics. In Toutanova, K. and Wu, H., editors, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 58–68, Baltimore, Maryland. Association for Computational Linguistics.

Himmelmann, N. P. (2005). The austronesian languages of asia and madagascar: typological characteristics. *The Austronesian languages of Asia and Madagascar*, 110:110–181.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

Hitaj, B., Ateniese, G., and Pérez-Cruz, F. (2017). Deep models under the GAN: information leakage from collaborative deep learning. In Thuraisingham, B., Evans, D., Malkin, T., and Xu, D., editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 603–618. ACM.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1):31.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Hoeken, S., Zarrieß, S., and Alacam, Ö. (2024). Hateful word in context classification. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 172–186, Miami, Florida, USA. Association for Computational Linguistics.

Holman, E. W., Wichmann, S., Brown, C. H., Velupillai, V., Müller, A., and Bakker, D. (2008). Explorations in automated language classification. *Folia Linguistica*, 42(3-4):331–354.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. (2019). Parameter-efficient transfer learning for NLP. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Hu, J., Ruder, S., Siddhant, A., Neubig, G., Firat, O., and Johnson, M. (2020). XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Hua, T., Yun, T., and Pavlick, E. (2024). mOthello: When do cross-lingual representation alignment and cross-lingual transfer emerge in multilingual models? In Duh, K., Gomez, H., and Bethard, S., editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1585–1598, Mexico City, Mexico. Association for Computational Linguistics.

Huang, H., Liang, Y., Duan, N., Gong, M., Shou, L., Jiang, D., and Zhou, M. (2019). Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2485–2494, Hong Kong, China. Association for Computational Linguistics.

Imani, A., Lin, P., Kargaran, A. H., Severini, S., Jalili Sabet, M., Kassner, N., Ma, C., Schmid, H., Martins, A., Yvon, F., and Schütze, H. (2023). Glot500: Scaling multilingual corpora and language models to 500 languages. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1082–1117, Toronto, Canada. Association for Computational Linguistics.

Ippolito, D., Tramer, F., Nasr, M., Zhang, C., Jagielski, M., Lee, K., Choquette Choo, C., and Carlini, N. (2023). Preventing generation of verbatim memorization in language

models gives a false sense of privacy. In Keet, C. M., Lee, H.-Y., and Zarrieß, S., editors, *Proceedings of the 16th International Natural Language Generation Conference*, pages 28–53, Prague, Czechia. Association for Computational Linguistics.

Jackson, J. C., Watts, J., Henry, T. R., List, J.-M., Forkel, R., Mucha, P. J., Greenhill, S. J., Gray, R. D., and Lindquist, K. A. (2019). Emotion semantics show both cultural variation and universal structure. *Science*, 366(6472):1517–1522.

Jalili Sabet, M., Dufter, P., Yvon, F., and Schütze, H. (2020). SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de Las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2023). Mistral 7b. *CoRR*, abs/2310.06825.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de Las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2024). Mixtral of experts. *CoRR*, abs/2401.04088.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. (2020). TinyBERT: Distilling BERT for natural language understanding. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Johnson, R. L., Pistilli, G., Menédez-González, N., Duran, L. D. D., Panai, E., Kalpokiene, J., and Bertulfo, D. J. (2022). The ghost in the machine has an american accent: value conflict in GPT-3. *CoRR*, abs/2203.07785.

Joshi, P., Santy, S., Budhiraja, A., Bali, K., and Choudhury, M. (2020). The state and fate of linguistic diversity and inclusion in the NLP world. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Kairouz, P., Liu, Z., and Steinke, T. (2021). The distributed discrete gaussian mechanism for federated learning with secure aggregation. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021,*

18-24 July 2021, Virtual Event, volume 139 of *Proceedings of Machine Learning Research*, pages 5201–5212. PMLR.

Kamholz, D., Pool, J., and Colowick, S. (2014). PanLex: Building a resource for panlingual lexical translation. In Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3145–3150, Reykjavik, Iceland. European Language Resources Association (ELRA).

Kanclerz, K., Gruza, M., Karanowski, K., Bielaniewicz, J., Milkowski, P., Kocon, J., and Kazienko, P. (2022). What if ground truth is subjective? personalized deep neural hate speech detection. In Abercrombie, G., Basile, V., Tonelli, S., Rieser, V., and Uma, A., editors, *Proceedings of the 1st Workshop on Perspectivist Approaches to NLP @LREC2022*, pages 37–45, Marseille, France. European Language Resources Association.

Karamolegkou, A., Li, J., Zhou, L., and Søgaard, A. (2023). Copyright violations and large language models. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7403–7412, Singapore. Association for Computational Linguistics.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. (2020). SCAFFOLD: stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR.

Keung, P., Lu, Y., Szarvas, G., and Smith, N. A. (2020). The multilingual Amazon reviews corpus. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568, Online. Association for Computational Linguistics.

Kim, J.-K., Kim, Y.-B., Sarikaya, R., and Fosler-Lussier, E. (2017). Cross-lingual transfer learning for POS tagging without cross-lingual resources. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2832–2838, Copenhagen, Denmark. Association for Computational Linguistics.

Kim, S., Yun, S., Lee, H., Gubri, M., Yoon, S., and Oh, S. J. (2023). Propile: Probing privacy leakage in large language models. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Klementiev, A., Titov, I., and Bhattarai, B. (2012). Inducing crosslingual distributed representations of words. In Kay, M. and Boitet, C., editors, *Proceedings of COLING 2012*, pages 1459–1474, Mumbai, India. The COLING 2012 Organizing Committee.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. (2019). Similarity of neural network representations revisited. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3519–3529. PMLR.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114.

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Blanco, E. and Lu, W., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Kuratov, Y. and Arkhipov, M. Y. (2019). Adaptation of deep bidirectional multilingual transformers for russian language. *CoRR*, abs/1905.07213.

Lample, G., Conneau, A., Denoyer, L., and Ranzato, M. (2018a). Unsupervised machine translation using monolingual corpora only. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Lample, G., Conneau, A., Ranzato, M., Denoyer, L., and Jégou, H. (2018b). Word translation without parallel data. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2020). ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Laurençon, H., Saulnier, L., Wang, T., Akiki, C., del Moral, A. V., Scao, T. L., von Werra, L., Mou, C., Ponferrada, E. G., Nguyen, H., Frohberg, J., Sasko, M., Lhoest, Q., McMillan-Major, A., Dupont, G., Biderman, S., Rogers, A., Allal, L. B., Toni, F. D., Pistilli, G., Nguyen, O., Nikpoor, S., Masoud, M., Colombo, P., de la Rosa, J., Villegas, P., Thrush, T., Longpre, S., Nagel, S., Weber, L., Muñoz, M., Zhu, J., van Strien, D., Alyafeai, Z., Almubarak, K., Vu, M. C., Gonzalez-Dios, I., Soroa, A., Lo, K., Dey, M., Suarez, P. O., Gokaslan, A., Bose, S., Adelani, D. I., Phan, L., Tran, H., Yu, I., Pai, S., Chim, J., Lepercq, V., Ilic, S., Mitchell, M., Luccioni, A. S., and Jernite, Y. (2022). The bigscience ROOTS corpus: A 1.6tb composite multilingual dataset. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Lauscher, A., Ravishankar, V., Vulić, I., and Glavaš, G. (2020). From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Lazaridou, A., Dinu, G., and Baroni, M. (2015). Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In Zong, C. and Strube, M., editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 270–280, Beijing, China. Association for Computational Linguistics.

Lazaridou, A., Kuncoro, A., Gribovskaya, E., Agrawal, D., Liska, A., Terzi, T., Gimenez, M., de Masson d'Autume, C., Kociský, T., Ruder, S., Yogatama, D., Cao, K., Young, S., and Blunsom, P. (2021). Mind the gap: Assessing temporal generalization in neural language models. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 29348–29363.

Le, H., Vial, L., Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L., and Schwab, D. (2020). FlauBERT: Unsupervised language model

pre-training for French. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France. European Language Resources Association.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324.

Lee, J., Hwang, S.-w., and Kim, T. (2022). FAD-X: Fusing adapters for cross-lingual transfer to low-resource languages. In He, Y., Ji, H., Li, S., Liu, Y., and Chang, C.-H., editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 57–64, Online only. Association for Computational Linguistics.

Lenc, K. and Vedaldi, A. (2015). Understanding image representations by measuring their equivariance and equivalence. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 991–999. IEEE Computer Society.

Lesage, J., Haynie, H. J., Skirgård, H., Weber, T., and Witzlack-Makarevich, A. (2022). Overlooked data in typological databases: What grambank teaches us about gaps in grammars. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2884–2890, Marseille, France. European Language Resources Association.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020a). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Lewis, M. P. and Simons, G. F. (2016). *Sustaining Language Use: Perspectives on Community-Based Language Development*. SIL International, Dallas, TX.

Lewis, P., Oguz, B., Rinott, R., Riedel, S., and Schwenk, H. (2020b). MLQA: Evaluating cross-lingual extractive question answering. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for*

Computational Linguistics, pages 7315–7330, Online. Association for Computational Linguistics.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. (2020). On the convergence of fedavg on non-iid data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Liang, D., Gonen, H., Mao, Y., Hou, R., Goyal, N., Ghazvininejad, M., Zettlemoyer, L., and Khabsa, M. (2023). XLM-V: Overcoming the vocabulary bottleneck in multilingual masked language models. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13142–13152, Singapore. Association for Computational Linguistics.

Libovický, J., Rosa, R., and Fraser, A. (2020). On the language neutrality of pre-trained multilingual representations. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1663–1674, Online. Association for Computational Linguistics.

Lin, P., Hu, C., Zhang, Z., Martins, A., and Schuetze, H. (2024). mPLM-sim: Better cross-lingual similarity and transfer in multilingual pretrained language models. In Graham, Y. and Purver, M., editors, *Findings of the Association for Computational Linguistics: EACL 2024*, pages 276–310, St. Julian's, Malta. Association for Computational Linguistics.

Lin, X. V., Mihaylov, T., Artetxe, M., Wang, T., Chen, S., Simig, D., Ott, M., Goyal, N., Bhosale, S., Du, J., Pasunuru, R., Shleifer, S., Koura, P. S., Chaudhary, V., O'Horo, B., Wang, J., Zettlemoyer, L., Kozareva, Z., Diab, M., Stoyanov, V., and Li, X. (2022). Few-shot learning with multilingual generative language models. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Lin, Y.-H., Chen, C.-Y., Lee, J., Li, Z., Zhang, Y., Xia, M., Rijhwani, S., He, J., Zhang, Z., Ma, X., Anastasopoulos, A., Littell, P., and Neubig, G. (2019). Choosing transfer languages for cross-lingual learning. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy. Association for Computational Linguistics.

List, J.-M., Terhalle, A., and Urban, M. (2013). Using network approaches to enhance the analysis of cross-linguistic polysemies. In Koller, A. and Erk, K., editors, *Proceedings of the 10th International Conference on Computational Semantics*

(IWCS 2013) – Short Papers, pages 347–353, Potsdam, Germany. Association for Computational Linguistics.

List, J. M., Tjuka, A., van Zantwijk, M., Blum, F., Ugarte, C. B., Rzymski, C., Greenhill, S., and Forkel, R., editors (2024). *CLLD Concepticon 3.2.0*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Littell, P., Mortensen, D. R., Lin, K., Kairis, K., Turner, C., and Levin, L. (2017). URIEL and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 8–14, Valencia, Spain. Association for Computational Linguistics.

Liu, F., Vulić, I., Korhonen, A., and Collier, N. (2021). Fast, effective, and self-supervised: Transforming masked language models into universal lexical and sentence encoders. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1459, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Liu, Y., Jabbar, H., and Schuetze, H. (2022). Flow-adapter architecture for unsupervised machine translation. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1253–1266, Dublin, Ireland. Association for Computational Linguistics.

Liu, Y., Lin, P., Wang, M., and Schuetze, H. (2024a). OFA: A framework of initializing unseen subword embeddings for efficient large-scale multilingual continued pretraining. In Duh, K., Gomez, H., and Bethard, S., editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1067–1097, Mexico City, Mexico. Association for Computational Linguistics.

Liu, Y., Ma, C., Ye, H., and Schuetze, H. (2024b). TransliCo: A contrastive learning framework to address the script barrier in multilingual pretrained language models. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2476–2499, Bangkok, Thailand. Association for Computational Linguistics.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Liu, Y., Ye, H., Weissweiler, L., Pei, R., and Schuetze, H. (2023a). Crosslingual transfer learning for low-resource languages based on multilingual colexification graphs. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8376–8401, Singapore. Association for Computational Linguistics.

Liu, Y., Ye, H., Weissweiler, L., Wicke, P., Pei, R., Zangenfeind, R., and Schütze, H. (2023b). A crosslingual investigation of conceptualization in 1335 languages. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12969–13000, Toronto, Canada. Association for Computational Linguistics.

Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Lukas, N., Salem, A., Sim, R., Tople, S., Wutschitz, L., and Béguelin, S. Z. (2023). Analyzing leakage of personally identifiable information in language models. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, pages 346–363. IEEE.

Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In Màrquez, L., Callison-Burch, C., and Su, J., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Ma, C., Imani, A., Ye, H., Asgari, E., and Schütze, H. (2023). Taxi1500: A multilingual dataset for text classification in 1500 languages. *CoRR*, abs/2305.08487.

Mager, M., Mager, E., Kann, K., and Vu, N. T. (2023). Ethical considerations for machine translation of indigenous languages: Giving a voice to the speakers. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4871–4897, Toronto, Canada. Association for Computational Linguistics.

Malaviya, C., Neubig, G., and Littell, P. (2017). Learning language representations for typology prediction. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2529–2535, Copenhagen, Denmark. Association for Computational Linguistics.

Malmasi, S. and Zampieri, M. (2017). Detecting hate speech in social media. In Mitkov, R. and Angelova, G., editors, *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017, Varna, Bulgaria, September 2 - 8, 2017*, pages 467–472. INCOMA Ltd.

Mandl, T., Modha, S., Majumder, P., Patel, D., Dave, M., Mandalia, C., and Patel, A. (2019). Overview of the HASOC track at FIRE 2019: Hate speech and offensive content identification in indo-european languages. In Majumder, P., Mitra, M., Gangopadhyay, S., and Mehta, P., editors, *FIRE '19: Forum for Information Retrieval Evaluation, Kolkata, India, December, 2019*, pages 14–17. ACM.

Maronikolakis, A., Wisiorek, A., Nann, L., Jabbar, H., Udupa, S., and Schuetze, H. (2022). Listening to affected communities to define extreme speech: Dataset and experiments. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1089–1104, Dublin, Ireland. Association for Computational Linguistics.

Marten, L. and Petzell, M. (2016). Linguistic variation and the dynamics of language documentation: Editing in 'pure' kagulu. In Seyfeddinipur, M., editor, *African Language Documentation: New Data, Methods and Approaches*, number 10 in Language Documentation & Conservation Special Publication, pages 105–129. University of Hawai'i at Mānoa, Honolulu.

Martins, A. and Astudillo, R. (2016). From softmax to sparsemax: A sparse model of attention and multi-label classification. In Balcan, M. F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1614–1623, New York, New York, USA. PMLR.

Mayer, T. and Cysouw, M. (2012). Language comparison through sparse multilingual word alignment. In Butt, M., Carpendale, S., Penn, G., Prokić, J., and Cysouw, M., editors, *Proceedings of the EACL 2012 Joint Workshop of LINGVIS & UNCLH*, pages 54–62, Avignon, France. Association for Computational Linguistics.

Mayer, T. and Cysouw, M. (2014). Creating a massively parallel Bible corpus. In Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3158–3163, Reykjavik, Iceland. European Language Resources Association (ELRA).

Mayhew, S., Tsai, C.-T., and Roth, D. (2017). Cheap translation for cross-lingual named entity recognition. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of*

the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2536–2545, Copenhagen, Denmark. Association for Computational Linguistics.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In Singh, A. and Zhu, X. J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR.

Meakins, F. (2013). Mixed languages. In Bakker, P. and Matras, Y., editors, *Contact Languages: A Comprehensive Guide*, pages 159–228. Mouton de Gruyter, Berlin.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In Bengio, Y. and LeCun, Y., editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Mikolov, T., Le, Q. V., and Sutskever, I. (2013b). Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013c). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

Minixhofer, B., Paischer, F., and Rekabsaz, N. (2022). WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3992–4006, Seattle, United States. Association for Computational Linguistics.

Minsky, M. and Papert, S. (1969). An introduction to computational geometry. *Cambridge tiass., HIT*, 479(480):104.

Moran, S. and McCloy, D., editors (2019). *PHOIBLE 2.0*. Max Planck Institute for the Science of Human History, Jena.

Moschella, L., Maiorca, V., Fumero, M., Norelli, A., Locatello, F., and Rodolà, E. (2023). Relative representations enable zero-shot latent space communication. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Mostafazadeh Davani, A., Díaz, M., and Prabhakaran, V. (2022). Dealing with disagreements: Looking beyond the majority vote in subjective annotations. *Transactions of the Association for Computational Linguistics*, 10:92–110.

Mozafari, M., Farahbakhsh, R., and Crespi, N. (2019). A bert-based transfer learning approach for hate speech detection in online social media. In Cherifi, H., Gaito, S., Mendes, J. F., Moro, E., and Rocha, L. M., editors, *Complex Networks and Their Applications VIII - Volume 1 Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019, Lisbon, Portugal, December 10-12, 2019*, volume 881 of *Studies in Computational Intelligence*, pages 928–940. Springer.

Mulki, H., Haddad, H., Bechikh Ali, C., and Alshabani, H. (2019). L-HSAB: A Levantine Twitter dataset for hate speech and abusive language. In Roberts, S. T., Tetreault, J., Prabhakaran, V., and Waseem, Z., editors, *Proceedings of the Third Workshop on Abusive Language Online*, pages 111–118, Florence, Italy. Association for Computational Linguistics.

Muller, B., Anastasopoulos, A., Sagot, B., and Seddah, D. (2021). When being unseen from mBERT is just the beginning: Handling new languages with multilingual language models. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 448–462, Online. Association for Computational Linguistics.

Muller, B., Sagot, B., and Seddah, D. (2020). Can multilingual language models transfer to an unseen dialect? A case study on north african arabizi. *CoRR*, abs/2005.00318.

Navigli, R. and Ponzetto, S. P. (2010). BabelNet: Building a very large multilingual semantic network. In Hajič, J., Carberry, S., Clark, S., and Nivre, J., editors, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 216–225, Uppsala, Sweden. Association for Computational Linguistics.

Navigli, R. and Ponzetto, S. P. (2012). Joining forces pays off: Multilingual joint word sense disambiguation. In Tsujii, J., Henderson, J., and Paşca, M., editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing*

and Computational Natural Language Learning, pages 1399–1410, Jeju Island, Korea. Association for Computational Linguistics.

Nie, E., Liang, S., Schmid, H., and Schütze, H. (2023). Cross-lingual retrieval augmented prompt for low-resource languages. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8320–8340, Toronto, Canada. Association for Computational Linguistics.

Nivre, J., Abrams, M., Agic, Z., Ahrenberg, L., Antonsen, L., et al. (2018). Universal dependencies 2.2 (2018). *URL http://hdl. handle. net/11234/1-1983xxx. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague, http://hdl. handle. net/11234/1-1983xxx*.

Nozza, D. (2021). Exposing the limits of zero-shot cross-lingual hate speech detection. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 907–914, Online. Association for Computational Linguistics.

Ogueji, K., Zhu, Y., and Lin, J. (2021). Small data? no problem! exploring the viability of pretrained multilingual language models for low-resourced languages. In Ataman, D., Birch, A., Conneau, A., Firat, O., Ruder, S., and Sahin, G. G., editors, *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 116–126, Punta Cana, Dominican Republic. Association for Computational Linguistics.

OpenAI (2023). GPT-4 technical report. *CoRR*, abs/2303.08774.

Östling, R. (2015). Word order typology through multilingual word alignment. In Zong, C. and Strube, M., editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 205–211, Beijing, China. Association for Computational Linguistics.

Östling, R. and Kurfali, M. (2023). Language embeddings sometimes contain typological generalizations. *Comput. Linguistics*, 49(4):1003–1051.

Östling, R. and Tiedemann, J. (2017). Continuous multilinguality with language vectors. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 644–649, Valencia, Spain. Association for Computational Linguistics.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller,

L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Palancar, E. L. (2016). A typology of tone and inflection: A view from the oto-manguean languages of mexico. *Tone and inflection: New facts and new perspectives*, pages 109–139.

Pan, L., Hang, C.-W., Qi, H., Shah, A., Potdar, S., and Yu, M. (2021). Multilingual BERT post-pretraining alignment. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 210–219, Online. Association for Computational Linguistics.

Pan, X., Zhang, B., May, J., Nothman, J., Knight, K., and Ji, H. (2017). Cross-lingual name tagging and linking for 282 languages. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Parikh, A., Täckström, O., Das, D., and Uszkoreit, J. (2016). A decomposable attention model for natural language inference. In Su, J., Duh, K., and Carreras, X., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Parović, M., Glavaš, G., Vulić, I., and Korhonen, A. (2022). BAD-X: Bilingual adapters improve zero-shot cross-lingual transfer. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1791–1799, Seattle, United States. Association for Computational Linguistics.

Patel, P., Choukse, E., Zhang, C., Goiri, Í., Warrier, B., Mahalingam, N., and Bianchini, R. (2023). POLCA: power oversubscription in LLM cloud providers. *CoRR*, abs/2308.12908.

Paulus, R., Xiong, C., and Socher, R. (2018). A deep reinforced model for abstractive summarization. In *6th International Conference on Learning Representations, ICLR*

2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net.

Pawar, S., Park, J., Jin, J., Arora, A., Myung, J., Yadav, S., Haznitrama, F. G., Song, I., Oh, A., and Augenstein, I. (2024). Survey of cultural awareness in language models: Text and beyond. *CoRR*, abs/2411.00860.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global vectors for word representation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Perrin, L.-M. (2010). Polysemous qualities and universal networks, invariance and diversity. *Linguistic Discovery*, 8:1–22.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Pfeiffer, J., Kamath, A., Rücklé, A., Cho, K., and Gurevych, I. (2021). AdapterFusion: Non-destructive task composition for transfer learning. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Pfeiffer, J., Vulić, I., Gurevych, I., and Ruder, S. (2020). MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Pham, T., Le, K., and Luu, A. T. (2024). UniBridge: A unified approach to cross-lingual transfer learning for low-resource languages. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3168–3184, Bangkok, Thailand. Association for Computational Linguistics.

Pitenis, Z., Zampieri, M., and Ranasinghe, T. (2020). Offensive language identification in Greek. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck,

T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5113–5119, Marseille, France. European Language Resources Association.

Ponti, E. M., O'Horan, H., Berzak, Y., Vulić, I., Reichart, R., Poibeau, T., Shutova, E., and Korhonen, A. (2019). Modeling language variation and universals: A survey on typological linguistics for natural language processing. *Computational Linguistics*, 45(3):559–601.

Price, I., Gifford-Moore, J., Flemming, J., Musker, S., Roichman, M., Sylvain, G., Thain, N., Dixon, L., and Sorensen, J. (2020). Six attributes of unhealthy conversations. In Akiwowo, S., Vidgen, B., Prabhakaran, V., and Waseem, Z., editors, *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 114–124, Online. Association for Computational Linguistics.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI Technical Report. `https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf`.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI Technical Report. `https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf`.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Rama, T., Beinborn, L., and Eger, S. (2020). Probing multilingual BERT for genetic and typological signals. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1214–1228, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ranasinghe, T. and Zampieri, M. (2021). MUDES: Multilingual detection of offensive spans. In Sil, A. and Lin, X. V., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 144–152, Online. Association for Computational Linguistics.

Ranasinghe, T. and Zampieri, M. (2022). Multilingual offensive language identification for low-resource languages. *ACM Trans. Asian Low Resour. Lang. Inf. Process.*, 21(1):4:1–4:13.

Rebuffi, S., Bilen, H., and Vedaldi, A. (2017). Learning multiple visual domains with residual adapters. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 506–516.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Roberts, J. S. and Montoya, L. N. (2023). In consideration of indigenous data sovereignty: Data mining as a colonial practice. In Arai, K., editor, *Proceedings of the Future Technologies Conference, FTC 2023, Vancouver, BC, Canada, 19-20 October 2023, Volume 2*, volume 814 of *Lecture Notes in Networks and Systems*, pages 180–196. Springer.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Röttger, P., Seelawi, H., Nozza, D., Talat, Z., and Vidgen, B. (2022). Multilingual HateCheck: Functional tests for multilingual hate speech detection models. In Narang, K., Mostafazadeh Davani, A., Mathias, L., Vidgen, B., and Talat, Z., editors, *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*, pages 154–169, Seattle, Washington (Hybrid). Association for Computational Linguistics.

Röttger, P., Vidgen, B., Nguyen, D., Waseem, Z., Margetts, H., and Pierrehumbert, J. (2021). HateCheck: Functional tests for hate speech detection models. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 41–58, Online. Association for Computational Linguistics.

Ruder, S., Constant, N., Botha, J., Siddhant, A., Firat, O., Fu, J., Liu, P., Hu, J., Garrette, D., Neubig, G., and Johnson, M. (2021). XTREME-R: Towards more challenging and nuanced multilingual evaluation. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10215–10245, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ruder, S., Vulic, I., and Søgaard, A. (2019). A survey of cross-lingual word embedding models. *J. Artif. Intell. Res.*, 65:569–631.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Rust, P., Pfeiffer, J., Vulić, I., Ruder, S., and Gurevych, I. (2021). How good is your tokenizer? on the monolingual performance of multilingual language models. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Rzymski, C., Tresoldi, T., Greenhill, S. J., Wu, M.-S., Schweikhard, N. E., Koptjevskaja-Tamm, M., Gast, V., Bodt, T. A., Hantgan, A., Kaiping, G. A., et al. (2020). The database of cross-linguistic colexifications, reproducible analysis of cross-linguistic polysemies. *Scientific data*, 7(1):13.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Sap, M., Card, D., Gabriel, S., Choi, Y., and Smith, N. A. (2019). The risk of racial bias in hate speech detection. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Sap, M., Gabriel, S., Qin, L., Jurafsky, D., Smith, N. A., and Choi, Y. (2020). Social bias frames: Reasoning about social and power implications of language. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online. Association for Computational Linguistics.

Sap, M., Swayamdipta, S., Vianna, L., Zhou, X., Choi, Y., and Smith, N. A. (2022). Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5884–5906, Seattle, United States. Association for Computational Linguistics.

Sarkar, D., Zampieri, M., Ranasinghe, T., and Ororbia, A. (2021). fBERT: A neural transformer for identifying offensive content. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1792–1798, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Scao, T. L., Fan, A., Akiki, C., Pavlick, E., Ilic, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., Gallé, M., Tow, J., Rush, A. M., Biderman, S., Webson, A., Ammanamanchi, P. S., Wang, T., Sagot, B., Muennighoff, N., del Moral, A. V., Ruwase, O., Bawden, R., Bekman, S., McMillan-Major, A., Beltagy, I., Nguyen, H., Saulnier, L., Tan, S., Suarez, P. O., Sanh, V., Laurençon, H., Jernite, Y., Launay, J., Mitchell, M., Raffel, C., Gokaslan, A., Simhi, A., Soroa, A., Aji, A. F., Alfassy, A., Rogers, A., Nitzav, A. K., Xu, C., Mou, C., Emezue, C., Klamm, C., Leong, C., van Strien, D., Adelani, D. I., and et al. (2022). BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2012, Kyoto, Japan, March 25-30, 2012*, pages 5149–5152. IEEE.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Shekhar, R., Karan, M., and Purver, M. (2022). CoRAL: a context-aware Croatian abusive language dataset. In He, Y., Ji, H., Li, S., Liu, Y., and Chang, C.-H., editors, *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 217–225, Online only. Association for Computational Linguistics.

Sheller, M. J., Edwards, B., Reina, G. A., Martin, J., Pati, S., Kotrotsou, A., Milchenko, M., Xu, W., Marcus, D., Colen, R. R., et al. (2020). Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific reports*, 10(1):12598.

Shliazhko, O., Fenogenova, A., Tikhonova, M., Kozlova, A., Mikhailov, V., and Shavrina, T. (2024). mGPT: Few-shot learners go multilingual. *Transactions of the Association for Computational Linguistics*, 12:58–79.

Singh, A. and Thakur, R. (2024). Generalizable multilingual hate speech detection on low resource Indian languages using fair selection in federated learning. In Duh, K., Gomez, H., and Bethard, S., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7211–7221, Mexico City, Mexico. Association for Computational Linguistics.

Skirgård, H., Haynie, H. J., Blasi, D. E., Hammarström, H., Collins, J., Latarche, J. J., Lesage, J., Weber, T., Witzlack-Makarevich, A., Passmore, S., et al. (2023). Grambank reveals the importance of genealogical constraints on linguistic diversity and highlights the impact of language loss. *Science Advances*, 9(16):eadg6175.

Socher, R., Bauer, J., Manning, C. D., and Ng, A. Y. (2013). Parsing with compositional vector grammars. In Schuetze, H., Fung, P., and Poesio, M., editors, *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria. Association for Computational Linguistics.

Song, K., Tan, X., Qin, T., Lu, J., and Liu, T. (2019). MASS: masked sequence to sequence pre-training for language generation. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Søren, W., Holman, E. W., and Brown, C. H. (2022). The ASJP database (version 20). `https://asjp.clld.org`.

Srinivasan, R. (2017). *Whose Global Village?: Rethinking How Technology Shapes Our World*. New York University Press, New York.

Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015). Training very deep networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2377–2385.

Stebbins, T. N., Eira, K., and Couzens, V. L. (2017). *Living Languages and New Approaches to Language Revitalisation Research*. Routledge, New York, 1st edition.

Swadesh, M. (2017). *The origin and diversification of language*. Routledge.

Tan, X., Chen, J., He, D., Xia, Y., Qin, T., and Liu, T.-Y. (2019). Multilingual neural machine translation with language clustering. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in*

Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 963–973, Hong Kong, China. Association for Computational Linguistics.

Tay, Y., Tran, V. Q., Ruder, S., Gupta, J. P., Chung, H. W., Bahri, D., Qin, Z., Baumgartner, S., Yu, C., and Metzler, D. (2022). Charformer: Fast character transformers via gradient-based subword tokenization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Thompson, B., Roberts, S. G., and Lupyan, G. (2020). Cultural influences on word meanings revealed through large-scale semantic alignment. *Nature Human Behaviour*, 4(10):1029–1038.

Tonneau, M., Liu, D., Fraiberger, S., Schroeder, R., Hale, S., and Röttger, P. (2024a). From languages to geographies: Towards evaluating cultural bias in hate speech datasets. In Chung, Y.-L., Talat, Z., Nozza, D., Plaza-del Arco, F. M., Röttger, P., Mostafazadeh Davani, A., and Calabrese, A., editors, *Proceedings of the 8th Workshop on Online Abuse and Harms (WOAH 2024)*, pages 283–311, Mexico City, Mexico. Association for Computational Linguistics.

Tonneau, M., Quinta De Castro, P., Lasri, K., Farouq, I., Subramanian, L., Orozco-Olvera, V., and Fraiberger, S. (2024b). NaijaHate: Evaluating hate speech detection on Nigerian Twitter using representative data. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9020–9040, Bangkok, Thailand. Association for Computational Linguistics.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023a). Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Canton-Ferrer, C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang,

S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. (2023b). Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Tran, K. M. (2020). From english to foreign languages: Transferring pre-trained language models. *CoRR*, abs/2002.07306.

Truex, S., Liu, L., Gursoy, M. E., Yu, L., and Wei, W. (2021). Demystifying membership inference attacks in machine learning as a service. *IEEE Trans. Serv. Comput.*, 14(6):2073–2089.

Turian, J., Ratinov, L.-A., and Bengio, Y. (2010). Word representations: A simple and general method for semi-supervised learning. In Hajič, J., Carberry, S., Clark, S., and Nivre, J., editors, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden. Association for Computational Linguistics.

Unanue, I. J., Haffari, G., and Piccardi, M. (2023). T3L: Translate-and-test transfer learning for cross-lingual text classification. *Transactions of the Association for Computational Linguistics*, 11:1147–1161.

Üstün, A., Aryabumi, V., Yong, Z., Ko, W.-Y., D'souza, D., Onilude, G., Bhandari, N., Singh, S., Ooi, H.-L., Kayid, A., Vargus, F., Blunsom, P., Longpre, S., Muennighoff, N., Fadaee, M., Kreutzer, J., and Hooker, S. (2024). Aya model: An instruction finetuned open-access multilingual language model. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15894–15939, Bangkok, Thailand. Association for Computational Linguistics.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Villalobos, P., Sevilla, J., Heim, L., Besiroglu, T., Hobbhahn, M., and Ho, A. (2022). Will we run out of data? an analysis of the limits of scaling datasets in machine learning. *CoRR*, abs/2211.04325.

Vulić, I. and Korhonen, A. (2016). On the role of seed lexicons in learning bilingual word embeddings. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 247–257, Berlin, Germany. Association for Computational Linguistics.

Vulic, I. and Moens, M. (2016). Bilingual distributed word representations from document-aligned comparable data. *J. Artif. Intell. Res.*, 55:953–994.

Vulić, I. and Moens, M.-F. (2015). Bilingual word embeddings from non-parallel document-aligned data applied to bilingual lexicon induction. In Zong, C. and Strube, M., editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 719–725, Beijing, China. Association for Computational Linguistics.

Vulić, I., Ruder, S., and Søgaard, A. (2020). Are all good word vector spaces isomorphic? In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3178–3192, Online. Association for Computational Linguistics.

Wang, D., Chen, J., Zhou, H., Qiu, X., and Li, L. (2021). Contrastive aligned joint learning for multilingual summarization. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2739–2750, Online. Association for Computational Linguistics.

Wang, K., Mathews, R., Kiddon, C., Eichner, H., Beaufays, F., and Ramage, D. (2019). Federated evaluation of on-device personalization. *CoRR*, abs/1910.10252.

Wang, M., Adel, H., Lange, L., Strötgen, J., and Schuetze, H. (2023a). GradSim: Gradient-based language grouping for effective multilingual training. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4631–4646, Singapore. Association for Computational Linguistics.

Wang, M., Adel, H., Lange, L., Strötgen, J., and Schütze, H. (2023b). NLNDE at SemEval-2023 task 12: Adaptive pretraining and source language selection for low-resource multilingual sentiment analysis. In Ojha, A. K., Doğruöz, A. S., Da San Martino, G., Tayyar Madabushi, H., Kumar, R., and Sartori, E., editors, *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 488–497, Toronto, Canada. Association for Computational Linguistics.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. (2020a). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Wang, Y., Mishra, S., Alipoormolabashi, P., Kordi, Y., Mirzaei, A., Naik, A., Ashok, A., Dhanasekaran, A. S., Arunkumar, A., Stap, D., Pathak, E., Karamanolakis, G.,

Lai, H., Purohit, I., Mondal, I., Anderson, J., Kuznia, K., Doshi, K., Pal, K. K., Patel, M., Moradshahi, M., Parmar, M., Purohit, M., Varshney, N., Kaza, P. R., Verma, P., Puri, R. S., Karia, R., Doshi, S., Sampat, S. K., Mishra, S., Reddy A, S., Patro, S., Dixit, T., and Shen, X. (2022). Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Wang, Z., K, K., Mayhew, S., and Roth, D. (2020b). Extending multilingual BERT to low-resource languages. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2649–2656, Online. Association for Computational Linguistics.

Waseem, Z. and Hovy, D. (2016). Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In Andreas, J., Choi, E., and Lazaridou, A., editors, *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Webb, T., Holyoak, K. J., and Lu, H. (2023). Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9):1526–1541.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. (2022a). Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022b). Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. (2022c). Chain-of-thought prompting elicits reasoning in large language models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Wei, X., Weng, R., Hu, Y., Xing, L., Yu, H., and Luo, W. (2021). On learning universal representations across languages. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Wen-Yi, A. W. and Mimno, D. (2023). Hyperpolyglot LLMs: Cross-lingual interpretability in token embeddings. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1124–1131, Singapore. Association for Computational Linguistics.

Wendler, C., Veselovsky, V., Monea, G., and West, R. (2024). Do llamas work in English? on the latent language of multilingual transformers. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394, Bangkok, Thailand. Association for Computational Linguistics.

Wenzek, G., Lachaux, M., Conneau, A., Chaudhary, V., Guzmán, F., Joulin, A., and Grave, E. (2020). Ccnet: Extracting high quality monolingual datasets from web crawl data. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 4003–4012. European Language Resources Association.

Woller, L., Hangya, V., and Fraser, A. (2021). Do not neglect related languages: The case of low-resource Occitan cross-lingual word embeddings. In Ataman, D., Birch, A., Conneau, A., Firat, O., Ruder, S., and Sahin, G. G., editors, *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 41–50, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wu, S. and Dredze, M. (2019). Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Wu, S. and Dredze, M. (2020). Are all languages created equal in multilingual BERT? In Gella, S., Welbl, J., Rei, M., Petroni, F., Lewis, P., Strubell, E., Seo, M., and Hajishirzi, H., editors, *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

Wulczyn, E., Thain, N., and Dixon, L. (2017). Ex machina: Personal attacks seen at scale. In Barrett, R., Cummings, R., Agichtein, E., and Gabrilovich, E., editors, *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 1391–1399. ACM.

Xhelili, O., Liu, Y., and Schuetze, H. (2024). Breaking the script barrier in multilingual pre-trained language models with transliteration-based post-training alignment. In

Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11283–11296, Miami, Florida, USA. Association for Computational Linguistics.

Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., and Raffel, C. (2022). ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. (2021). mT5: A massively multilingual pre-trained text-to-text transformer. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Yang, Y., Cer, D., Ahmad, A., Guo, M., Law, J., Constant, N., Hernandez Abrego, G., Yuan, S., Tar, C., Sung, Y.-h., Strope, B., and Kurzweil, R. (2020). Multilingual universal sentence encoder for semantic retrieval. In Celikyilmaz, A. and Wen, T.-H., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Young, H. (2015). The digital language divide. *The Guardian*. `http://labs.theguardian.com/digital-language-divide`.

Yu, D., He, T., and Sagae, K. (2021). Language embeddings for typology and cross-lingual transfer learning. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7210–7225, Online. Association for Computational Linguistics.

Zampieri, M., Premasiri, D., and Ranasinghe, T. (2024). A federated learning approach to privacy preserving offensive language identification. *CoRR*, abs/2404.11470.

Zhang, L., Wang, S., and Liu, B. (2018a). Deep learning for sentiment analysis: A survey. *WIREs Data Mining Knowl. Discov.*, 8(4).

Zhang, Z., Robinson, D., and Tepper, J. A. (2018b). Detecting hate speech on twitter using a convolution-gru based deep neural network. In Gangemi, A., Navigli, R., Vidal, M., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., and Alam, M., editors, *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 745–760. Springer.

Zhao, Y., Zhang, W., Wang, H., Kawaguchi, K., and Bing, L. (2024). Adamergex: Cross-lingual transfer with large language models via adaptive adapter merging. *CoRR*, abs/2402.18913.

Zhou, S., Shan, H., Plank, B., and Litschko, R. (2024). MaiNLP at SemEval-2024 task 1: Analyzing source language selection in cross-lingual textual relatedness. In Ojha, A. K., Doğruöz, A. S., Tayyar Madabushi, H., Da San Martino, G., Rosenthal, S., and Rosá, A., editors, *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1842–1853, Mexico City, Mexico. Association for Computational Linguistics.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14747–14756.

Zhu, Y., Kiros, R., Zemel, R. S., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

Zoph, B., Yuret, D., May, J., and Knight, K. (2016). Transfer learning for low-resource neural machine translation. In Su, J., Duh, K., and Carreras, X., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.