Digitale Tiefenerschließung traditioneller Lexikographie - am Beispiel des Romanischen Etymologischen Wörterbuchs

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

vorgelegt von
Florian Zacherl
aus Gräfelfing

2023

Erstgutachter: Prof. Dr. Thomas Krefeld

Zweitgutachter: Prof. Dr. Hinrich Schütze

Datum der mündlichen Prüfung: 27.09.2022

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Viele traditionelle linguistische Ressourcen sind inzwischen als frei verfügbare Web-Portale in der digitalen Welt vertreten (vgl. z.B. TLIO, FEW, DWDS, DRG, NavigAIS, SchweizId.). Die Art und vor allem die Tiefe der Erschließung ist dabei allerdings höchst unterschiedlich. Während manche Portale aus kaum mehr als den gescannten Seiten der Quelle bestehen, bieten andere stärker verarbeitete Daten an, auf deren Basis zusätzliche neue Funktionalitäten aufbauen, die im gedruckten Werk nicht möglich waren. Lücke 2016 unterscheidet in diesem Kontext drei „Digitalisierungsgrade“, die auf Stufe 1 mit dem (bildbasierten) Scan beginnen und über den „linearisierte[n] elektronische[n] Text“ bis hin zu Stufe 3, den strukturierten Daten, reichen. Auch Digitalisierungen der höchsten Stufe können sich allerdings durchaus substanziell in ihrer Ausprägung unterscheiden. Eine häufige „Grenze“ für die digitale Erschließung von Wörterbüchern sind das Lemma und der zugehörige Artikeltext. Der Text selbst wird zum Teil zwar verarbeitet, indem bestimmte Bestandteile entsprechend ihrer Funktion annotiert werden (vgl. z.B. Renders 2011, Tasovac 2020, Burch und Rapp 2006), bleibt aber als solcher bestehen. Diese Annotationen haben dabei durchaus ihre Berechtigung, da sie gewisse analytische Abfragen erlauben. Es stellt sich allerdings die Frage, wie weitreichend die Möglichkeiten dabei tatsächlich sind. Das folgende Zitat ist in dieser Hinsicht interessant:

> Die Nutzer/innen können jede mögliche Anfrage stellen, die sich aus der Strukturierung der Daten ergibt [...].(Neumann 2007, S. 650)

Hier wird nicht von „jeder Anfrage, die sich aus den Daten ergibt“ gesprochen, sondern explizit auf deren Struktur Bezug genommen. Diese Einschränkung ist bei annotiertem Text durchaus vorhanden, da dieser weiterhin die durch den Autor verwendete Perspektive einnimmt und anderweitige Zugänge zumindest schwierig sind. Für die technische Verarbeitung der Daten, die das Potential hat große Teile der immer noch häufig vorhanden manuellen Quellenrecherche in der Linguistik zu ersetzen, ist allerdings ein möglichst uneingeschränkter Zugriff auf die jeweiligen Informationen notwendig. Besonders entscheidend ist dies bei Vernetzungsprojekten wieder der *Linguistic Linked Open Data Cloud* (vgl. Chiarcos, Hellmann und Nordhoff 2011).

Diese Arbeit beschäftigt sich mit der Frage wie traditionelle lexikographische Ressourcen möglichst feingranuliert erschlossen und deren Informationsgehalt möglichst vielseitig und unaufwendig zu Forschungszwecken bereitgestellt werden kann.

Sie beinhaltet dabei eine intensive Analyse traditioneller Lexikographie und wie deren Normen und Konventionen technisch ausgelesen und abgebildet werden können, und konzipiert Mechanismen und Strukturen, um mit diesem Material im wissenschaftlichen Kontext zu arbeiten. Der Fokus ist dabei ein sehr methodischer, so wird der vollständige Prozess vom Scan des Quellenmaterials, über die Transformation in strukturierte Daten, bis hin zur Publikation und Vernetzung betrachtet, wobei in jedem Abschnitt charakteristische Problematiken analysiert und entsprechende Lösungsansätze vorgestellt werden. Die Implementierungen für die Behandlung bestimmter Teilprobleme sind dabei in einem gewissen Maße als exemplarisch zu verstehen und könnten wohl zum Teil durch effizientere Ansätze aus der Informatik oder Computerlinguistik ersetzt werden. Die einzelnen Themengebiete werden also nicht unbedingt immer in größtmöglicher Tiefe behandelt, sondern mehr deren Zusammenspiel untereinander, das die Basis für eine übergeordnete Konzeption liefert, die an allen Stellen zum Tragen kommt. Die Perspektive ist also eine umfassendere, als sie in ähnlichen Arbeiten bisher betrachtet wurde, die beispielsweise durchaus eine intensive Quellenanalyse (vgl. z.B. Tasovac 2020) oder Überlegungen zur strukturierten Modellierung linguistischer Daten (vgl. z.B. Zimmermann 2006) angestellt haben, aber weniger die technische und konzeptuelle Herangehensweise in ihrer Gesamtheit betrachtet haben.

Als Beispiel für die Veranschaulichung der Überlegungen dient das *Romanische Etymologische Wörterbuch (REW)* in der dritten Auflage von 1935. Der Fokus liegt somit auf den dort enthaltenen lexikalischen Daten und den etymologischen Zusammenhängen, auch wenn sich viele Konzepte auf allgemeinere Anwendungsfälle übertragen lassen. Für die technische Umsetzung wurde ausschließlich freie Software verwendet, den Kern bildet hier Webtechnologie unter Verwendung des Content Management Systems Wordpress.

Der Aufbau der Arbeit kann grob in vier Teilen dargestellt werden. Die ersten Kapitel beschäftigen sich mit den abstrakten Grundlagen, so werden theoretische Überlegungen über die Eigenschaften von lexikalischen Daten und ihrer Darstellung (Kapitel 2), sowie zum Prozess der Erzeugung von strukturiertem Datenmaterial aus textuellen Rohdaten (Kapitel 3) angestellt. Im zweiten Teil steht dann dieser Prozess als solcher im Vordergrund, dessen drei Hauptphasen in den Kapiteln 4, 5 und 7 beschrieben werden, während Kapitel 6 einen Einschub zur Betrachtung der Datenmodellierung vor allem der Resultatdaten darstellt. Im weiteren werden zusätzliche sinnvolle Arbeitsschritte auf Basis dieser Daten vorgestellt. Dabei geht es einerseits um die Korrektur und Verbesserung von diesen (Kapitel 8) und andererseits um Bündelung und generellen Umgang mit sprachlichen Formen (Kapitel 9) und Bedeutungsangaben (Kapitel 10). Das letzte Kapitel des dritten Teils (Kapitel 11) beschäftigt sich mit der Vernetzung und Anreicherung von Entitäten, die weniger zentral sind, als die der vorherigen beiden, aber durchaus das Potential haben weiteren Mehrwert der digitalen Darstellung zu bieten. Zuletzt wird die Veröffentlichung der Daten betrachtet, die zum einem aus dem Webportal mit unterschiedlichen Recherchefunktionen (Kapitel 12) und

andererseits aus verschiedenformatigen Datenexporten (Kapitel 13) besteht.

Anmerkungen:

Teile dieser Arbeit wurden bereits in zwei Artikeln besprochen (Zacherl In Vorb. und Zacherl 2022), die zur Publikation eingereicht wurden und sich in verschiedenen Stadien des redaktionellen Prozesses befinden.

Das parallel zur Arbeit entwickelte Web-Portal ist unter rew-online.gwi.uni-muenchen.de/ zugreifbar. Der verwendete Programmcode wird unter https://gitlab.lrz.de/rew-online veröffentlicht. Zum Zeitpunkt der Einreichung sind noch gewisse Einschränkungen vorhanden. So sind die *gitlab*-Repositorien aktuell nur für LMU-Mitarbeiter zugreifbar, eine allgemeine Veröffentlichung muss noch beantragt werden. Auch sind die dort enthaltenen Daten noch nicht zur Gänze vollständig. Das Web-Portal ist in weiten Teilen funktional. Einige einzelne Elemente sind noch nicht vollständig fertiggestellt, dies betrifft vor allem den Zugriff auf die erweiterten Suchfunktionalitäten über die Eingabemaske, den Download der verschiedenen Exportformate und die Verlinkung aller Seiten an der Oberfläche. Auch eine öffentliche Registrierung ist zum aktuellen Zeitpunkt noch nicht möglich.

2 Ein allgemeines Datenmodell für lexikalische Daten

Ein wichtiger Bestandteil dieser Arbeit ist die Erzeugung eines strukturierten Datensatzes aus einer textuellen linguistischen Ressource. Dieses Kapitel beschäftigt sich somit mit der grundlegenden Frage, welche Art von Information ein solches Werk enthält und wie diese generalisiert und modelliert werden kann. Für den Gesamtbestand dieser extrahierten Kerndaten wird hier der Begriff *lexikalische Daten* verwendet, wobei dessen Verwendung vom gängigen Verständnis dieses Begriffs abweicht. Im Kontext der *Digital Humanities* wird dieser oftmals über die Herkunft der Daten definiert, d.h. als *lexikalische Daten* werden solche bezeichnet, die aus einem Wörterbuch stammen. Exemplarisch hierfür ist die folgende Definition:

> Lexikalische Datenbanken sind digitale lexikalische Ressourcen, die in einer Form abgespeichert sind, dass die einzelnen Datensätze konsistent im Hinblick auf eine formale Beschreibung ihrer Struktur sind. Ein einzelner Datensatz kann dabei einem Wörterbuchartikel entsprechen oder einem Artikelteil. Er kann aber auch artikelübergreifende Strukturen umfassen. [...](Kunze und Lemnitzer 2007, S. 12)

Hier wird also weniger die Information an sich in der Vordergrund gestellt, die aus einem Wörterbuch erschlossen werden kann, sondern der Fokus auf die digitale Repräsentation eines solchen in einer Datenbank gelegt. Dieser Grundgedanke spiegelt sich oftmals auch in etablierten technischen Standards wider, deren Strukturen sich an den traditionellen Bestandteilen von Wörterbüchern orientieren. Im Gegensatz dazu soll hier ein anderer Ansatz vorgeschlagen werden, der bestimmt, welche grundlegenden Informationen in einem Wörterbuchtext kodiert sind und daraus ein möglichst generalisiertes abstraktes Datenmodell erstellt, das für diesen Typus von Daten verwendet werden kann, unabhängig davon aus welcher Quelle sie stammen. Dazu werden zunächst die beiden großen traditionellen Publikationsformen der Geolinguistik und der Lexikographie, der Sprachatlas und das Wörterbuch, aus informationstheoretischer Sicht näher beleuchtet und deren inhaltliche Gemeinsamkeiten herausgearbeitet (Kap. 2.1). Darauf aufbauend wird ein grundlegendes Datenmodell als theoretisches Konstrukt (Kap. 2.2) entwickelt. Zuletzt werden vor diesem Hintergrund zwei in den *Digital Humanities* weit verbreitete Datenmodelle betrachtet und deren Anwendbarkeit vor diesem Hintergrund herausgearbeitet (Kap. 2.3).

2.1 Informationsgehalt von Sprachatlanten und Wörterbüchern

Dieses Kapitel betrachtet verschiedene traditionelle linguistische Ressourcen und vergleicht auf abstrakter Basis, welche Informationen daraus gewonnen werden können. Mit Information sind in diesem Fall strukturierte Daten gemeint, die einen gewissen Anspruch an Generalisierbarkeit aufweisen. Somit werden hier vor allem stark strukturierte Passagen betrachtet. Natürlichsprachige, diskursive Abschnitte (vgl. dazu Kap. 6.5.3) werden an dieser Stelle nicht behandelt.

In allen Beispielen werden Lemmata und sonstige sprachliche Formen identisch behandelt. Eine genauere Betrachtung dieser Vorgehensweise findet sind in Kap. 2.2.1. Weiterhin werden zum Teil Informationen verwendet, die nicht explizit in den Beispielen enthalten sind (z.B. Abkürzungsverzeichnisse, um Ortskürzel aufzulösen). Alle Angaben werden hier sehr informell dargestellt und dienen als Illustration und nicht als Vorlage für eine tatsächliche technische Umsetzung. So wäre beispielsweise die Angabe eines konkreten Transkriptionsschema sicher sinnvoller als die Markierung als „phonetisch“, um beispielsweise unterschiedliche Arten der phonetischen Transkription in verschiedenen Quellen voneinander abzugrenzen.

Begonnen wird mit folgendem Ausschnitt aus dem *Dicziunari Rumantsch Grischun (DRG)*, einem Wörterbuch des Bündnerromanischen. Dieses enthält in den meisten Artikeln eine lemmatisierte Grundform und eine Liste von phonetisch transkribierten Formen in den verschiedenen Gemeinden:

> **BÜMATSCH** m., uengad. ‘Schafbock, Widder’. E 10–25 *bümáč*, E 30–34 *jümáč*, *ǧümáč*, *ǧumáč* [...] (DRG, S. 2, 610)

In diesem Fall wird der Großteil der Information des Lemmas auf die einzelnen Varianten „vererbt“, sodass eine tabellarische Aufstellung unter Auflösung der Ortsabkürzungen beispielsweise folgendes Format haben könnte:

Form	**Transkription**	**Sprache / Dialekt**	**Genus**	**Bedeutung**	**Ort**
bümatsch	Orthographie	unterengadinisch	maskulin	Schafbock, Widder	—
bümáč	phonetisch	unterengadinisch	maskulin	Schafbock, Widder	Tschlin, Martina, Strada, Ramosch, Vna, Sent, Scuol, Tarasp, Ftan, Ardez, Guarda, Lavin, Susch, Zernez
jümáč	phonetisch	unterengadinisch	maskulin	Schafbock, Widder	Tschierv, Fuldera, Lü, Valchava, Santa Maria
g̊ümáč	phonetisch	unterengadinisch	maskulin	Schafbock, Widder	Tschierv, Fuldera, Lü, Valchava, Santa Maria
g̊umáč	phonetisch	unterengadinisch	maskulin	Schafbock, Widder	Tschierv, Fuldera, Lü, Valchava, Santa Maria

Tabelle 2.1: Information aus dem Ausschnitt des DRG in tabellarischer Form

Diese Tabelle spiegelt den für Menschen begreifbaren Informationsgehalt des Artikels grundsätzlich gut wider, es fehlt allerdings noch die Information über den Zusammenhang zwischen dem Lemma und den phonetischen Formen. Diese Beziehung könnte beispielsweise in folgender Weise formalisiert werden:

Form 1	**Beziehung**	**Form 2**
bümáč	ist phonetische Variante	bümatsch
jümáč	ist phonetische Variante	bümatsch
g̊ümáč	ist phonetische Variante	bümatsch
g̊umáč	ist phonetische Variante	bümatsch

Tabelle 2.2: Beziehung zwischen den verwendeten Formen im Ausschnitt des DRG

Eine zweite wichtige Form von (geo-)linguistischer Information ist der Sprachatlas. Somit wird hier ein kleiner Ausschnitt aus dem *Sprach- und Sachatlas Italiens und der Südschweiz (AIS)* betrachtet (Die Abbildung ist aus dem Titel und einem Teil der eigentlichen Karte zusammengesetzt):

Abbildung 2.1: Ausschnitt aus Karte 1069 des AIS

Obwohl die Art der Darstellung in diesem Fall stark abweicht, können die dort enthalten Informationen in ähnlicher Form dargestellt werden:

Form	**Transkription**	**Sprache / Dialekt**	**Genus**	**Numerus**	**Bedeutung**	**Ort**
nú	phonetisch	bündnerromanisch[2]	maskulin	Singular	Schafbock	Brigels-Breil
nú s	phonetisch	bündnerromanisch	maskulin	Plural	Schafbock	Brigels-Breil
bó̱c	phonetisch	bündnerromanisch	maskulin	Singular	Schafbock	Ems-Domat
bó̱cs	phonetisch	bündnerromanisch	maskulin	Plural	Schafbock	Ems-Domat

Tabelle 2.3: Information aus dem Ausschnitt des AIS in tabellarischer Form

Die Artikel der einzelnen Formen werden an dieser Stelle nicht explizit angegeben und dienen nur als Mittel zur Numerusbestimmung. Falls deren phonetische Transkription oder sonstige Details der Verwendung relevant sind, könnten sie natürlich ebenfalls als einzelne Formen dargestellt werden. Auch in diesem Fall ist über die Art der Notation, die eine entsprechende Singular- und Pluralform angibt, eine weitere Form von (impliziter) Information gegeben. Die Zugehörigkeit zu einem Lexem (das an dieser Stelle nicht explizit repräsentiert wird) könnte also beispielsweise folgendermaßen dargestellt werden:

Form 1	Beziehung	Form 2
nú	gleiches Lexem	nú s
bǫ́ć	gleiches Lexem	bǫ́ćs

Tabelle 2.4: Beziehung zwischen den verwendeten Formen im Ausschnitt des AIS

Im Gegensatz zum vorherigen Beispiel findet hier keine hierarchische Bündlung über die die Verknüpfung zu einem „übergeordneten“ Worttyp statt, sondern eine Bündelung der Form durch eine Verknüpfung untereinander, die grundsätzliche Information ist aber ähnlich[3].

Das dritte Beispiel stammt aus dem REW selbst und besteht aus folgendem Ausschnitt:

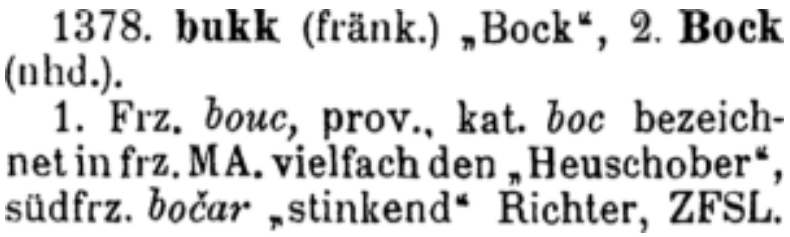

1378. **bukk** (fränk.) „Bock“, 2. **Bock** (nhd.).
1. Frz. *bouc*, prov., kat. *boc* bezeichnet in frz. MA. vielfach den „Heuschober“, südfrz. *bočar* „stinkend“ Richter, ZFSL.

Abbildung 2.2: Ausschnitt aus Eintrag 1378 des REW

Auch hier können die explizit vorhandenen Daten in ähnlicher Weise in einer Tabelle angegeben werden:

Form	Transkription	Sprache / Dialekt	Bedeutung	Ort
bukk	Orthographie	fränkisch	Bock	—
Bock	Orthographie	neuhochdeutsch	Bock	—
bouc	Orthographie	französisch	Bock	—
boc[1]	Orthographie	katalanisch	Bock	—
boc[2]	Orthographie	französische Mundart	Heuschober	—
bočar	phonetisch	französisch	stinkend	Südfrankreich

Tabelle 2.5: Information aus dem Ausschnitt des REW in tabellarischer Form

In diesem Fall wird eine Ortsangabe nur in dem Fall angegeben, in dem die Sprachabkürzung explizit geographische Information enthält. Je nach Anwendungsfall könnte es aber auch sinnvoll sein beispielsweise den deutschsprachigen Raum als räumliche Zuordnung zur hochdeutschen Form zu verwenden (vgl. Kap. 11.2). Wiederum ist auch hier ein Teil der Information nicht explizit angeben, nämlich die etymologischen Beziehungen zwischen den einzelnen Formen (im Falle eines etymologischen Wörterbuchs ist dies vielleicht sogar die wichtigste Information). Sie kann aber aus dem Grundaufbau des Artikels erschlossen werden:

[3] Eine Trennung von phonetischer Variation und morphologischen Unterschieden wird hier aus Vereinfachungsgründen nicht vorgenommen.

Form 1	Beziehung	Form 2
bouc	stammt ab von	bukk
boc[1]	stammt ab von	bukk
boc[2]	stammt ab von	bukk
bočar	stammt ab von	bukk

Tabelle 2.6: Beziehung zwischen den verwendeten Formen im Ausschnitt des REW

Der letzte Ausschnitt stammt nun aus dem *Oxford English Dictionary*. Im Unterschied zu den vorherigen Quellen sind hier explizite Zeiträume für bestimmte Formen angegeben:

> **buck, n.1**
>
> **Pronunciation:** Brit. /b k/ U.S. /bək/
>
> **Forms:** (sense 'he-goat') Old English **bucca**, Middle English **buc**, Middle English **bucke**, Middle English–1500s **bukke**; (senses 'male deer', etc.) Old English, Middle English **buc** [...](OED Online, buck, n.1)

Auch hier können die Informationen in einem entsprechenden tabellarischen Format dargestellt werden:

Form	Transkription	Sprache / Dialekt	Wortart	Bedeutung	Ort	Zeitraum
buck	Orthographie	englisch	Substantiv	—	—	modern
b k	phonetisch	englisch	Substantiv	—	Großbritannien	modern
bək	phonetisch	englisch	Substantiv	—	USA	modern
bucca	Orthographie	englisch	Substantiv	Ziegenbock	—	altenglisch
buc[1]	Orthographie	englisch	Substantiv	Ziegenbock	—	mittelenglisch
bucke	Orthographie	englisch	Substantiv	Ziegenbock	—	mittelenglisch
bukke	Orthographie	englisch	Substantiv	Ziegenbock	—	mittelenglisch bis 15. Jahrhundert
buc[2]	Orthographie	englisch	Substantiv	Hirsch	—	altenglisch, mittelenglisch

Tabelle 2.7: Information aus dem Ausschnitt des OED in tabellarischer Form

In diesem Fall sind sowohl phonetische Relationen analog zum DRG vorhanden, als auch unterschiedliche historische Schreibvarianten des modernen Lexems:

Form 1	**Beziehung**	**Form 2**
b k	ist phonetische Variante	buck
bək	ist phonetische Variante	buck
bucca	ist Schreibweise von	buck
buc[1]	ist Schreibweise von	buck
bucke	ist Schreibweise von	buck
bukke	ist Schreibweise von	buck
buc[2]	ist Schreibweise von	buck

Tabelle 2.8: Beziehung zwischen den verwendeten Formen im Ausschnitt des OED

Der hauptsächliche Unterschied zwischen diesem Ausschnitt und den vorherigen ist das Vorhandensein von expliziten zeitlichen Angaben. Diese ist für eine Einordnung der jeweiligen Form durchaus relevant, aber oftmals nicht direkt enthalten. Je nach Quellenmaterial kann eine solche Information allerdings zum Teil durchaus hergeleitet werden, beispielsweise über den Erhebungszeitraum eines Werks oder Sprachangaben mit zeitlicher Zuordnung wie „afrz.“ o.ä. Eine näherere Betrachtung findet sich in Kap. 2.2.4.

Zusammenfassend zeigt sich, dass die Art der Information, die sich aus strukturierten Abschnitten verschiedener Gattungen und Typen von lexikalischen Ressourcen ergibt, keine grundlegenden Unterschiede aufweist. Insbesondere sind die Unterschiede zwischen den Gattungen Wörterbuch und Sprachtatlas (z.B. Sprachatlas und Dialektwörterbuch) oftmals geringer, als solche innerhalb der Gattung (z.B. Dialektwörterbuch vs. etymologisches Wörterbuch). Digitale Repräsentationen können dies ausnutzen, da sie nicht auf eine eindeutige Darstellung festgelegt sind:

> Im Druckmedium ist das Buch zugleich Speicher- und Präsentationsmedium. In der digitalen Welt treten Aufbewahrung und Präsentation auseinander. (Prätor 2011, S. 172)

Durch eine konsequente Angleichung von Datenmaterial aus unterschiedlichen Quellen, können sich somit völlig neue Möglichkeiten der Publikation ergeben. Ein gutes Beispiel hierfür ist das Projekt VerbaAlpina, welches Material aus Sprachatlanten und Dialektwörterbüchern aus dem Alpenraum sammelt, vereinheitlicht und parallel als interaktive Karte und Wörterbuch veröffentlicht.

2.2 Definition eines abstrakten Datenmodells

Auf Basis der Analyse aus dem letzten Kapitel wird hier nun ein Datenmodell für die allgemeine Darstellung von lexikalischen Daten vorgeschlagen. Dieses ist *abstrakt* in dem Sinne, dass es weder ein konkretes Datenformat, noch die detaillierte Ausgestaltung der verschiedenen Kernelemente festlegt. Die Anforderungen an diese werden in den folgenden Kapiteln untersucht, aber deren genaue Formulierung nicht abschließend festgelegt. Hiermit soll also keine (informatische) Ontologie erstellt werden, sondern die Basis einer möglichst verwendungsunabhängigen Datenmodellierung gelegt werden, die es erlaubt die Daten aus verschiedenen Perspektiven anzusprechen (vgl. Kap. 12.1) und leichter wiederzuverwenden. Eine generische Ontologie als solche ist auch selten spezialisiert genug, um wirklich alle relevanten Informationen aus einer bestimmten Quelle vollständig abzubilden, sodass selbst in verhältnismäßigen einfachen Fällen wieder ein eigenes Modell bzw. eine Erweiterung nötig ist (vgl. z.B. Lüschow 2020). Dies zeigt, dass Ontologien zur Normierung zwar eine wichtige Rolle erfüllen, aber nicht als Basis eines Datenmodells fungieren sollten. Zielführender ist es meist ein spezifisches Modell zu erstellen, das die jeweiligen Projektdaten optimal abbildet und in einem zweiten Schritt daraus Exporte in eine oder mehrere Formate entsprechender Ontologien zu erstellen. Dieser zweite Schritt wird meistens mit einem Verlust von Information einhergehen, in vielen Anwendungsfällen, in denen Daten aus verschiedenen Quellen zusammengeführt werden, ist allerdings Vernetzung und Standardisierung wichtiger, als hundertprozentige Abbildung der Informationen jeder einzelnen.

Entsprechend der beiden Varianten von Daten aus dem vorherigen Kapitel, wird hier ein zweiteiliges Modell vorgestellt, welches einerseits den *Sprachbeleg* und andererseits Relationen zwischen verschiedenen sprachlichen Formen beinhaltet. Die verschiedenen Spalten des ersten Typs von Beispieltabellen werden zusammengefasst zu vier Dimensionen: Sprachliche Form, Bedeutung, Ort und Zeit. Dabei entsprechen die letzteren drei direkt einzelnen Spalten der Tabellen, während die textuelle Abbildung einer sprachlichen Form sowie grammatikalische Angaben, die Art der Transkription und die Sprache bzw. der Dialekt zur Beschreibung der Form zusammengefasst werden (vgl. auch Kap. 2.2.1). Der *Sprachbeleg* kontextualisiert somit eine einzelne sprachliche Form, verknüpft sie also mit den drei anderen Dimensionen: Bedeutung, Lokalisierung und zeitlicher Einteilung. Grundsätzlich wird hier die Verwendung einer sprachlichen Form abgebildet. Diese wird auch häufig mit dem englischen Begriff *usage* bezeichnet. Der Ausdruck *Sprachbeleg* baut eher auf dem Vorgang der Erhebung von Sprachdaten auf und beschreibt dabei ein einzelnes Datum, das von einem bestimmten Informanten aufgenommen wurde. Ein *Sprachbeleg* kann somit als „Rohdatum“ aufgefasst werden, wie es z.B. in einem entsprechenden Fragebogen vorkommt, bevor daraus dann beispielsweise ein Wörterbuch erstellt wurde.

Die Relationen geben eine Beziehung zwischen zwei sprachlichen Form wieder (nicht

zwischen zwei *Sprachbelegen*, vgl. Kap. 2.2.5). Während die *Sprachbelege* eher konkreten Daten sind, die der tatsächlichen Verwendung eines Wortes entsprechen, sind die Relationen tendenziell eher wissenschaftlicher und/oder normierender Natur, sodass sie einen größeren Interpretationsspielraum bieten. Im Kontext der Lexikographie entstehen sie in weiten Teilen aus der Verarbeitung des Belegmaterials, wenn dieses kategorisiert und analysiert wird.

Eine entscheidende Zusatzinformation für beide Bestandteile des Modells ist die Quellenangabe. Dieses Metadatum beschreibt beispielsweise im Kontext eines Wörterbuchs den genauen Artikel, aus dem ein Datum stammt. Entscheidend ist hier nicht nur die Nachvollziehbarkeit der Herkunft selbst, sondern auch eine gewisse „Relativierung“ der Information, die nötig ist, um mit widersprüchlichen Angaben umzugehen. Somit wird konzeptuell eine absolute Aussage der Form „Form f1 stammt von Form f2“ zu „Quelle q gibt an, dass Form f1 von Form f2 stammt“. Widersprüche treten grundsätzlich eher im Abgleich mit anderen Werken oder Datenbanken auf, sind aber auch innerhalb einer Quelle möglich. So werden im REW beispielsweise als Herkunft der galicischen Form *acadar* in der Bedeutung „auffangen“ in den Einträgen 62 und 63 zwei (homonyme) lateinische Etyma zugewiesen, was an dieser Stelle nur als Fehler aufgefasst werden kann. Wenn allerdings jede etymologische Relation einem (oder mehreren) Wörterbuchartikeln zugeordnet ist, aus dem diese stammt, kann dieser Fall trotzdem konsistent abgebildet werden[4]. Die folgende Abbildung illustriert das vollständige Modell:

[4] Schwieriger ist es allerdings diesen Fall von solchen abzugrenzen, in denen tatsächlich mehrere Etyma vorhanden sind, wie sie bei Kompositionen oder auch Kontaminationen auftreten. Zumindest im REW werden diese allerdings im Normalfall klar markiert (in diesen Beispielen über die direkte Angabe des zweiten Etymons im Artikel des ersten bzw. die Führung einer eigenen Liste mit zusammengesetzten Formen).

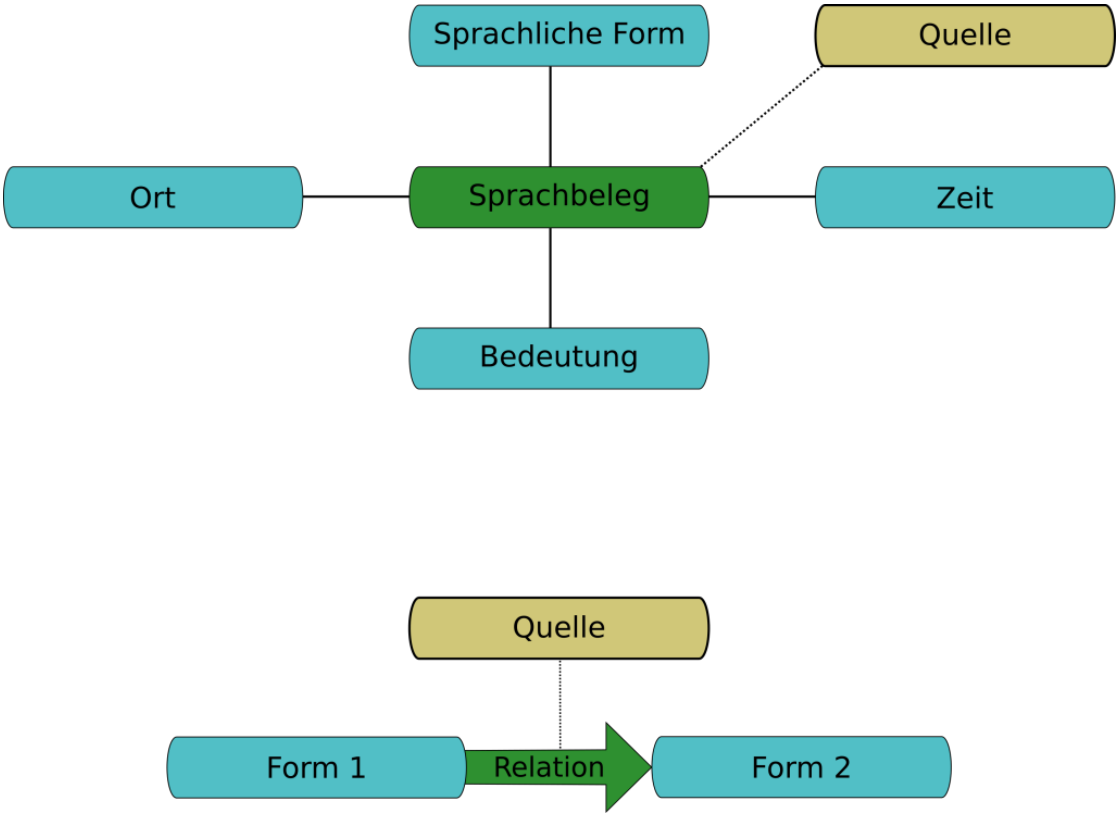


Abbildung 2.3: Illustration des abstrakten Modells für lexikalische Daten

Sowohl *Sprachbelege* als auch Relationen können zusätzlich genauer spezifiziert werden, so kann die Verwendung einer Form beispielsweise als umgangssprachlich markiert werden oder eine Relation als unsicher. Die folgenden Unterkapitel beschreiben die einzelnen Bestandteile des Modells im Detail und gehen auch insbesondere darauf ein, wie diese konsequent strukturiert und normiert werden können.

2.2.1 Sprachliche Form

Die sprachliche Form nimmt eine Sonderstellung unter den vier Dimensionen ein, da sie die einzige darstellt, die nicht optional ist. Ein minimaler *Sprachbeleg* kann somit rein aus einer Form ohne weitere Angaben bestehen[5]. Aus linguistischer Sicht hängt es vom Anwendungsfall ab, wie sinnvoll ein solcher Minimalbeleg ist, gerade im Bereich der Etymologie kann beispielsweise zumindest eine grundsätzliche etymologische Relation definiert werden auch, wenn eine der sprachlichen Formen (oder beide) nicht genauer kontextualisiert werden. Aus technischer Sicht ist ein Minimalbeleg auch in Fällen notwendig, in denen keine explizite Bedeutung vorhanden ist und sie auch nicht

[5]Für die technische Verarbeitung kann es u. U. trotzdem sinnvoll sein den Beleg zumindest formell zumindest mit einer „Pseudo-Bedeutung“ zu verknüpfen, vgl. Kap. 7.2.1

erschlossen werden kann (vgl. Kap. 7.2.1).

Die Basisinformation einer sprachlichen Form besteht aus einer Sprach- bzw. Dialektzuordnung und der eigentlichen textuellen Repräsentation[6]. Letzteres schließt grundsätzlich die Angabe ein, mit welchem Transkriptionssystem diese kodiert wurde. Oftmals ist eine solche Angabe allerdings intern nicht nötig, weil alle Formen in gleicher Art kodiert sind und spielt nur eine Rolle für die Ausgestaltung von Exportformaten. Im REW hängt die Transkription rein von der zugehörigen Sprachzuweisung ab (vgl. REW, S. XI–XII), die zum Teil mit einem Transkriptionssystem bzw. Alphabet verknüpft werden (vgl. hierzu auch Kap. 8.2.2). Die textuelle Repräsentation einer sprachlichen Form muss nicht aus einem einzelnen Token bestehen, auch längere Mehrwortlexien sind möglich.

Im Fall von multilingualen Ressourcen ist meist zu einer Form eine explizite Sprachzuordnung vorhanden, die im Normalfall aus einem quellenspezifischen Vokabular stammt. Vor allem bei der Angabe kleinräumiger Dialekte ist dies allerdings oftmals eher eine geographischen Angabe als eine Sprache oder ein Dialekt im eigentlichen Sinne. Auch eine tatsächliche Sprachangabe kann um zusätzliche geographische oder zeitliche Angaben ergänzt werden. Diese kann also im Extremfall Informationen aus drei der vier Dimension kodieren:

Sprachangabe	**Sprache**	**Ort**	**Zeit**
frz.	französisch	—	—
altwestfrz.	französisch	Westfrankreich	altfranzö-sisch
h.-alp. = MA. des Départements Hautes-Alpes	französisch	Département Hautes-Alpes	—

Tabelle 2.9: Kodierung von Information über die Angabe einer Sprache oder eines Dialekts

Der zeitliche Aspekt im zweiten Beispiel kommt hier über den Verwendungszeitraum einer gewissen Sprachstufe zustande, der mit einer gewissen Genauigkeit quantifiziert werden kann (vgl. auch Kap. 2.2.4). Die Sprachzuordnung im letzten Beispiel stammt aus einer Hierarchisierung der verschiedenen Sprach- und Dialektvorkommen im REW. Diese ist als solche nicht in der Quelle enthalten und wurde auf Basis der jeweiligen Abkürzungen erstellt, was auch für bestimmte Aspekte der Datenverarbeitung notwendig ist (vgl. Kap. 7.2.2). Im Einzelfall kann eine solche durchaus Spielraum für Interpretation lassen.

Anmerkung zum Begriff *Sprache* in der weiteren Arbeit: Für die automatisierte Verarbeitung ist die Abgrenzung von Sprachen und Dialekten meistens nicht notwendig (und wäre auch gerade im Fall von linguistisch umstrittenen Fällen nicht unbedingt sinnvoll). Aus Vereinfachungsgründen wird deshalb gerade im Bezug

[6] Zwei Formen mit gleicher Sprachzuordnung und textueller Repräsentation müssen allerdings nicht immer identisch sein, vgl. hierzu Kapitel 9

auf die technische Modellierung meist nur der Begriff *Sprache* verwendet. Falls der Unterschied relevant ist, wird explizit darauf hingewiesen. Auf die Verwendung eines Ausdrucks, der beides umfasst wie *Langoid* (vgl. Glottopedia Languoid, Good und Hendryx-Parker 2006) wird hier aufgrund des Fehlens einer allgemein verbreiteten Formulierung verzichtet.

Wie bereits aus den einführenden Beispielen ersichtlich, werden in diesem Modell Lemmata und sonstige sprachliche Formen strukturell nicht voneinander unterschieden. Jedes Lemma wird durch eine sprachliche Form repräsentiert, welche wiederum im Kontext einer anderen Quelle (oder auch im gleichen Werk) als „reguläre“ Form vorkommen kann. Die Verwendung als Lemma ist keine lexikalische Information im Sinn der hier verwendeten Definiton, sondern ein Spezifikum der jeweiligen Quelle und kann zusätzlich in deren Repräsentation in der Datenbank abgebildet werden (vgl. Kap. 6.5). In den meisten Fälle lassen sich allerdings aus der Verwendung eine Form als Lemma je nach dem Typ einer linguistischen Ressource bestimmte Relationen (vgl. Kap. 2.2.5) oder auch grammatikalische Informationen (je nach Sprache sind Verben traditionell im Infinitiv etc.) herleiten.

Zusätzlich kann die Form durch zusätzliche grammatikalische Angaben oder auch Meta-Informationen (beispielsweise ob die Form rekonstruiert ist) beschrieben werden. In den meisten Fällen ist allerdings die entsprechende Grundform angegeben. Für die grammatikalischen Angaben kann die Verwendung von Werten aus einem geeigneten kontrollierten Vokabular[7] (oder eine entsprechende Verknüpfung) sinnvoll sein. Aufgrund der sehr spärlichen grammatikalischen Angaben im REW wurde das in diesem Fall nicht umgesetzt.

Die Normierung von sprachlichen Formen bzw. die Vernetzung solcher aus verschiedenen Quellen untereinander ist ein Thema für sich, das an dieser Stelle nur kurz angerissen werden kann. Für eine quellenübergreifende Vernetzung ist grundsätzlich eine Verknüpfung entsprechender Nennformen untereinander zielführender als beispielsweise auf Ebene einzelner phonetischer Varianten. Letzteres würde zu einer sehr großen Anzahl von Relationen führen, die zum größten Teil redundant sind. In einfachen Fällen ist diese Vernetzung trivial, wenn beispielsweise auf Ebene der großen Standardsprachen einheitliche Verschriftlichungen existieren und somit die Nennformen identisch sind. Dies muss allerdings nicht immer so sein bzw. sind je nach Quelle keine solchen Nennformen vorhanden. In diesem Fall ist eine zusätzliche Typisierung (vgl. z.B. Krefeld und Lücke 2018) der Formen vor der Verknüpfung nötig. Diese kann entweder explizit über die Definition eigener Typen vorgenommen werden oder über die Verknüpfung mit einer entsprechenden Datenbank für lexikalische Informationen. Eine Möglichkeit kann hier *Wikidata* (siehe hierzu auch Kap. 10.2) sein, welches Lexeme definiert und diese über eindeutige IDs ansprechbar macht (vgl. z.B. Krefeld und Zacherl 2022). Die Anzahl der Lexeme ist gerade in den romanischen Sprachen zum aktuellen Zeitpunkt allerdings eher gering. Die

[7] Siehe beispielsweise https://lexinfo.net/.

verschiedenen *Wordnets* (vgl. Miller 1995) eignen sich in diesem Kontext eher nicht, da sie auf Gruppen von Synonymen aufgebaut sind und die einzelnen Formen nicht als solche referenzierbar sind.

2.2.2 Bedeutung

Bedeutungen werden (aus Mangel an Alternativen) durch einzelsprachliche textuelle Beschreibungen oder Übersetzungen angegeben. Der Abgleich mit anderen Quellen ist dadurch in vielen Fällen erschwert, da unterschiedliche Beschreibungssprachen (oder auch nur unterschiedliche Formulierungen) verwendet werden. Gerade hier ist also eine Normierung besonders entscheidend. Diese findet meist über die Verknüpfung mit einer entsprechenden Wissensdatenbank statt, wobei diese im Bezug auf Automatisierung aber auch konzeptuell durchaus herausfordernd sein kann. Kapitel 10 beschäftigt sich intensiv mit dieser Thematik, sodass sie an dieser Stelle nicht weiter ausgeführt wird.

2.2.3 Lokalisierung

Die Wichtigkeit einer geographischen Angabe ist in den verschiedenen Typen von linguistischen Ressourcen sehr unterschiedlich ausgeprägt. Je kleinräumiger und dialektaler der Umfang eines bestimmten Werks angelegt ist, desto entscheidender ist tendenziell eine genaue Lokalisierung von sprachlichen Formen. Aber auch sehr weiträumige Wörterbücher[8], die wie das REW unterschiedliche Daten zusammenführen, können zum Teil individuelle eng lokalisierte Belege angeben. Dabei können meist die geographischen Informationen aus der Angabe der Sprache der Form erschlossen werden (vgl. Kap. 2.2.1). Diese Beziehung ist allerdings nicht immer vorhanden oder variiert in ihrer Konfidenz. So kann beispielsweise aus der Sprachangabe „nordfranzösisch“ ein eindeutiges (wenn auch nicht unbedingt exakt eingrenzbares) Verbreitungsgebiet geschlossen werden kann, während „französisch“ allein nicht unbedingt mit dem französischen Staatsgebiet gleichzusetzen ist (vgl. auch Kap. 11.2). Ob diese Verknüpfung trotzdem Sinn ergibt, kann nicht unbedingt pauschal beantwortet werden. Gerade für Visualisierungzwecke kann die durchaus hilfreich sein (s. Kap. 12.3), ihre Integration in Daten für den Export ist zumindest dokumentationsbedürftig.

Die Beziehung zwischen sprachlicher Form und Bedeutung stellt oftmals die (alleinige) Grundlage eines Modells für lexikalische Daten dar (siehe hierzu das im folgenden besprochene O*ntoLex Lemon* Modell in Kap. 2.3.2). Diese Sicht mag für Anwendungen des *Natural Language Processing* bis zu einem gewissen Maß Sinn ergeben, aber auch hier kann eine Einbindung von dialektalen Daten (die immer bis zu einem gewissen Maße ortsbezogen sind) hilfreich sein oder neue Möglichkeiten eröffnen (vgl. z.B.

[8] Für Sprachatlanten ist eine geographische Zuordnung formbedingt immer vorhanden.

Scherrer und Rambow 2010). Für die linguistische Forschung und vor allem Dialektologie sind diese unverzichtbar und sollten deshalb eine prominentere Rolle in der Datenmodellierung einnehmen, als dies bislang oft der Fall ist.

Eine Normierung geographischer Angaben kann einerseits die Abbildung der räumlichen Dimension auf konkrete geographische Polygondaten sein, die den jeweiligen Raum definieren (siehe auch Kap. 12.3), aber auch die Verknüpfung mit dedizierten Ortsdatenbanken wie *Geonames*[9]. Zum Teil kann ersteres aus letzterem erzeugt werden (vgl. auch Kap. 11.2).

2.2.4 Zeitliche Dimension

Zeitliche Angaben stellen die vielleicht schwierigste Form der Spezifikation einer sprachlichen Form dar. Zum einen ist der genaue Verwendungszeitraum einer Form nur sehr eingeschränkt bestimmbar, andererseits ist auch die Frage komplexer, was genau mit einer zeitlichen Einordnung gemeint ist. Je nach Quelle können dabei drei unterschiedliche Ausprägungen unterschieden werden:

- **Gebrauchszeitraum:** Entspricht den Angaben aus dem OED Online und gibt an, wann diese Form tatsächlich verwendet wurde bzw. seit wann sie verwendet wird.
- **Erhebungszeitpunkt bzw. -zeitraum*:*** Gibt an, wann die Erhebung des *Sprachbelegs* beispielsweise für einen Sprachatlas oder ein Wörterbuch stattgefunden hat. Je nach Informationslage kann diese Angabe tagesaktuell sein. Dies ist gerade bei neueren Online-Erhebungen der Fall (vgl. z.B. Möller und Elspaß 2014, Krefeld und Lücke 2021).
- **Publikationszeitpunkt**: Der Zeitpunkt, an dem das Werk publiziert wurde, das den *Sprachbeleg* enthält.

Aus sprachwissenschaftlicher Sicht scheint zunächst die erste Variante den bestmöglichen Fall darzustellen, man muss allerdings beachten, dass es sich dabei grundsätzlich bereits um aggregierte und/oder wissenschaftlich interpretierte Information handelt. Eine solche Angabe beruht also streng genommen immer auf mehreren zugrundeliegenden *Sprachbelegen*. Dies widerspricht somit dem Anspruch des Datenmodells Rohdaten darzustellen. Besser wäre es grundsätzlich die eigentlichen Einzelbelege darzustellen, was je nach Quellenlage oftmals allerdings nicht realistisch ist. Die Nutzung eines Verwendungszeitraums hat also durchaus ihre Berechtigung, aus datentheoretischer Sicht ist die Verwendung der einzelnen Erhebungszeitpunkte passender[10] und liefert auch einen höheren Informationsgehalt.

[9] https://www.geonames.org
[10] Dies entspricht der Intuition eines Erhebungsdatenpunkts, die hinter dem Modell steht.

Grundsätzlich resultiert aus einer Erhebung die Information, dass die (durch die restlichen Dimensionen beschriebene) Verwendung aktuell ist. Wenn die Form (bzw. die Verwendung zusammen mit einer gewissen Bedeutung) zum Zeitpunkt der Erhebung (oder auch der Publikation) bereits als veraltet beschrieben wurde, stellt dies gewissermaßen einen Negativbeleg dar, der angibt, dass die Verwendung zu diesem Zeitpunkt an diesem Ort nicht mehr gegeben war. Man kann diesen also (zumindest lokal) als eine mögliche Obergrenze für den Verwendungszeitraum interpretieren. Die Verwendung des Publikationszeitpunkts an sich hat keine Vorteile und kann nur ein Notbehelf sein, wenn keine anderen Informationen zur Verfügung stehen.

Die meisten Quellen, die nicht explizit einen sprachhistorischen Anspruch haben, enthalten wenig bis gar keine zeitlichen Angaben. Im Fall des REW treten explizite zeitliche Angaben nur sehr vereinzelt auf. So sind zum Teil bei Lemmata ein Zeitraum (z.B. „Karolingerzeit“ (REW, S. 51)) oder ein Jahrhundert gegeben. Bei den sonstigen sprachlichen Formen sind an manchen Stellen Formulierungen wie „arag. *meseguero* früher „Feldhüter“, heute „Weinbergwächter“ García de Diego 401“ (REW, S. 5543) vorhanden. Eine genaue Quantifizierung letzterer Art von Angaben ist schwierig, streng genommen könnte man in diesem Fall schließen, dass die ältere Bedeutung maximal bis zum Veröffentlichungszeitpunkt des Wörterbuchs bestanden hat und die neuere Bedeutung spätestens ab diesem besteht. Man hätte somit im ersten Fall wiederum einen „Negativbeleg“. Eine bessere Eingrenzung des Zeitraums ist (ohne die Miteinbeziehung der Sekundärquelle) kaum möglich. Der Hauptteil der zeitlichen Einordnung kommt allerdings über Sprachangaben wie „altfranzösisch“ oder „mittelhochdeutsch“, die man entsprechend eines wissenschaftlichen Konsenses[11] verhältnismäßig sicher einem bestimmten Zeitintervall zuordnen kann[12]. Jede Angabe einer (etablierten) Sprachstufe kann somit auch als zeitliche Information gewertet werden (vgl. auch Khan 2020). Jeder *Sprachbeleg*, der keinem der genannten Muster entspricht kann grundsätzlich als aktuell gewertet und damit dem Zeitpunkt der Publikation zugeordnet werden.

Eine Normierung der zeitlichen Information ist durch die Verwendung von Jahreszahlen bzw. Intervallen bereits bis zu einem gewissen Maß gegeben. Gerade bei Namen bestimmter Zeitperioden, deren Zuordnung zu konkreten Jahreszahlen schwierig ist, kann analog zu den vorherigen Kapiteln die Zuordnung zu einer dedizierten „Zeit-Datenbank“ sinnvoll sein. Ein Beispiel hierfür ist das Projekt *PeriodO* (Rabinowitz u. a. 2016), das allerdings gerade für die in diesem Kontext wichtigen Sprachschichten keine Einträge enthält.

[11] Grundsätzlich müsste man den Konsens zum Zeitpunkt der Publikation betrachten, falls sich seitdem relevante Änderungen in der Definition ergeben haben.

[12] Trotzdem sollte die Zuordnung durch eine Quellenangabe markiert werden und somit nachvollziehbar sein.

2.2.5 Relationen zwischen sprachlichen Formen

Im Gegensatz zu den bisher behandelten Informationen, ergeben sich Relationen meist aus der Anordnung und Darstellung der sprachlichen Formen in der Quelle und werden selten[13] explizit angegeben. Alle Relationen in den Beispielen beziehen sich dabei auf sprachliche Formen, es wäre aber auch grundsätzlich möglich diese zum Teil auf Belegebene zu definieren. Dies ist wenig sinnvoll für die Zuordnung von Formen zueinander auf morphologischer Ebene, wäre aber für etymologische Relationen denkbar. In diesem Fall würde man die (sehr viel spezifischere) Information ablegen, aus welcher Verwendung einer sprachlichen Form sich welche Verwendung einer anderen sprachlichen Form ergeben hat. Insbesondere wäre die Bedeutung Teil der etymologischen Relation. Im REW findet sich eine sehr geringe Anzahl von Fällen, in denen eine Entlehnung nicht auf die sprachliche Form, sondern auf die Bedeutung bezogen ist. In diesem Fall hat man gewissermaßen die Zusatzangabe, dass aus genau dieser Bedeutung des Etymons eine romanische Form entstanden ist. Dies kann im bestehenden Modell so nicht abgebildet werden. Trotzdem erscheint die Definition der etymologischen Relationen aus Belegbasis wenig vielversprechend, da in den meisten Fällen keine genaue Information vorhanden ist und man im Fall verschiedener Bedeutungen des Etymon so mehrere alternative Relationen anlegen müsste. Eine bessere Lösung wäre wohl die Relation selbst mit Angabe einer Bedeutung als zusätzlichen Qualifikator zu versehen, um diese Spezialfälle ebenfalls darstellen zu können.

Es sind auch Relationen zwischen Elementen der anderen Dimensionen wie Orten oder Bedeutungen möglich. Diese können allerdings im Normalfall nicht aus einer linguistischen Quelle erschlossen werden. Durch die Vernetzung mit externen Wissensdatenbanken können dort vorhandene Relationen (z.B. Ortshierarchien von *Geonames* oder die *Statements* von *Wikidata)* auch in Verbindung mit den lexikalischen Kerndaten genutzt werden.

2.3 Umsetzung mit bestehenden Standards zur Kodierung lexikalischer Daten

In den letzten Jahren haben sich in den Geisteswissenschaften bzw. den *Digital Humanities* vor allem zwei Formate zur Darstellung von sprachlichen Daten etabliert, die ursprünglich mit sehr unterschiedlichen Zielsetzungen entwickelt wurden, deren Verwendung sich aber aber zunehmend überschneidet.

Zum einen gibt es das von der *Text Encoding Initiative* entwickelte gleichnamige Format *TEI* (vgl. Consortium 2021 für die aktuellste Fassung), das auf XML basiert

[13] Eine Ausnahme ist beispielsweise die Angabe der Entlehnungen im REW, vgl. Kap. 6.5.1

und als „*de facto* standard for electronic text encoding in the humanities“ (Cantara 2005, S. 36) gilt. Wie der Name schon sagt, eignet es sich primär zur Darstellung und Annotation von elektronischem Text, also vor allem um bestehende Publikationen unterschiedlicher Art für eine digitale Nutzung aufzubereiten. Für die Verwendung mit lexikalischen Daten bietet *TEI* das *Dictionaries*-Modul an, das entsprechende Entitäten für diesen Kontext definiert. Darauf aufbauend wurde weiterhin die *TEI Lex-0* Spezifikation[14] entwickelt, die die sehr variablen und wenig restriktiven *TEI*-Richtlinien deutlich stärker einschränkt und die Zielsetzung hat damit die Interoperabilität zwischen verschiedenen Ressourcen zu erhöhen (Romary und Tasovac 2018).

Der zweite weit verbreitete Standard ist das sogenannte *OntoLex Lemon* Modell. Es wurde erstmals erstmals in J. McCrae, Spohr und Cimiano 2011 als *lexicon model for ontologies* (l*emon)* vorgeschlagen und seitdem überarbeitet und unter dem aktuellen Namen veröffentlicht (Cimiano, John P. McCrae und Buitelaar 2016). Es basiert auf dem *Resource Description Framework* (*RDF*) und verfolgt die grundsätzliche Idee lexikalische bzw. linguistische Daten an bestehende Strukturen des *Semantic Webs* anzubinden:

> Our goal is thus to provide a formalisms that ‘connects these worlds’, i.e. the world of lexical resources and the world of ontologies and semantic data as available on the Semantic Web.(J. McCrae, Aguado-de-Cea u. a. 2012, S. 702)

Im Kontext des *Semantic Webs* wird das Modell als „primary mechanism for the representation of lexical data“ (John P McCrae u. a. 2017, S. 1) und „de-facto standard for this purpose“ (Abromeit u. a. 2016, S. 14) bezeichnet. Obwohl die ursprüngliche Absicht dieses Modells hauptsächlich die bessere Verfügbarmachung solcher Daten für Anwendungen des *Natural Language Processings* (*NLP*) war (vgl. J. McCrae, Spohr und Cimiano 2011), wurde es auch zunehmend relevant für originär geisteswissenschaftliche Fragestellungen (vgl z.B. Chavula und Keet 2014, Abromeit u. a. 2016, Declerck 2017), was sich auch in (bereits vorhandenen und geplanten) Anpassungen und Erweiterungen widerspiegelt (vgl. John P McCrae u. a. 2017). Seit Ende 2019 gibt es auch ein *Lexicography Module* (vgl. Bosque-Gil, Gracia und Montiel-Ponsoda 2017, Bosque-Gil, Gracia, J. McCrae u. a. o. D.), um die Abbildung von bestehenden lexikographischen Ressourcen und die Erstellung von neuen unter Verwendung von *Linked Data* zu ermöglichen (Bosque-Gil, Gracia, J. McCrae u. a. o. D., §1.2).

Beide Standards haben einen Einfluss, der weit über deren ursprüngliche Konzeption hinausgeht, sodass inzwischen trotz der sehr unterschiedlichen Ausgangslage ähnliche und zum Teil überlappende Anwendungsfälle vorkommen. Während das *TEI*-Format

[14] https://dariah-eric.github.io/lexicalresources/pages/TEILex0/TEILex0.html

inzwischen auch zur Repräsentation von linguistischen Daten verwendet wird, die nicht als textbasierte Publikation erschienen sind (vgl. z.B. Bowers und Stöckle 2018) bzw. solche bei Weiterentwicklungen durchaus berücksichtigt werden (vgl. Romary und Tasovac 2018), wird wiederum *OntoLex Lemon* auch verstärkt zur Darstellung von traditionellen Wörterbüchern verwendet[15]. Aus praktischer Sicht ist ein Unterschied allerdings, dass *TEI* im Kontexts eines Wörterbuchs durchaus zur Strukturierung der Primärdaten verwendet wird, während in *OntoLex Lemon* repräsentierte Daten im Normalfall eher sekundär aus bereits in einem anderen Format vorhandenen Daten zusätzlich erzeugt werden. Trotzdem kann es sinnvoll sein die grundlegenden Aspekte der Modellierung auch für die Primärdaten zu übernehmen, um sowohl die Nachnutzung als auch eine Konversion zu erleichtern. Im Kontext der im vorigen Kapitel aufgestellten Anforderungen weisen allerdings beide Formen der Datenmodellierung Nachteile auf, die im folgenden betrachtet werden.

2.3.1 *TEI* und lexikalische Daten

Während *TEI* grundsätzlich die natürliche Wahl für die Darstellung der Wörterbuchtexte selbst ist, eignet es sich in verschiedener Hinsicht weniger gut für die Darstellung von Daten nach dem definierten Modell. Es werden alle benötigten Entitäten abgebildet, die Problem sind allerdings grundsätzlicherer Natur. Somit beschäftigt sich dieses Kapitel weniger damit, wie gut das *Dictionary*-Modul von *TEI* sich in der Praxis für die Abbildung eines Wörterbuchs eignet oder wie die Umwandlung automatisiert durchgeführt werden kann, sondern thematisiert grundsätzlichere Probleme bei der Repräsentation von lexikalischen Daten als Annotationen in einem text-basierten Format, die sich auch auf andere, ähnlich konzipierte Standards übertragen ließen.

Der Artikel eines Wörterbuchs kann also als semi-strukturierter Text aufgefasst werden (vgl. dazu Kap. 5.2), in dem die natürlichsprachlichen Passagen als „Notbehelf“ genutzt werden, um zusätzliche Information anzubringen, die allein durch Struktur und Aufbau nicht transportiert werden kann. Der Fokus liegt hier zwar auf den strukturierten Daten, es sollen aber auch durchaus die natürlichsprachigen Abschnitte in geeigneter Weise dargestellt werden (vgl. Kap. 6.5.3). Somit sind zwei grundsätzliche Herangehensweisen denkbar, die beide ihre Vor- und Nachteile haben. Wenn die strukturierten Daten im Vordergrund stehen und entsprechend in einer Datenbank abgelegt werden, ist die Behandlung der restlichen Passagen aufwendiger und weniger natürlich. Umgekehrt ist bei einem auf dem Ursprungstext aufbauenden Format wie *TEI* die Darstellung der strukturierten Daten in einer Art und Weise, in der sie möglichst vielseitig genutzt werden können, schwieriger. Da in diesem Fall die Kerndaten strukturierte Natur sind und die restlichen Elemente eher nachgelagert betrachtet werden, liegt der erste Ansatz grundsätzlich näher. Trotzdem soll hier der

[15] Eine ausführliche Liste solcher Projekte findet sich in (John P McCrae u. a. 2017, S. 5)

umgekehrte Weg genauer betrachtet werden. Dieser weist im Bezug auf die lexikalischen Daten zwei hauptsächliche Probleme auf.

Einerseits ist ein durchaus relevanter Anteil von Information ist gar nicht explizit im Text enthalten und kann nur aus dessen Aufbau hergeleitet werden. Das ist insbesondere der Fall bei Relationen zwischen verschiedenen Formen, aber auch im Fall verschiedenster Auslassungen (vgl. Kap. 7.2.1). Diese Art von Information müsste also entweder zusätzlich in den annotierten Wörterbuchtext eingefügt werden oder bei Verwendung jedes Mal neu berechnet werden. Letzteres ist aufgrund der komplexen Problematik (vgl. Kapitel 7) und des relativ hohen Rechenaufwands wenig realistisch. Somit müssen die Möglichkeiten näher betrachtet werden Anpassungen am Originaltext vorzunehmen.

Das *Dictionary*-Modul von *TEI* unterscheidet drei verschiedene Sichten auf den Wörterbuchtext, angefangen mit dem *typographic view*, also einer exakten Wiedergabe des gesetzten Textes bis hin zu Zeilen- und Seitenumbrüchen, über den *editorial view*, bei der der Text darstellungsunabhängig abgelegt wird, bis hin zum sogenannten *lexical view* (Consortium 2021, Kap. 9.5). Diese letztgenannte Sicht erlaubt gewisse Abweichungen im Vergleich zur Textbasis:

> This view includes the underlying information represented in a dictionary, without concern for its exact textual form(Consortium 2021, Kap. 9.5)

Als Beispiele für solche Abweichungen werden Normalisierung, Auflösung von Abkürzungen und Änderungen in der Reihenfolge genannt (Consortium 2021, Kap. 9.5.2). In einem gewissen Maß ist also eine Unterscheidung von Darstellung und zugrundeliegenden Daten durchaus vorgesehen. Die Auflösung von Abkürzung, das Einfügen nicht explizit vorhandener Bedeutungen und ähnliches könnte dabei durchaus grundsätzlich auch auf Basis von *TEI* durchgeführt werden, auch wenn die Darstellung von Relationen weiterhin schwierig wäre.

Der zweite Nachteil ist allerdings schwerwiegender und ergibt sich aus der Festlegung auf eine Art des Zugangs. Jedes Wörterbuch ist aus einer bestimmten Perspektive konzipiert. Der Autor muss sich also entscheiden, auf welche Weise primär auf die Inhalte zugegriffen werden kann. Zusätzliche Zugriffswege können zwar beispielsweise über Wortverzeichnisse ergänzt werden, diese sind allerdings aufwendiger zu nutzen, als der primäre Zugang über die Anordnung der Artikel und kosten zusätzlichen Platz, der in gedruckten Werken grundsätzlich knapp ist. Dies führt dazu, dass diese oftmals nicht vollständig sind:

> Die Wortverzeichnisse der anderen Sprachen sind möglichst vollständig, das deutsch-romanische bietet naturgemäß nur eine Auswahl, ist gegen die erste Ausgabe in den Stichwörtern kaum erweitert worden, erschöpft auch nicht

> den im Texte enthaltenen Stoff, da eine noch weitere Ausdehnung des Raumes ausgeschlossen war […](REW, S. 815)

In Digitalrepräsentationen ist diese Einschränkung auf eine Zugriffsperspektive allerdings weder notwendig noch sinnvoll (vgl. z.B. Prätor 2011), vielmehr sollte ein möglichst generalisiertes Datenmodell erstellt werden[16], das den Zugang aus verschiedenen Richtungen möglichst unkompliziert möglich macht (vgl. Kap. 12.1). Gerade auch im Fall von neu erhobenen lexikalischen Daten erscheint es wenig sinnvoll diese entsprechend traditioneller Wörterbuchformate zu strukturieren, wenn die zugrundeliegenden Daten sowohl einen semasiologischen, als auch einen onomasiologischen Zugang auf natürliche Weise erlauben. Ein Format, welches auf dem eigentlichen Wörterbuchtext basiert, kann diese Einschränkungen aber nie ablegen und ist deshalb keine optimale Basis für die Darstellung der strukturierten Information.

2.3.2 Ableich mit dem *OntoLex Lemon* Modell

Im Gegensatz zum Hauptnachteil, der im vorherigen Kapitel beschrieben wurde, ist der Gedanke einer zugangsneutralen Abbildung durchaus in die Konzeption des *lemon*-Modells eingeflossen. Es wird dementsprechend als „being descriptive but not prescriptive, which facilitates neutrality towards different lexicographic views“ (Bosque-Gil, Gracia, J. McCrae u. a. o. D., §1.1) beschrieben. Der Kern des Modells ist der *Lexical Entry*, der eine Einheit aus mindestens einer Form und einer Menge von Bedeutungen darstellt:

> A **lexical entry** represents a unit of analysis of the lexicon that consists of a set of forms that are grammatically related and a set of base meanings that are associated with all of these forms.(Cimiano, John P. McCrae und Buitelaar 2016, §3.1)

Zu beachten ist dabei, dass der Begriff *meaning* bzw. *Lexical sense* dabei nicht durch eine konkrete Bedeutung im klassischen Sinne repräsentiert wird, sondern eine reine Verknüpfung zu einem Konzept in einer entsprechenden Ontologie darstellt (vgl. John P McCrae u. a. 2017, S. 589). Der *Lexical Sense* wird dabei dem *Lexical Entry* untergeordnet, ist also immer auf einen bestimmten Eintrag bezogen und kann nicht von verschiedenen verwendet werden. Im Gegensatz zum klassischen *lemon* definiert *Ontolex Lemon* zusätzlich ein *Lexical Concept*, welches die Definition von allgemeinen Konzepten erlaubt, denen einzelne konkrete *Lexical sense* Elementen zugeordnet werden können. *Ontolex Lemon* selbst erlaubt allerdings weder die Repräsentation von geographischen noch von zeitlichen Informationen, auch wenn es zumindest für letztere Vorschläge zur Integration in das Modell gibt (s. Khan 2020), die allerdings etwas

[16] Vgl. hierzu auch die ähnlichen Überlegungen im Bezug auf den Unterschied zwischen Sprachatlas und Wörterbuch in Kap. 2.1

künstlich erscheinen, indem die Hauptklassen durch abgewandelte Varianten ersetzt werden, und nicht unbedingt für die Generalisierbarkeit des Modells an sich sprechen. Es ist allerdings über das Modul *vartrans* die Erstellung von diachronen oder diatopischen Relationen auf Ebene der *Lexical Senses* möglich (Cimiano, John P. McCrae und Buitelaar 2016, §6.1).

Ontolex Lemon enthält selbst keine grammatikalischen Begriffe wie Wortarten oder ähnliches, sondern erfordert die Verwendung eines zusätzliche Vokabulars aus einer passenden Ontologie, um diese abzubilden. In der Theorie ist diese nicht festgelegt, praktisch gesehen besteht allerdings eine sehr enge Verflechtung mit der *lexinfo*-Ontologie[17], die speziell für *lemon* entwickelt wurde (vgl. Cimiano, Buitelaar u. a. 2011) und auch in der Dokumentation von *Ontolex Lemon* für entsprechende Beispiele verwendet wird. Obwohl diese also streng genommen kein Teil des Modells ist, kann dies in der Praxis durchaus so aufgefasst werden. Unter Berücksichtigung von *Lexinfo* stehen weitere Möglichkeiten der Darstellung von temporaler und räumlicher Information zur Verfügung, nämlich die *Property dating*, mit der allerdings nur die Werte *old* und *modern* zugeordnet werden können, und (seit Version 3.0) die *usage-Property geographic,* mit der einem *Lexical Sense* ein (hauptsächliches) Verwendungsgebiet zugeordnet werden kann. In *lexinfo* sind auch verschiedene Relationen für beispielsweise etymologische Herleitungen auf Ebene der *Lexical Entries* möglich.

Insgesamt können die Bestandteile also folgendermaßen im *OntoLex Lemon* Modell (unter zusätzlicher Verwendung von *lexinfo*) abgebildet werden: Der *Sprachbeleg* entspricht gewissermaßen dem *Lexical Sense*, welcher Formen mit Konzepten verknüpft. Die geographische Zuordnung ist nicht direkt Teil des Modells, also weniger prominent, kann aber zusätzlich dem *Lexical Sense* zugeordnet werden. Eine zeitliche Einordnung der Verwendung ist auf dieser Basis allerdings nur sehr eingeschränkt möglich. Die Markierung als *veraltet* und *modern* entspricht zwar des öfteren Angaben wie sie auch in traditionellen Ressourcen vorkommen und können diese somit leicht abbilden, aus konzeptueller Sicht ist dies aber eher unsauber. Vor allem im Bereich des *Semantic Web*, in dem die Daten keinen Zeitschnitt wie bei einer traditionellen Quelle bilden, der eine solche Verwendung bis zu einem gewissen Maß rechtfertigt, ist diese Verwendung problematisch, da vor allem die Markierung *modern* auch Änderungen unterworfen sein kann oder regionale Unterschiede aufweisen kann. Eine konkrete Zeitzuordnung könnte allerdings über die Verwendung einer zusätzlichen Ontologie geschehen, die entsprechende *Properties* enthält.

Ein grundlegender Unterschied zwischen dieser Modellierung und den einzelnen Sprachbelegen ist allerdings die Bündelung verschiedener Belege über das Konstrukt des *Lexical Entry*, welcher die zentrale Einheit darstellt und genauen Regeln unterworfen ist:

[17] http://lexinfo.net/

> Two terms may be different lexical entries if they are distinct in part-of-speech, gender, inflected forms or etymology.(Bosque-Gil, Gracia, J. McCrae u. a. o. D., §3.3)

Während im vorgestellten Modell Belege die Datenbasis bilden, die in einem zusätzlichen Schritt typisiert werden können, stellt hier somit gewissermaßen die Typisierung (zumindest auf morphologischer Ebene) den Kern des Modells dar, ohne die überhaupt keine Daten abgebildet werden können. Über die Behandlung von phonetischen Varianten wird weiterhin keine explizite Aussage getroffen, nach der Regel im Zitat oben, würde solche allerdings als einzelne *Lexical Entries* dargestellt werden. Die Relationen, wie sie im abstrakten Modell vorgestellt wurden, können grundsätzlich abgebildet werden, wobei ein Teil der Beispiele bereits implizit über die Bündelung der Formen zu einem *Lexical Entry* abgedeckt sind. Etymologische Relationen sind allerdings auf Basis von Belegen definiert und nicht auf Basis von Formen bzw. *Lexical Entries* (vgl. hierzu die Diskussion in Kap. 2.2.5).

Somit sind beide Modelle bis zu einem gewissen Maß durchaus aufeinander abbildbar, es tritt allerdings ein relevanter Unterschied auf, der sich aus den verschiedenen Ansprüchen ergibt. *OntoLex Lemon* bildet lexikalische Information für die weitere Verwendung (z.B. im Kontext des *Natural Language Processings*) ab, es beschreibt somit wissenschaftlich aufbereitete Daten und somit gewissermaßen die Resultate linguistischer Forschung, während das hier beschriebene Modell auf Rohdaten basiert, auf deren Basis eine solche stattfinden kann. Dies zeigt sich zum Teil auch daran, dass die Abbildung von spezifisch dialektologischer Information kaum vorgesehen ist, da *OntoLex Lemon* einen sehr hochsprachlich und wenig variablen Zugang zu lexikalischen Daten impliziert, der für die Sprachforschung selbst nicht unbedingt günstig ist. Gerade zur Darstellung von grundlegenden Rohdaten ist dieses Modell somit weniger geeignet und dient eher der Veröffentlichung der Ergebnisse von linguistischen Forschungen.

3 Konzeption von Transformationsprozessen

Der Kern dieser Arbeit beschäftigt sich mit der Umwandlung von textuell kodierter Information aus einem Wörterbuch in strukturierte Daten. Dabei wird dies nicht als einmaliger Prozess aufgefasst, sondern im Kontext eines Redaktionssystems, das es ermöglicht die einzelnen Schritte zu modifizieren und zu wiederholen, um Fehler zu korrigieren und die entstehenden Resultatdaten zu verbessern. Dieses Kapitel bespricht auf abstrakter Basis grundlegende konzeptuelle Entscheidungen und begründet deren Zustandekommen.

Anmerkung zu den Begriffen: Der hier vorgestellte Prozess transformiert (textuelle) Eingangsdaten in (strukturierte) Ausgangsdaten, wobei beides zusammen in einer relationalen Datenbank abgelegt wird. Somit kann er nicht im Wortsinn als *Import* bezeichnet werden. In bestimmten Kontexten wird dennoch dieser Begriff verwendet, da er gerade im Vergleich mit anderen Strategien zu intuitiveren Formulierungen führt.

3.1 Grundlegende Design-Prinzipien

Ein wichtiges grundsätzliches Prinzip kann als iterativer Ansatz beschrieben werden. Dieser Begriff wird häufig in der Softwareentwicklung verwendet und kann folgendermaßen definiert werden:

> Bei iterativem Vorgehen ist ein Zurückspringen zu vorangegangenen Arbeitsschritten möglich, bspw. wird nach einer Probeimplementierung wieder bei der Definition oder dem Entwurf angesetzt.(Krcmar 2015, S. 231)

Im vorliegenden Fall wird dieses Konzept in beiden grundlegenden Phasen im Lebenszyklus einer Digitalprojekts angewandt, in der initiale Umsetzungsphase und der darauf folgenden Betriebsphase. Mit ersterer wird hier sowohl die inhaltliche als auch die technische Vollendung beschrieben, d.h. das vollständige Datenmaterial wurde importiert und der technische Zugang dazu wurde fertiggestellt[1]. Die zweite Phase

[1] Im Kontexts eines Softwareprojekts von „fertig“ zu sprechen ist grundsätzlich schwierig, da selbst, wenn keine Änderungen an der Funktionalität mehr vorgenommen werden, weiterhin ein gewisser Wartungsaufwand zur Fehlerbehebung bzw. Anpassung an neue technische Entwicklungen notwendig ist. Hier wird also nur der Zustand der Vollständigkeit der gewünschten Funktionen und der Möglichkeit von deren Verwendung damit bezeichnet.

bezeichnet hier den darauf folgenden Abschnitt, in dem das Projekt verwendet werden kann und keine neuen Daten importiert werden. In der Praxis können sich beide Phasen durchaus überschneiden bzw. die erste Phase selbst mehrfach auf Basis unterschiedlicher Eingangsdaten durchgeführt werden, im vorliegenden Fall ist aufgrund des begrenzten Datenmaterials allerdings eine klare Trennung möglich. In diesem Abschnitt wird der iterative Ansatz im Kontext der ersten Phase besprochen und bezieht sich auf die grundsätzliche Methodik der Datenverarbeitung. Die spezifischere Anwendung auf das technische System wird vor allem in Kap. 3.3 behandelt.

Die grundlegende Vorgehensweise ist dabei, dass die einzelnen Arbeitsschritte des Import- bzw. Transformationsprozesses nicht vollständig ausgeführt werden, bevor die nächste Phase begonnen wird, sondern auf Ausschnitten alle Prozessschritte ausgeführt werden, sodass Ergebnisse und Erfahrungen aus diesen wiederum in früheren Schritten für neue Eingabedaten genutzt werden können. So werden beispielsweise nicht alle Scans zu Beginn mit einem Texterkennungssystem verarbeitet, bevor die daraus entstandenen Daten weiter verarbeitet werden, sondern nur Teile, die manuell nachkorrigiert werden, sodass diese Ergebnisse für die Verbesserung neu eingelesener Seiten dienen können (vgl. Kap. 4.3). Auch werden in einem ersten Schritt die Verzeichnisse mit Abkürzungen zu Beginn des Werkes importiert, so dass diese sowohl in früheren Prozessschritten (vgl. Kap. 4.2, Kap. 4.3) als auch in späteren (vgl. z.B. Kap. 5.2.3) verwendet werden können. Dabei muss die Verabeitungsreihenfolge nicht immer der Anordnung im Quellenmaterial entsprechen. Zum Beispiel erwies es sich im REW sinnvoll vor der Verarbeitung der Bibliographie, die später gelisteten allgemeinen Abkürzungen zu erfassen, so dass diese Information bei der strukturellen Erfassung der Bibliographie-Einträge genutzt werden konnten, um Abkürzungen innerhalb der Bibliographie als solche zu erkennen[2].

Für den Hauptteil des Materials (in diesem Fall die eigentlichen Wörterbuchartikel) wird ein Anteil des gesamten Materials sehr intensiv verarbeitet und korrigiert. Damit kann folgendes erreicht werden:

1. Die Erkennung, welche Konventionen, Notationen etc. häufig sind und was selten vorkommt, also die Entscheidung zwischen Regel und Ausnahme, ist möglich. Dafür sollten die verarbeitenden Ausschnitte möglichst aus verschiedenen Bereichen der Quelle stammen und nicht nur den Beginn des Quellentexts abbilden (vgl. Kap. 5.2.3)

2. Die Qualität der maschinellen Texterkennung bzw. einer entsprechenden Nachverbesserung (Kap. 4.3) kann gesteigert werden, bevor alles verarbeitet wird.

[2]Dies ist in diesem Fall wichtiger als es auf den ersten Blick wirkt, da die Nicht-Erkennung einer Abkürzung häufig dazu führt, dass der meist dort enthaltende Punkt als Strukturelement behandelt wird, was zu falschen Ergebnissen führt. Diese „Doppelverwendung“ des Punkts zur Beendung von Abkürzungen und Sätzen kann häufig problematisch sein (vgl. auch Kap. 5.3.3).

3. Trotzdem auftretende Fehler können erkannt werden, sodass sie später auch im Rest der Daten korrigiert werden können (vgl. Kap. 8.2).

Innerhalb dieses Ausschnittes wird ebenfalls ein iteratives Vorgehen angewendet, sodass nur wenige Seiten gleichzeitig verarbeitet werden, um sehr augenscheinlich häufig auftretende Probleme im aktuellen Ablauf direkt beheben zu können. Ein grundsätzlicher Nachteil dieses Vorgehens ist allerdings, dass ein erstes vollständiges Ergebnis erst sehr spät produziert werden kann, was man durchaus in die Planung der Prozesse einbeziehen sollte.

Ein zweites Prinzip ist die Integration des Vorhandenseins von Fehlern in die Konzeption des Prozesses. Auch die manuelle Korrektur von solchen wird hier nicht als Ausschlusskriterium behandelt, wie es in älteren Publikationen zu diesem Thema zum Teil dargestellt wird:

> Da diese Fehler zwangsläufig zu mangelhaften Extraktionsresultaten führen und eine manuelle Korrektur unverhältnismäßig aufwendig ist, müssen diese Artikel bei der Wiederverwendung ausgeschlossen werden.(Heyn 1992, S. 188)

Vielmehr wird hier der Fehler als Bestandteil des Gesamtsystems betrachtet, wobei der Fokus darauf liegt, wie moderne Web-Technologie und eine passende Datenmodellierung genutzt werden können, um die nicht vermeidbaren Fehler in den automatisierten Prozessen möglichst leicht manuell oder teilautomatisiert zu beheben. Ein perfektes Ergebnis ist keine realistische Annahme, sodass stattdessen Methoden und Werkzeuge entwickelt werden, um Fehler dauerhaft einfach (d.h. unaufwendig und intuitiv) korrigieren zu können (vgl. Kapitel 8, Kap. 12.5).

Für den Kern-Transformationsprozess, also die Umwandlung von Text in entsprechende strukturierte Daten, wird eine strikte Zweiteilung angewandt. Im ersten Schritt wird die Struktur eines Wörterbuchs beschrieben und daraus eine abstrakte Repräsentation von dessen Inhalten generiert (s. Kapitel 5), im zweiten Schritt werden daraus „explizite“ Daten generiert (s. Kapitel 7). Für beide Schritte ist zuerst eine intensive intellektuelle Analyse des Quellenmaterials und eine Vertrautheit mit allgemeinen Konventionen der jeweiligen Disziplin (in diesem Fall der Lexikographie bzw. der Linguistik als Ganzes) nötig, um spezifische Konventionen zu verstehen und entsprechend technisch abbilden zu können.

Der Unterschied zwischen den beiden Phasen kann abstrakt am Beispiel der Bedeutungen im REW exemplifiziert werden. In den meisten Fällen ist im Wörterbuchtext für einzelne Formen keine explizite Bedeutung angegeben:

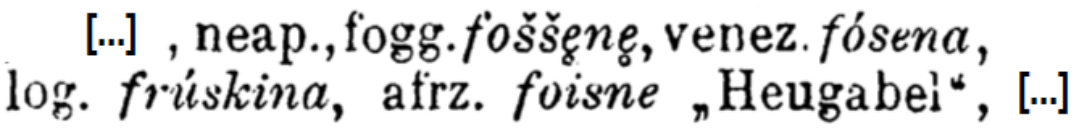

[...] , neap., fogg. *foššęnę*, venez. *fósena*, log. *frúskina*, afrz. *foisne* „Heugabel“, [...]

Abbildung 3.1: Ausschnitt aus REW, S. 3610

Im ersten Schritt wird dieser Zustand als solcher abgebildet, z.B.:

Sprachen	**Form**	**Bedeutung**
neap., fogg.	foššene	—
venez.	fósena	—
log.	frúskina	—
afrz.	foisne	Heugabel

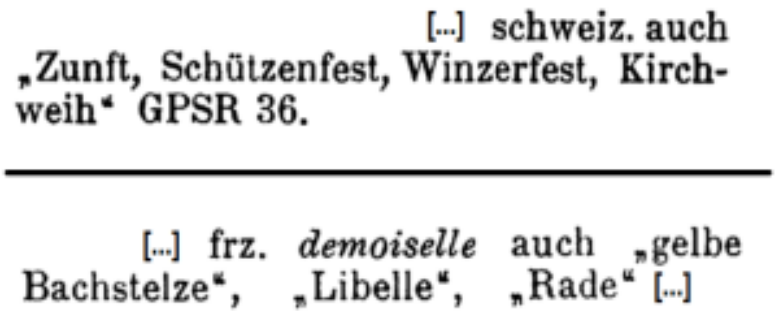
Tabelle 3.1: Abstrakte Darstellung der Daten in der ersten Phase des Transformationsprozesses

Im zweiten Schritt werden dann Lücken gefüllt, indem die fehlenden Bedeutungen je nach Position der Formen im Artikel erschlossen (vgl. hierzu Kap. 7.2.1) und abkürzende Schreibweisen aufgelöst werden:

Sprachen	**Form**	**Bedeutung**
neap.	foššene	Harpune
fogg.	foššene	Harpune
venez.	fósena	Harpune
log.	frúskina	Harpune
afrz.	foisne	Heugabel

Tabelle 3.2: Abstrakte Darstellung der Daten in der zweiten Phase des Transformationsprozesses

Beide Schritte sind im Allgemeinen durchaus komplex, so dass eine klare Trennung dabei hilft den Gesamtprozess überschaubarer zu gestalten. Im Zuge der zweite Phase werden zusätzlich diverse Inkonsistenzen wie unterschiedliche Abkürzungsvarianten oder verschiedene Arten der Notation (vgl. Abb. 3.2) angeglichen.


[...] schweiz. auch „Zunft, Schützenfest, Winzerfest, Kirchweih“ GPSR 36.

[...] frz. *demoiselle* auch „gelbe Bachstelze“, „Libelle“, „Rade“ [...]

Abbildung 3.2: Inkonsequente Angabe von mehreren Bedeutungen in REW, S. 9 (oben) und REW, S. 2737 (unten)

Weiterhin wird Wert auf einen möglichst hohe Formalisierung der verwendeten Methoden gelegt. Der Grundgedanke dabei ist den Programmcode selbst möglichst generalisiert zu gestalten, um dessen Wiederverwendbarkeit zu erhöhen, und spezifischere Anteile entsprechend eines formalen Modells als Daten aufzufassen (vgl. Kapitel 5) oder zumindest generische wiederverwendbare Grundfunktionalitäten zu entwickeln, die zumindest auf Programmcodeebene eine gewisse Formalisierung bewirken (vgl. Kapitel 7).

Zusätzlich wird eine strenge Quellentreue im Bezug auf die Eingangsdaten angewendet. Diese entsprechen jeweils exakt dem Text des originalen Werkes und werden nicht verändert.[3]. So werden die Regeln der strukturierten Erfassung in Kapitel 5 sehr strikt formuliert, um eventuelle Fehler im Quellenmaterial aufzufinden[4]. Alle folgenden Schritte bauen auf diesem Material auf. Falls in Ausnahmefällen eine Vorbehandlung des Textes vor der weiteren Bearbeitung notwendig ist, ist diese Teil des Umwandlungsprozesses und es findet keine Änderung der Eingangsdaten statt. Somit kann dieser Text einerseits (neben den Scans, s. Kap. 12.2) zum Abgleich mit den erzeugten Daten bzw. deren Repräsentation im Onlineportal verwendet werden, andererseits stellt er einen eigenen Datensatz dar, der zur weiteren Verwendung exportiert werden kann (vgl. Kap. 13.1).

Alle Daten werden in einer relationalen Datenbank abgelegt, das gilt für die Eingangsdaten[5] wie für die daraus erstellten eigentlichen Resultatdaten. Relationale Datenbanken legen Daten ausschließlich in tabellarischer Form ab. Verbindungen zwischen den Tabellen werden über Identifikatoren erzeugt, die in anderen Tabellen referenziert werden. Jede Tabelle kann dazu einen sogenannten *Primärschlüssel* (aus einer oder mehreren Spalten) definieren, der eine Tabellenzeile eindeutig identifiziert. Umgekehrt kann eine referenzierende Tabelle eine oder mehrere Spalten als

[3] Es werden allerdings sehr offensichtliche Fehler im Quellenmaterial wie fehlende schließende Klammern, Satzzeichen oder klare Tippfehler korrigiert (vgl. Kap. 8.1).

[4] Beispielsweise wird nach Kommata immer ein Leerzeichen erwartet, obwohl eine weniger strikte Formulierung dieses optional machen würde.

[5] Die Zeilen der Quelle werden nach den Operationen in Kapitel 4 in die Datenbank importiert. Alle weiteren Schritte finden in dieser bzw. im entsprechenden Webportal statt.

Fremdschlüssel definieren, sodass die Werte dort dem Primärschlüssel einer anderen Tabelle entsprechen müssen. So kann die referentielle Integrität sichergestellt werden. In der Praxis ist es meist sinnvoll eine einzelne numerische Spalte als Primärschlüssel zu verwenden, deren Werte automatisch hochgezählt werden. Ein weiteres Konzept aus dem Bereich der relationalen Datenbanken ist die sogenannte *Normalisierung*. Dabei wird die Konzeption einer Datenbank nach gewissen Regeln beschrieben, die dazu dienen redundante Angaben möglichst zu verhindern und so potentielle Fehler und ungültige Datenbankzustände zu vermeiden. Auf eine detaillierte Erklärung der mathematisch durchaus anspruchsvollen Grundlagen wird hier verzichtet, eine solche findet sich beispielsweise in Unterstein und Matthiessen 2012. Unter praktischen Gesichtspunkten führt eine solche Modellierung zwar zu einer Struktur, die viele Arten von Fehlern unmöglich macht, aber auch zu einer hohen Anzahl unterschiedlicher Tabellen, was durchaus auch einer Herausforderung sein kann (vgl. Kap. 3.4).

In vielen Fällen werden die erzeugten Primärdaten mit weiteren externen Quellen vernetzt oder in anderer Form angereichert (vgl. vor allem Kap. 10.2 und Kapitel 11). Im Widerspruch zum sonstigen Desgin von relationalen Datenbanken werden Verknüpfungen zwischen den Kerndaten und zusätzlichen Informationen nicht auf Basis der eindeutigen numerischen Identifikatoren vorgenommen. Der Grund hierfür ist, dass diese sich bei Erstellung jeder neuen Version ändern. Wird beispielsweise ein Bibliographie-Eintrag korrigiert und eine neue Version von diesem erstellt, sollte die Verknüpfung zu einer externen Quelle (vgl. Kap. 11.1) weiterhin verwendbar sein. Somit werden an diese Stelle andere (nicht eindeutige) Identifikatoren verwendet, die versionsübergreifend gültig sind (in diesem Fall die bibliographische Abkürzung[6]). Ist es aus Effizienzgründen sinnvoll, dass eine reguläre Verknüpfung über die numerischen IDs in der Datenbank vorhanden ist, so kann diese automatisiert beim Erstellen der jeweiligen Version zusätzlich erstellt werden.

3.2 Behandlung von Fehlern im Prozessablauf und Versionierung

Prozesse, in denen bestimmte Eingangsdaten verarbeitet und die Resultate als neue Daten abgelegt werden, kommen in nahezu allen digitalen Projekten vor. Oftmals werden diese allerdings als einmalige Vorgänge aufgefasst, die nach einer gewissen Test- und Optimierungsphase für alle Daten ausgeführt werden und damit abgeschlossen sind. Werden Fehler oder Probleme entdeckt, können diese dann nur entweder in den Resultatdaten selbst korrigiert werden oder bei schwerwiegenden systematischen Fehlern eine Veränderung der Importroutine und ein vollständiger Neuimport durchgeführt werden, der dann allerdings alle anderen eventuell vorhandenen

[6] Ändert sich die Abkürzung selbst, ist die Verknüpfung in dieser Form nicht mehr nutzbar. In diesem Fall müssten auch die Vernetzungsdaten angepasst werden. In der Praxis ist ein solcher Fall allerdings unwahrscheinlich.

Korrekturen überschreibt. Gerade bei komplexen Systemen kann hier ein weiterer Nachteil sein, dass Änderungen in der Importroutine zum Teil schwer vorhersehbare Nebeneffekte haben, sodass zwar bestimmte Problemfälle behoben werden, aber an anderer Stelle neue Fehler auftreten. Aufgrund dieser Problematik ist der Neuimport in vielen Fällen die schlechtere Lösung, was zu einem erheblichen manuellen (oder teil-automatisierten) Korrekturaufwand führt, der wiederum eine Quelle für neue Fehler bzw. eine unvollständige Umsetzung der Verbesserungen ist.

Gerade bei der Tiefenerschließung eines Wörterbuchtexts sind die grundlegenden Arbeitsschritte (vgl. Kapitel 5 und 7) allerdings hochkomplex, sodass eine längerfristige Anpassung und Verbesserung von diesen unvermeidbar ist. Deshalb ist es sinnvoll ein System zu verwenden, das einen (teilweisen) Neu-Import (sowohl bei Änderungen in den Eingangsdaten als auch bei Änderungen der algorithmischen Verarbeitung) ohne die erwähnten Nachteile unterstützt. Gleichzeitig ist es bei Eingangsdaten einer gewissen Größe illusorisch eine perfekte Routine zu erstellen, die alle vorkommenden Fälle korrekt verarbeitet. Es ist somit eine einzelfallbasierte Anpassung nötig, die aber so in den Verarbeitungsprozess eingebunden ist, dass sie bei einem erneuten Import ebenfalls verwendet werden kann.

Vor allem im wissenschaftlichen Bereich ist aber auch die mangelnde Stabilität und die fehlende Nachvollziehbarkeit von Änderungen gerade bei Onlineportalen problematisch. Werden die Änderungen des Datenbestands nicht gesondert dokumentiert (vgl. z.B. Bürgermeister 2019) oder der vollständige Datenbestand in festen Zeitintervallen versioniert (vgl. z.B. Lücke 2021b), ist eine Zitation nach den für gedruckten Werken verwendeten Maßgaben nicht möglich. Letzteres hat allerdings den Nachteil, dass neuere Änderungen erst mit Erreichen des nächsten Versionierungszeitpunkts stabil und damit zitierbar sind. Gerade im Kontext eines Redaktionssystems, das jederzeit die Änderungen von Nutzenden vorsieht (vgl. Kap. 3.3) ist dies allerdings ungünstig und die sofortige Erzeugung einer zitierbaren Version wäre wünschenswert. Erstere Lösung basiert wiederum auf einer Korrektur von Einzeldatensätzen, die schlecht mit Änderungen in der algorithmischen Verarbeitung vereinbar ist.

Hier wird also (wie bereits zu Beginn von Kapitel 3 angesprochen) eine fundamental andere Sicht auf den Gesamtprozess verwendet. Das System ist so aufgebaut, dass es ständige Verbesserungen sowohl der Eingangsdaten als auch des Transformationsprozesses und daraus folgende Anpassungen der Ausgangsdaten unterstützt. Beide Kategorien von Daten sind dabei statisch. Die Eingangsdaten ändern sich nach dem initialen Import grundsätzlich nicht mehr, während bei den Ausgangsdaten jeweils eine neue Version des entsprechenden Objekts erzeugt wird, sodass bestehende Einträge ebenfalls dauerhaft stabil sind. Alle Änderungen werden als explizite Datensätze einer dritten Kategorie dargestellt (vgl. auch Kapitel 6). Somit werden sowohl Korrekturen der Eingangsdaten als explizite Einträge in der Datenbank modelliert (vgl. Kap. 6.1), als auch alle individuellen Eingriffe in den Prozessablauf. Diese werden im weiteren als *Ausnahmen* bezeichnet. Sie können in verschiedenen

Kontexten definiert werden und dienen zur Anpassung einzelner Ergebnisse, bei denen die allgemeinen Routinen zu falschen Ergebnissen führen bzw. als eine Form von Annotation, damit bestimmte Bestandteile korrekt erkannt werden. Als Beispiel soll hier eine Liste von Bedeutungen in REW, 7473b dienen:

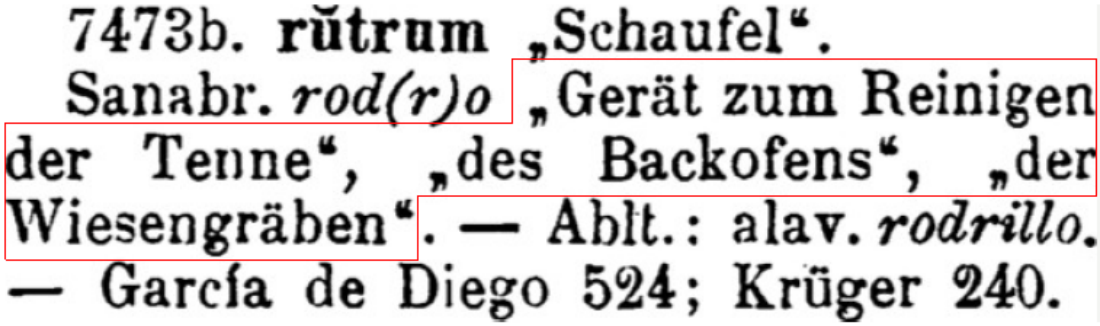

7473b. **rŭtrum** „Schaufel“.
Sanabr. *rod(r)o* „Gerät zum Reinigen der Tenne“, „des Backofens“, „der Wiesengräben“. — Ablt.: alav. *rodrillo*. — García de Diego 524; Krüger 240.

Abbildung 3.3: Beispiel für abgekürzte Bedeutungen, die zur korrekten Erfassung die Verwendung von Ausnahmen benötigen

Hier werden einer Form drei Bedeutungen zugewiesen, wobei die letzten beiden abgekürzt sind. Da die abkürzenden Schreibweisen sich nicht strukturell von anderen Bedeutungsangaben unterscheiden, können sie nicht als solche erkannt werden[7]. Es würden somit drei Bedeutungen „Gerät zum Reiningen der Tenne“, „des Backofens“ und „der Wiesengräben“ erzeugt. Diese sind im Kontext des Artikels leicht verständlich, falls aber beispielsweise ein Vergleich mit anderen Formen, die diese Bedeutungen haben, stattfinden soll (innerhalb des REW oder auch im Abgleich mit anderen Quellen) ist diese Notation wenig hilfreich. Somit kann hier mit Ausnahmen ein besseres Ergebnis erzeugt werden. In diesem Fall sind im hier verwendeten System zwei Formen von Ausnahmen nötig, zum einen auf struktureller Ebene (vgl. Kap. 5.2.6), damit die Zeichenketten „der Tenne“ und „des Backofens“ entsprechend als abkürzende Schreibweisen erkannt werden und zum anderen auf Verarbeitungsebene (vgl. Kap. 7.3), um anzugeben wie die Abkürzungen aufgelöst werden sollen[8]. Im Endresultat werden somit die beiden Bedeutungen „Gerät zum Reinigen des Backofens“ und „Gerät zum Reinigen der Tenne“ erstellt.

Bei der Definition des Formats für die verschiedenen Ausnahmen spielen folgende Überlegungen eine Rolle. Einerseits sollten die Typen der Ausnahmen allgemein genug sein, um in möglichst vielen verschiedenen Kontexten angewendet werden zu können.

[7] Meistens könnte man auch die Routine anpassen, um bestimmte Spezialfälle ebenfalls korrekt zu verarbeiten. Hier ist vor allem die Häufigkeit des Auftretens relevant, die in diesem Fall nicht groß genug ist, um eine eigene Regel zu rechtfertigen (s. u.). Weiterhin könnte eine solche Regel je nach Ausgestaltung auch wieder zu neuen Fehlern an anderer Stelle führen.

[8] Auch hier könnte eine Routine aufgestellt werden, die versucht diese Abkürzungen automatisch aufzulösen. Dies ist nicht der Fall, da die abkürzenden Schreibweisen bei Bedeutungen sehr unterschiedlich sind und keine Variante häufig genug vorkommt, um dies zu rechtfertigen. Somit müssen alle Bedeutungsabkürzungen über explizite Ausnahmen aufgelöst werden. Für Abkürzungen bei sprachlichen Formen existiert aber beispielsweise eine solche Routine (vgl. Kap. 7.1.3)

Sie sollten also mächtig genug sein, um im Einzelfall ein Maximum an extrahierter Information zu ermöglichen, auch wenn dies in der Praxis zum Teil einen massiven manuellen Korrekturaufwand bedeuteten würde und nicht unbedingt immer großflächig angewandt werden kann. Es sollte aber zumindest das Potential vorhanden sein, sodass beispielsweise einzelne Fälle, die für eine bestimmte linguistische Fragestellung relevant sind, intensiver nachgebessert werden können, um für diese eine verbesserte (zitierbare) Variante zu erzeugen. Andererseits sollten sie nicht zu allgemein sein, da die Bündelung von gleichartigen Fällen im weiteren Projektverlauf leichter erkennbar ist und die Ausnahmen mit hoher Häufigkeit zur Anpassung der entsprechenden Routinen verwendet werden können. Dabei ist einerseits die manuelle Erweiterung der entsprechenden Routine gemeint, aber auch beispielsweise die Verwendung zusammen mit Methoden des maschinellen Lernens zur Behandlung von zukünftigen ähnlichen Fällen (vgl. Kapitel 14). Eine konkrete Ausgestaltung der verschiedenen Typen von Ausnahmen wird in Kap. 7.3 besprochen.

Weiterhin ist es sinnvoll den Kontext, in dem eine bestimmte Ausnahme definiert wird, möglichst unabhängig von konkreten Identifikatoren zu definieren. Das Ziel ist hierbei, dass die Ausnahmen auch bei eventuellen Änderungen an anderer Stelle möglichst gültig bleiben. Gerade zu Beginn der Erstellung des entsprechenden Umwandlungsroutinen können somit auch alle Daten vollständig neu importiert werden, ohne dass die entsprechenden Ausnahmen unbrauchbar werden. Gerade bei größeren konzeptuellen Änderungen ist dies hilfreich[9].

Insgesamt können mit dem beschriebenen System systematische Fehler durch Anpassung der Importroutinen behoben werden, ohne dass bestehende Einzeländerungen verloren gehen[10]. Eine erneute Verarbeitung aller Eingangsdaten ist jederzeit möglich, wobei für veränderte Objekte eine neue Version unter Markierung des Zeitpunkts erstellt wird. Gleichzeitig können so auch bei Algorithmusanpassungen anhand der bestehenden Objekte (oder einer Stichprobe) die Änderungen validiert und eventuell an anderer Stelle auftretende Fehler aufgefunden werden (vgl. Kap. 3.4.1).

3.3 Umsetzung im Kontext eines Redaktionssystems

Bestehende Webportale, die auf Digitalisierungen von traditionellen Werken beruhen, sind in der Betriebsphase meist sehr statisch angelegt. Falls überhaupt Korrekturen möglich ist, können diese maximal unsystematisch über Kommunikation mit den entsprechenden Verantwortlichen gemeldet werden. Exemplarisch hierfür ist die folgende Angabe:

[9] Selbstverständlich wird damit die Versionierung umgangen, d.h. ab dem Zeitpunkt der erstmaligen Publikation sollte dies vermieden werden.

[10] Bei sehr drastischen Änderungen könne einzelne solche Fälle allerdings auch nicht immer verhindert werden.

> Falls Sie einen Erfassungsfehler entdecken, dann schreiben Sie uns bitte unter Angabe der Wörterbuchsigle und der betreffenden Kontextstelle.(FAQ Wörterbuchnetz)

Dies hat den Nachteil, dass ein gewisser redaktionelle Aufwand durch solche Anfragen entsteht, vor allem wenn mehrere Instanzen involviert sind, weil beispielsweise erst wissenschaftliches Personal eine Korrektur verifizieren muss, bevor sie dann von technischem Personal eingepflegt werden kann. Umgekehrt baut es aber auch eine gewisse Hemmschwelle auf Seiten der Nutzenden auf, die einerseits nicht genau wissen, wie lange es dauert, bis ein bestimmter Fehler tatsächlich korrigiert wird, und anderseits unter Umständen erst Kontaktadressen suchen und Nachrichten formulieren müssen.

Zur Lösung dieses Problems wird das entstandene Webportal hier als Redaktionssystem konzipiert, das sowohl projektintern die weitere Korrektur und Verarbeitung der generierten Resultatdaten erlaubt, als auch externen Nutzenden (gegebenenfalls in eingeschränkter Form) die selben Möglichkeiten bietet. Insbesondere die Vorteile des Ansatzes aus dem vorherigen Kapitel können somit voll ausgenutzt werden, indem man Nutzenden nach einer Fehlerkorrektur den Anstoß eines Neu-Imports ermöglicht, der unmittelbar eine neue zitierbare Version erzeugt. Das Gesamtsystem, dessen Bestandteile in den nächsten Kapitel im Detail besprochen werden, kann insgesamt folgendermaßen illustriert werden:

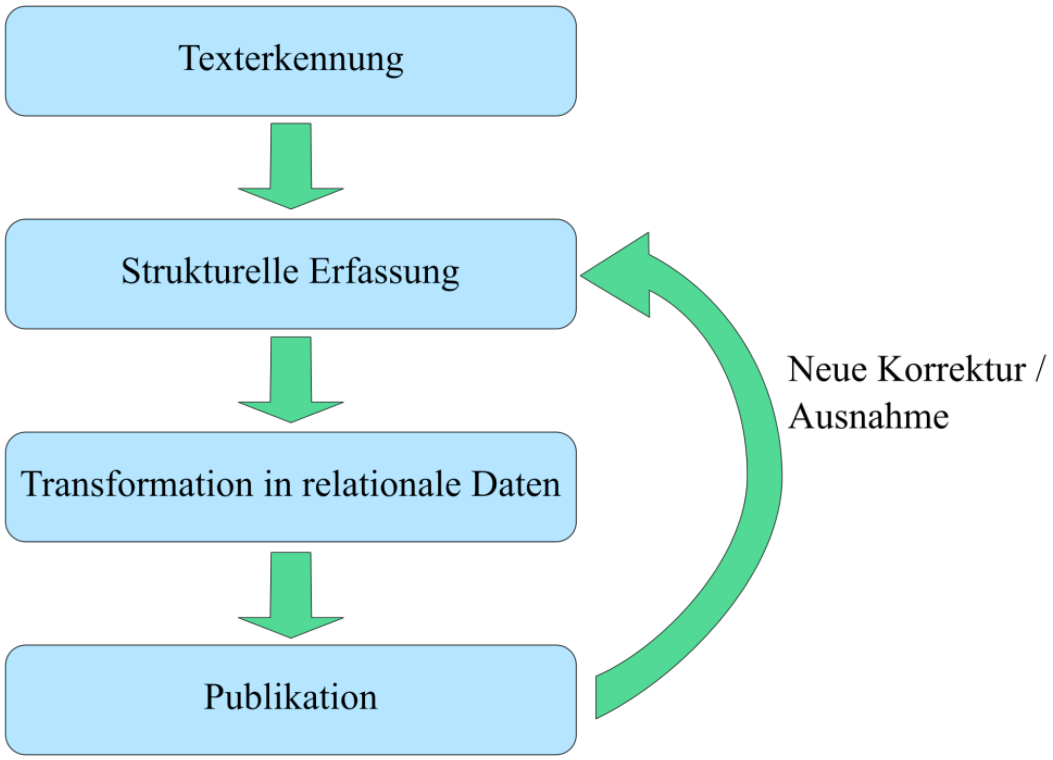


Abbildung 3.4: Schematische Darstellung der Konzeption des Transformationsprozesses

Alle Schritte (mit Ausnahme der initialen Texterkennung) können somit innerhalb des Webportals wiederholt werden, um eine verbesserte Version (in diesem Fall) eines Wörterbuchartikels zu erstellen. Die genaue Ausgestaltung eines solchen Portals wird in Kapitel 12 betrachtet.

3.4 Ergebnisdaten im relationalen Modell

Die Ergebnisse des Transformationsprozesses können als eine Menge von Tupeln aufgefasst werden, die in die relationale Datenbank eingefügt werden. Darin ist mindestens ein Eintrag enthalten, der die Daten für die eigentliche Basistabelle enthält (z.B. Wörterbucheinträge, Bibliographie-Einträge, etc.). Weitere Einträge sind über Fremdschlüssel direkt oder indirekt mit der Basistabelle verknüpft. Schon bei einfachen Objekten können in einer normalisierten Datenbank bereits einige solcher Tupel vorhanden sein. Als Beispiel soll hier der folgende Bibliographie-Eintrag aus dem REW dienen:

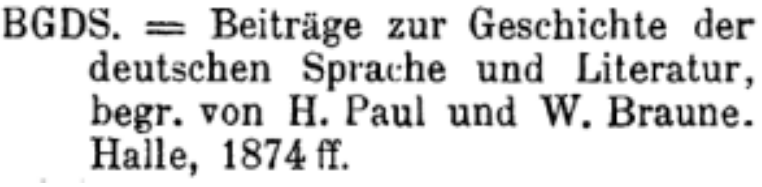

BGDS. = Beiträge zur Geschichte der deutschen Sprache und Literatur, begr. von H. Paul und W. Braune. Halle, 1874 ff.

Abbildung 3.5: Eintrag aus der Bibliographie in REW, S. XVI

Im folgenden werden die daraus generierten Datenbank-Einträge gezeigt:


```
[
    0: ["persons", {
            "prename": "H.",
            "surname": "Paul"
        }
    ],
    1: ["bib_person_connections", {
            "id_entry": "###6###",
            "connection_type": "founded",
            "prefix": "begr. von",
            "id_person": "###0###"
        }
    ],
    2: ["persons", {
            "prename": "W.",
            "surname": "Braune"
        }
    ],
    3: ["bib_person_connections", {
            "id_entry": "###6###",
            "connection_type": "founded",
            "prefix": "begr. von",
            "id_person": "###2###"
        }
    ],
    4: ["time_periods", {
            "name": "1874ff",
            "beginning": "1874",
            "end": null
        }
    ],
    5: ["bib_publication_data", {
            "id_entry": "###6###",
            "location": "Halle",
```

```
            "id_time_period": "###4###"
        }
    ],
    6: ["bibliography", {
            "abbreviation": "BGDS.",
            "type": "entry",
            "title": "Beiträge zur Geschichte der deutschen Sprache und Literatur"
        }
    ]
]
```

Insgesamt werden sieben verschiedene Zeilen erzeugt:

- Zwei Einträge in die Tabelle *persons*
- Zwei Einträge in die Tabelle *bib_person_connections*, die die Bibliographie-Einträge mit Personen verknüpft (in diesem Fall die Herausgeber)
- Ein Eintrag in die Tabelle *time_periods*
- Ein Eintrag in die Tabelle *bib_publication_data*, die einen Bibliographie-Eintrag mit Ort und Zeitpunkt der Publikation verknüpft und
- Ein Eintrag in die Basistabelle *bibliography*

Fremdschlüssel, die auf andere Tabellen verweisen werden hier in mit drei #-Zeichen umschlossen dargestellt und enthalten den Index des jeweiligen Eintrags auf sie verweisen. Bei den Tabellen werden außerdem zwei verschiedene Klassen unterschieden, die hier *Objekttabellen* und *Geteilte Tabellen* genannt werden. Objekttabellen enthalten Informationen, die sich nur auf das Objekt selbst beziehen und seine aktuelle Version festlegen. Geteilte Tabellen sind solche, die zwar auch im Importprozess erstellt werden, aber Einträge enthalten, die potentiell von verschiedenen Objekten referenziert werden können. Insbesondere werden innerhalb dieser Tabellen keine Duplikate angelegt. Im Beispiel sind *bibliography*, *bib_person_connections* und *bib_publication_data* Objekttabellen, während *persons* und *time_periods* geteilte Tabellen darstellen. So kann beispielsweise ein Autor bzw. Herausgeber mit mehreren Werken verknüpft werden. Die Definition, welche Tabelle zu welcher Klasse gehört ist entscheidend für den Importprozess, da bei den geteilten Tabellen nur ein neuer Eintrag erstellt wird, wenn nicht bereits ein identischer existiert, während für Objekttabellen immer ein neuer Eintrag angelegt wird.

Die Basistabellen enthalten zusätzlich zu den eigentlichen inhaltlichen Daten weitere Spalten, die für die Verwaltung und Versionierung gebraucht werden:

Spalte	Beschreibung
imported	Gibt den Zeitpunkt an, zu dem dieser Eintrag erstellt wurde.
replaced	Gibt den Zeitpunkt an, zu dem dieser Eintrag durch eine neuere Version ersetzt wurde (oder leer)
newer_variant	Gibt die ID der Eintrags an, der diesen ersetzt hat (oder leer)
needs_update	Markiert, falls dieser Eintrag sich verändert hat und neu importiert werden muss. Das ist beispielsweise der Fall, wenn eine der Zeilen, aus denen er erzeugt wurde korrigiert wurde.

Bei den Feldern *replaced* und *newer_variant* ist noch zu beachten, dass es zwei mögliche Kombinationen gibt: Während *replaced* immer gesetzt wird, sobald der Eintrag veraltet ist, wird *newer_variant* nicht zwingend gefüllt. Dazu muss man zwei Arten von neuen Versionen unterscheiden, diejenigen, bei denen sich die Zeilenzuordnung geändert hat und diejenigen, bei denen dies nicht der Fall ist. Ersteres beschreibt den regulären Fall, in dem beispielsweise eine Korrektur in einem Wörterbuchartikel stattgefunden hat und die neue Version exakt diesen Eintrag ersetzt. Der zweite Fall kann auftreten, wenn beispielsweise eine Artikelgrenze nicht erkannt wurde und anstatt eines alten Artikels zwei neue erstellt werden. In diesem Fall ist nicht definiert, welcher von den beiden neuen Artikeln den alten ersetzt, somit bleibt das Feld *newer_variant* leer.

3.4.1 Vergleich mit vorhandenen Daten

Um zu erkennen, ob sich die Daten zu einem bestimmten Objekt geändert haben, ist ein Abgleich der neu erstellten Tupel mit den bestehenden Zeilen in der Datenbank notwendig. In Fällen mit vielen Tupeln ist dies nicht immer trivial. Zum einen muss eine genaue Definition der relevanten Tabellen und ihrere Beziehungen untereinander vorhanden sein, sodass die Vergleichsroutine die relevanten Elemente aus den verschiedenen Tabellen rekonstruieren kann. Um dies zu vereinfachen werden per Konvention alle Tabellen mit einem einspaltigen numerischen Primärschlüssel als Identifikator definiert. Zum anderen müssen die Beziehungen zwischen den einzelnen Elementen aufgelöst werden. In der Datenbank werden diese durch konkrete Zahlen dargestellt, die auf den jeweiligen anderen Eintrag verweisen, während die Tupel relative Angaben bezogen auf ihre Indizierung enthalten. Im Fall von identischen Daten können die konkreten IDs und die relativen Indexe sowie die restlichen Spaltenwerte 1:1 aufeinander abgebildet werden. Ist das nicht der Fall wird trotzdem versucht eine möglichst zutreffende Abbildung zu erstellen, um leichter erkennen zu können, an welchen Stellen sich die Daten seit der letzten Version verändert haben:

New row (7): [form => „alcuña , id_lang => „###6### , pos => , num => 1 , lang_unsure => false , learned_word => false , reconstructed => false , gender => , reflexive => false , number => , word_type => , person => , case => , mood =>] Old row (54759): [form => „alouña , id_lang => „11 ,

pos => , num => „1 , lang_unsure => „0 , learned_word => „0 , reconstructed => „0 , gender => , reflexive => „0 , number => , word_type => , person => , case => , mood =>]

Mit diesen Vergleichsoperationen ist es unabhängig von der Versionierung auch möglich bei Änderungen der formellen Grammatik für die strukturelle Erkennung (vgl. Kapitel 5) oder ähnlichen Anpassungen die bestehenden Artikel (insbesondere jene die intensiv manuell korrigiert wurden) systematisch zu überprüfen. Diese können somit als Testfälle dienen, ob durch die Änderungen an anderer Stelle „Kollateralschäden“ auftreten bzw. um tatsächlich richtige Änderungen aufzufinden und zu verifizieren.

4 Texterkennung und Post-Processing

In diesem Kapitel wird die erste Phase des Transformationsprozesses behandelt, die aus gescannten Buchseiten digitalen Text erzeugt, der in geeigneter Weise (vgl. Kap. 6.1) in einer Datenbank abgelegt werden kann. Dies ist der einzige Abschnitt, der als *Import* im strengeren Sinne des Wortes bezeichnet werden kann, gleichzeitig ist es der einzige Schritt, der außerhalb des Webportals stattfindet. Die weitere Verarbeitung findet ausschließlich dort statt, sodass jeder Arbeitsschritt bei fehlerhaften Eingangsdaten oder Ergebnissen nach dem Modell des vorherigen Kapitels angepasst und wiederholt werden kann. Der Verlauf kann dabei in drei Teilschritte eingeteilt werden: Zuerst wird die gescannte Seite entsprechend ihrer Grobstruktur segmentiert (Kap. 4.1), danach findet die eigentliche Texterkennung statt (Kap. 4.2), während zuletzt ein Post-Processing-Schritt durchgeführt wird, der strukturelle Fehler verbessert (Kap. 4.3).

Als Grundlage wurde eine bestehende digitalisierte Ausgabe des REW in der dritten Auflage verwendet. Diese ist unter https://nbn-resolving.org/urn:nbn:de:bvb:355-ubr07799-0 frei zugänglich. Für die Durchführung der eigentlichen Texterkennung *(Optical Character Recognition = OCR)* wurde die freie Software *tesseract*[1] in der Version 3.05[2] genutzt.

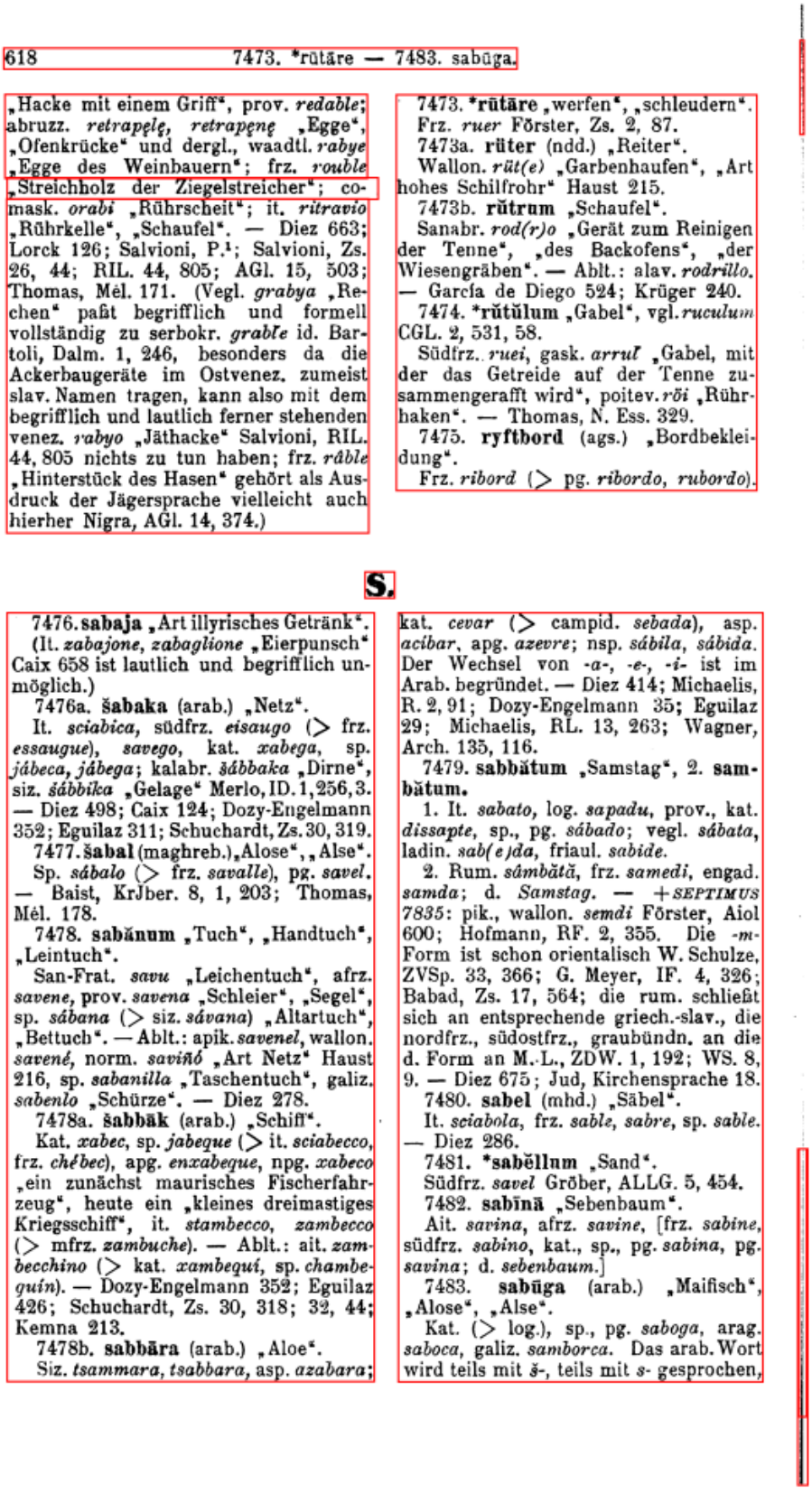
4.1 Segementierung der Scans

OCR-Systeme führen initial eine Erkennung von Textsegmenten innerhalb der zugrundeliegenden Bilder durch[3]. Für eine Beispielseite erzeugt *tesseract* folgendes Ergebnis:

[1] https://github.com/tesseract-ocr/tesseract

[2] *Tesseract* verwendet ab Version 4.0 eine neue Methode zur Texterkennung unter Verwendung von neuronalen Netzen, die in den meisten Fällen zu besseren Ergebnissen führt (vgl. tesseract-ocr o. D.). Der Hauptgrund, warum diese nicht verwendet wurde, ist, dass diese zum aktuellen Zeitpunkt keine Erkennung der Formatierung unterstützt (vgl. *https://github.com/tesseract-ocr/tesseract/issues/684*). Der maschinenlernengestütze Ansatz aus Kap. 4.3 kommt grundsätzlich auch ohne Textdaten mit expliziter Markierung der Formatierung aus, für das Erstellen der initialen Trainingsdaten, ist diese allerdings nützlich. Eine hybride Verwendung beider Versionen ist aufgrund verschiedener Formate für die Trainingsdaten zur Texterkennung aufwendig, wäre aber prinzipiell möglich.

[3] Bei *tesseract* kann bis zu einem gewissen Maße Einfluss darauf genommen werden, um sehr spezielle Fälle wie einzelne Textzeilen oder Wörter abzudecken, vgl. https://tesseract-ocr.github.io/tessdoc/ImproveQuality.html#page-segmentation-method.



618 7473. *rūtāre — 7483. sabūga.

„Hacke mit einem Griff“, prov. *redable*; abruzz. *retrapęlę*, *retrapęnę* „Egge“, „Ofenkrücke“ und dergl., waadtl. *rabye* „Egge des Weinbauern“; frz. *rouble* „Streichholz der Ziegelstreicher“; comask. *orabi* „Rührscheit“; it. *ritravio* „Rührkelle“, „Schaufel“. — Diez 663; Lorck 126; Salvioni, P.[1]; Salvioni, Zs. 26, 44; RIL. 44, 805; AGl. 15, 503; Thomas, Mél. 171. (Vegl. *grabya* „Rechen“ paßt begrifflich und formell vollständig zu serbokr. *grablje* id. Bartoli, Dalm. 1, 246, besonders da die Ackerbaugeräte im Ostvenez. zumeist slav. Namen tragen, kann also mit dem begrifflich und lautlich ferner stehenden venez. *rabyo* „Jäthacke“ Salvioni, RIL. 44, 805 nichts zu tun haben; frz. *râble* „Hinterstück des Hasen“ gehört als Ausdruck der Jägersprache vielleicht auch hierher Nigra, AGl. 14, 374.)

7473. ***rūtāre*** „werfen“, „schleudern“. Frz. *ruer* Förster, Zs. 2, 87.

7473a. **rüter** (ndd.) „Reiter“. Wallon. *rüt(e)* „Garbenhaufen“, „Art hohes Schilfrohr“ Haust 215.

7473b. **rŭtrum** „Schaufel“. Sanabr. *rod(r)o* „Gerät zum Reinigen der Tenne“, „des Backofens“, „der Wiesengräben“. — Ablt.: alav. *rodrillo*. — García de Diego 524; Krüger 240.

7474. ***rŭtŭlum*** „Gabel“, vgl. *ruculum* CGL. 2, 531, 58. Südfrz. *ruei*, gask. *arrul* „Gabel, mit der das Getreide auf der Tenne zusammengerafft wird“, poitev. *rŏi* „Rührhaken“. — Thomas, N. Ess. 329.

7475. **ryftbord** (ags.) „Bordbekleidung“. Frz. *ribord* (> pg. *ribordo*, *rubordo*).

S.

7476. **sabaja** „Art illyrisches Getränk“. (It. *zabajone*, *zabaglione* „Eierpunsch“ Caix 658 ist lautlich und begrifflich unmöglich.)

7476a. **šabaka** (arab.) „Netz“. It. *sciabica*, südfrz. *eisaugo* (> frz. *essaugue*), *savego*, kat. *xabega*, sp. *jábeca*, *jábega*; kalabr. *šábbaka* „Dirne“, siz. *šábbika* „Gelage“ Merlo, ID. 1, 256, 3. — Diez 498; Caix 124; Dozy-Engelmann 352; Eguilaz 311; Schuchardt, Zs. 30, 319.

7477. **šabal** (maghreb.) „Alose“, „Alse“. Sp. *sábalo* (> frz. *savalle*), pg. *savel*. — Baist, KrJber. 8, 1, 203; Thomas, Mél. 178.

7478. **sabănum** „Tuch“, „Handtuch“, „Leintuch“. San-Frat. *savu* „Leichentuch“, afrz. *savene*, prov. *savena* „Schleier“, „Segel“, sp. *sábana* (> siz. *sávana*) „Altartuch“, „Bettuch“. — Ablt.: apik. *savenel*, wallon. *savené*, norm. *saviñó* „Art Netz“ Haust 216, sp. *sabanilla* „Taschentuch“, galiz. *sabenlo* „Schürze“. — Diez 278.

7478a. **šabbāk** (arab.) „Schiff“. Kat. *xabec*, sp. *jabeque* (> it. *sciabecco*, frz. *chébec*), apg. *enxabeque*, npg. *xabeco* „ein zunächst maurisches Fischerfahrzeug“, heute ein „kleines dreimastiges Kriegsschiff“, it. *stambecco*, *zambecco* (> mfrz. *zambuche*). — Ablt.: ait. *zambecchino* (> kat. *xambequí*, sp. *chambequín*). — Dozy-Engelmann 352; Eguilaz 426; Schuchardt, Zs. 30, 318; 32, 44; Kemna 213.

7478b. **sabbāra** (arab.) „Aloe“. Siz. *tsammara*, *tsabbara*, asp. *azabara*; kat. *cevar* (> campid. *sebada*), asp. *acíbar*, apg. *azevre*; nsp. *sábila*, *sábida*. Der Wechsel von *-a-*, *-e-*, *-i-* ist im Arab. begründet. — Diez 414; Michaelis, R. 2, 91; Dozy-Engelmann 35; Eguilaz 29; Michaelis, RL. 13, 263; Wagner, Arch. 135, 116.

7479. **sabbătum** „Samstag“, 2. **sambătum.** 1. It. *sabato*, log. *sapadu*, prov., kat. *dissapte*, sp., pg. *sábado*; vegl. *sábata*, ladin. *sab(e)da*, friaul. *sabide*.

2. Rum. *sâmbătă*, frz. *samedi*, engad. *samda*; d. *Samstag*. — +*SEPTIMUS* *7835*: pik., wallon. *semdi* Förster, Aiol 600; Hofmann, RF. 2, 355. Die *-m-* Form ist schon orientalisch W. Schulze, ZVSp. 33, 366; G. Meyer, IF. 4, 326; Babad, Zs. 17, 564; die rum. schließt sich an entsprechende griech.-slav., die nordfrz., südostfrz., graubündn. an die d. Form an M.-L., ZDW. 1, 192; WS. 8, 9. — Diez 675; Jud, Kirchensprache 18.

7480. **sabel** (mhd.) „Säbel“. It. *sciabola*, frz. *sable*, *sabre*, sp. *sable*. — Diez 286.

7481. ***saběllum*** „Sand“. Südfrz. *savel* Gröber, ALLG. 5, 454.

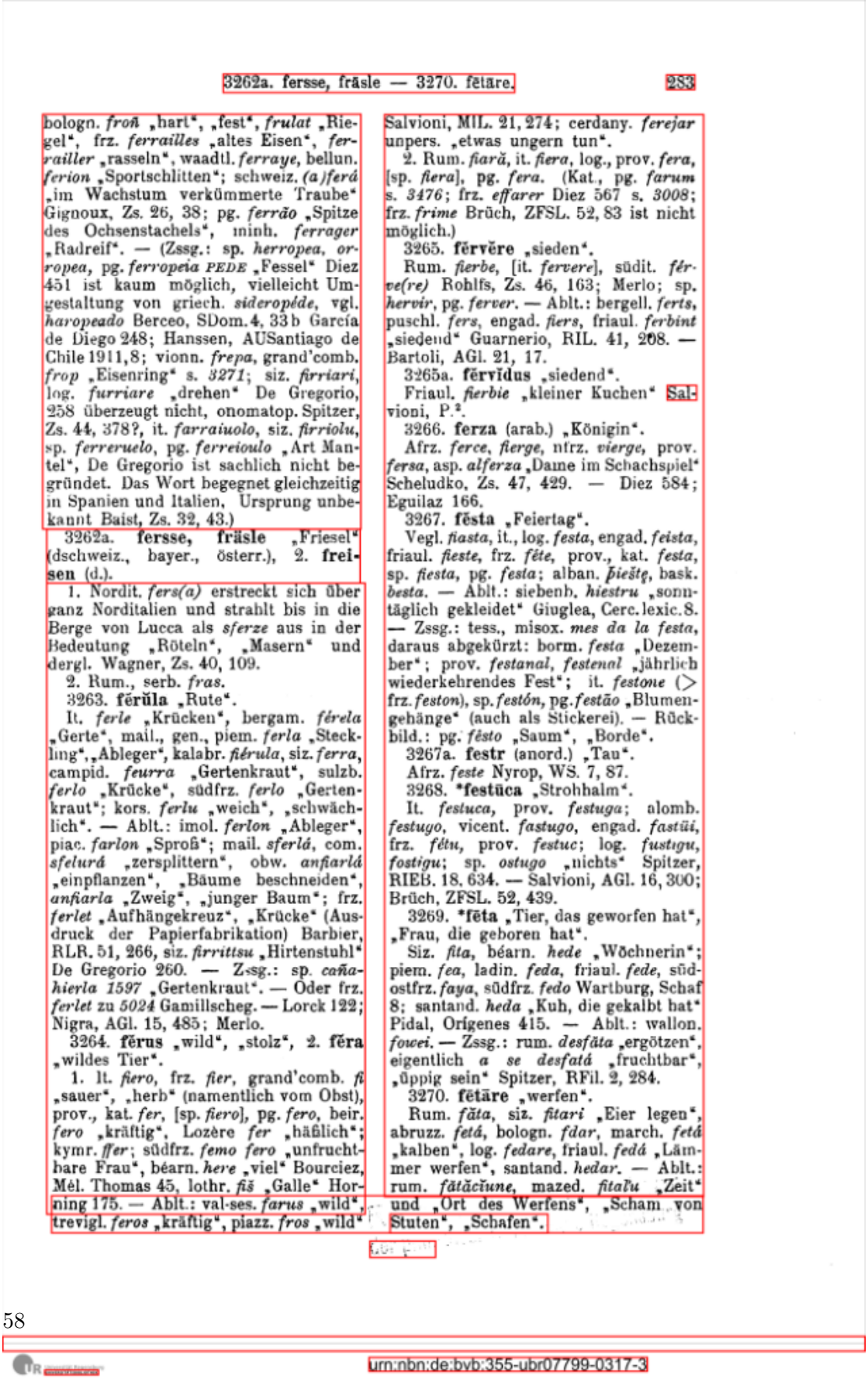
7482. **sabīnā** „Sebenbaum“. Ait. *savina*, afrz. *savine*, [frz. *sabine*, südfrz. *sabino*, kat., sp., pg. *sabina*, pg. *savina*; d. *sebenbaum*.]

7483. **sabūga** (arab.) „Maifisch“, „Alose“, „Alse“. Kat. (> log.), sp., pg. *saboga*, arag. *saboca*, galiz. *samborca*. Das arab. Wort wird teils mit *š-*, teils mit *s-* gesprochen,

Abbildung 4.1: Seitensegmentierung von *tesseract* für Seite 618 aus dem REW

Obwohl an den Rändern einige unerwünschte Elemente erkannt wurden[4], ist das Ergebnis in diesem Fall durchaus als gut zu bewerten. Bis auf eine Stelle links oben wurden die grundsätzlichen Textblöcke korrekt erkannt. Dies ist allerdings nicht immer der Fall. Zum Teil werden deutlich mehr inkorrekte Unterteilungen erkannt bis hin zu Überlappungen wie im unteren Teil dieser Seite:

[4]Die könnte durch einen Zuschnitt des Scans behoben werden.



3262a. fersse, fräsle — 3270. fētāre. 283

bologn. *froñ* „hart“, „fest“, *frulat* „Riegel“, frz. *ferrailles* „altes Eisen“, *ferrailler* „rasseln“, waadtl. *ferraye*, bellun. *ferion* „Sportschlitten“; schweiz. *(a)ferá* „im Wachstum verkümmerte Traube“ Gignoux, Zs. 26, 38; pg. *ferrão* „Spitze des Ochsenstachels“, mlnh. *ferrager* „Radreif“. — (Zssg.: sp. *herropea, orropea*, pg. *ferropeia* PEDE „Fessel“ Diez 451 ist kaum möglich, vielleicht Umgestaltung von griech. *sideropéde*, vgl. *haropeado* Berceo, SDom. 4, 33 b García de Diego 248; Hanssen, AUSantiago de Chile 1911, 8; vionn. *frepa*, grand'comb. *frop* „Eisenring“ s. *3271*; siz. *firriari*, log. *furriare* „drehen“ De Gregorio, 258 überzeugt nicht, onomatop. Spitzer, Zs. 44, 378?, it. *farraiuolo*, siz. *firriolu*, sp. *ferreruelo*, pg. *ferreioulo* „Art Mantel“, De Gregorio ist sachlich nicht begründet. Das Wort begegnet gleichzeitig in Spanien und Italien, Ursprung unbekannt Baist, Zs. 32, 43.)

3262a. **fersse, fräsle** „Friesel“ (dschweiz., bayer., österr.), 2. **freisen** (d.).

1. Nordit. *fers(a)* erstreckt sich über ganz Norditalien und strahlt bis in die Berge von Lucca als *sferze* aus in der Bedeutung „Röteln“, „Masern“ und dergl. Wagner, Zs. 40, 109.

2. Rum., serb. *fras*.

3263. **fĕrŭla** „Rute“.

It. *ferle* „Krücken“, bergam. *férela* „Gerte“, mail., gen., piem. *ferla* „Steckling“, „Ableger“, kalabr. *fiérula*, siz. *ferra*, campid. *feurra* „Gertenkraut“, sulzb. *ferlo* „Krücke“, südfrz. *ferlo* „Gertenkraut“; kors. *ferlu* „weich“, „schwächlich“. — Ablt.: imol. *ferlon* „Ableger“, piac. *farlon* „Sproß“; mail. *sferlá*, com. *sfelurá* „zersplittern“, obw. *anfiarlá* „einpflanzen“, „Bäume beschneiden“, *anfiarla* „Zweig“, „junger Baum“; frz. *ferlet* „Aufhängekreuz“, „Krücke“ (Ausdruck der Papierfabrikation) Barbier, RLR. 51, 266, siz. *firrittsu* „Hirtenstuhl“ De Gregorio 260. — Zssg.: sp. *cañahierla 1597* „Gertenkraut“. — Oder frz. *ferlet* zu *5024* Gamillscheg. — Lorck 122; Nigra, AGl. 15, 485; Merlo.

3264. **fĕrus** „wild“, „stolz“, 2. **fĕra** „wildes Tier“.

1. It. *fiero*, frz. *fier*, grand'comb. *fi* „sauer“, „herb“ (namentlich vom Obst), prov., kat. *fer*, [sp. *fiero*], pg. *fero*, beir. *fero* „kräftig“, Lozère *fer* „häßlich“; kymr. *ffer*; südfrz. *femo fero* „unfruchtbare Frau“, béarn. *here* „viel“ Bourciez, Mél. Thomas 45, lothr. *fiš* „Galle“ Horning 175. — Ablt.: val-ses. *farus* „wild“, trevigl. *feros* „kräftig“, piazz. *fros* „wild“ Salvioni, MIL. 21, 274; cerdany. *ferejar* unpers. „etwas ungern tun“.

2. Rum. *fiară*, it. *fiera*, log., prov. *fera*, [sp. *fiera*], pg. *fera*. (Kat., pg. *farum* s. *3476*; frz. *effarer* Diez 567 s. *3008*; frz. *frime* Brüch, ZFSL. 52, 83 ist nicht möglich.)

3265. **fĕrvĕre** „sieden“.

Rum. *fierbe*, [it. *fervere*], südit. *férve(re)* Rohlfs, Zs. 46, 163; Merlo; sp. *hervir*, pg. *ferver*. — Ablt.: bergell. *ferts*, puschl. *fers*, engad. *fiers*, friaul. *ferbint* „siedend“ Guarnerio, RIL. 41, 208. — Bartoli, AGl. 21, 17.

3265a. **fĕrvĭdus** „siedend“.

Friaul. *fierbie* „kleiner Kuchen“ Salvioni, P.[2].

3266. **ferza** (arab.) „Königin“.

Afrz. *ferce, fierge*, nfrz. *vierge*, prov. *fersa*, asp. *alferza* „Dame im Schachspiel“ Scheludko, Zs. 47, 429. — Diez 584; Eguilaz 166.

3267. **fēsta** „Feiertag“.

Vegl. *fiasta*, it., log. *festa*, engad. *feista*, friaul. *fieste*, frz. *fête*, prov., kat. *festa*, sp. *fiesta*, pg. *festa*; alban. *pieštę*, bask. *besta*. — Ablt.: siebenb. *hiestru* „sonntäglich gekleidet“ Giuglea, Cerc. lexic. 8. — Zssg.: tess., misox. *mes da la festa*, daraus abgekürzt: borm. *festa* „Dezember“; prov. *festanal, festenal* „jährlich wiederkehrendes Fest“; it. *festone* (> frz. *feston*), sp. *festón*, pg. *festão* „Blumengehänge“ (auch als Stickerei). — Rückbild.: pg. *fêsto* „Saum“, „Borde“.

3267a. **festr** (anord.) „Tau“.

Afrz. *feste* Nyrop, WS. 7, 87.

3268. ***festūca** „Strohhalm“.

It. *festuca*, prov. *festuga*; alomb. *festugo*, vicent. *fastugo*, engad. *fastüi*, frz. *fétu*, prov. *festuc*; log. *fustigu, fostigu*; sp. *ostugo* „nichts“ Spitzer, RIEB. 18, 634. — Salvioni, AGl. 16, 300; Brüch, ZFSL. 52, 439.

3269. ***fēta** „Tier, das geworfen hat“, „Frau, die geboren hat“.

Siz. *fita*, béarn. *hede* „Wöchnerin“; piem. *fea*, ladin. *feda*, friaul. *fede*, südostfrz. *faya*, südfrz. *fedo* Wartburg, Schaf 8; santand. *heda* „Kuh, die gekalbt hat“ Pidal, Orígenes 415. — Ablt.: wallon. *fowei*. — Zssg.: rum. *desfăta* „ergötzen“, eigentlich *a se desfată* „fruchtbar“, „üppig sein“ Spitzer, RFil. 2, 284.

3270. **fētāre** „werfen“.

Rum. *făta*, siz. *fitari* „Eier legen“, abruzz. *fetá*, bologn. *fdar*, march. *fetá* „kalben“, log. *fedare*, friaul. *fedá* „Lämmer werfen“, santand. *hedar*. — Ablt.: rum. *fătăciune*, mazed. *fitaľu* „Zeit“ und „Ort des Werfens“, „Scham von Stuten“, „Schafen“.

Abbildung 4.2: Seitensegmentierung von *tesseract* für Seite 283 aus dem REW

Dies kann es durchaus kompliziert machen die einzelnen Teile und ihre Anordnung zueinander korrekt zu rekonstruieren. Außerdem ist eine klare Erkennung der Textspalten auch für die in Kap. 12.5 vorgestellten Korrekturwerkzeuge notwendig, sodass in diesem Fall eine robustere Erkennung der Seitensegmente vorgenommen wurde. Der einfache Algorithmus basiert dabei auf Histogrammen der jeweiligen Scans[5] und nutzt die spezielle Spaltenstruktur vieler traditioneller Wörterbücher. Im ersten Schritt werden unter Verwendung von Histogrammen auf Basis der einzelnen Bildspalten die vertikalen Grenzen der Textblöcke gesucht. Der Algorithmus beginnt dabei in einem Bereich der mit Sicherheit zum linken Textblock gehört und wandert nach rechts, bis bei Unterschreitung eines Schwellwerts der Beginn der Mitte erreicht wurde:

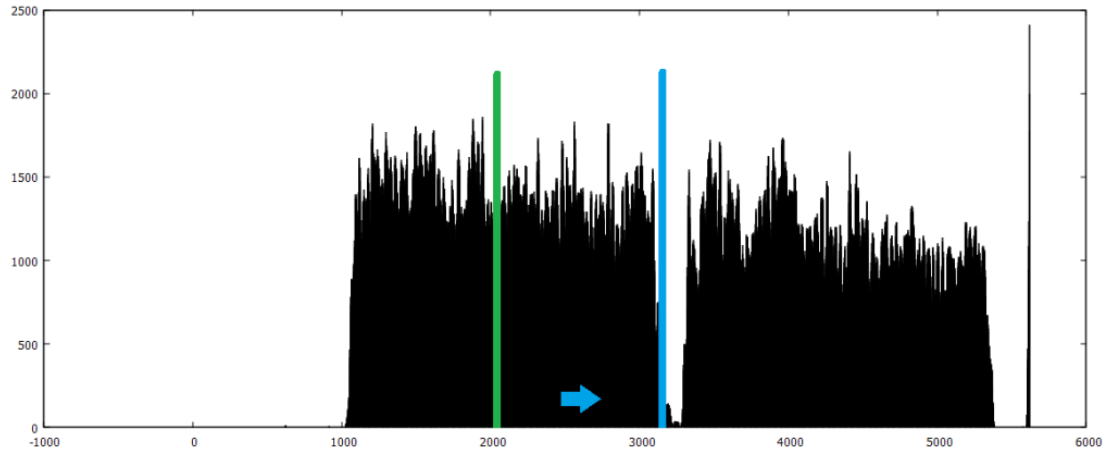
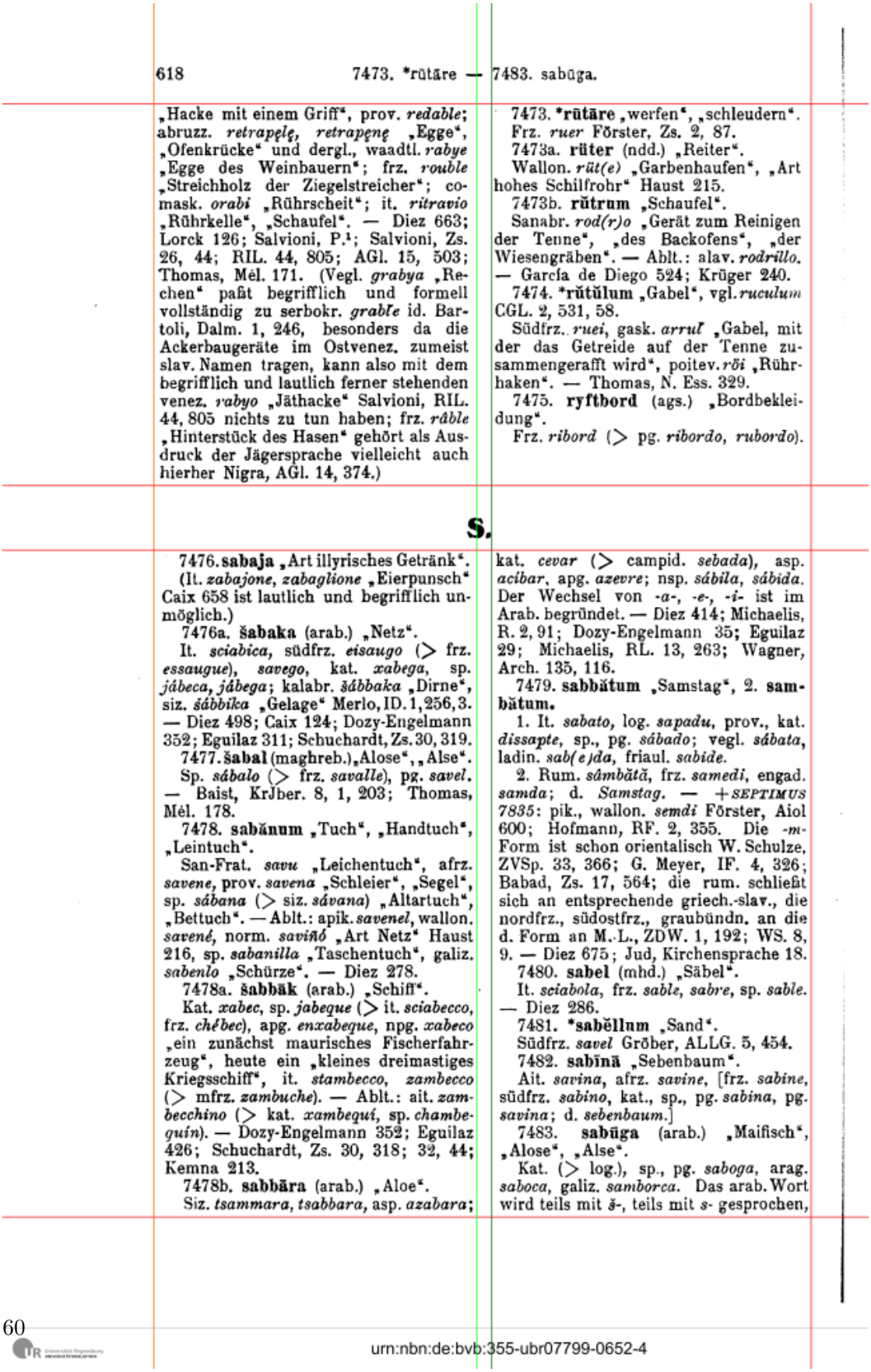


Abbildung 4.3: Auffinden des Beginns der Mitte zwischen den beiden Textblöcken

Umgekehrt kann das Ende des Leerraums in der Mitte (also der Beginn der rechten Textspalte) gefunden werden, indem man im rechten Block beginnt und sich nach links bewegt, bis der Schwellwert unterschritten wird. Von dort ausgehend können der linke und der rechte Rand der Textspalten auf analoge Weise erkannt werden. Auch der obere und unterer Rand kann mit einem entsprechenden Histogramm der Bildzeilen auf erkannt werden, beim oberen Rand muss lediglich der erste kleinere Block der Kopfzeile übersprungen werden. Für diejenigen Seiten, die den Beginn eines neuen Buchstabens enthalten, wird zusätzlich das Vorhandensein einer entsprechenden Lücke untersucht und falls anwendbar deren Ränder markiert:

[5] Diese entstehen durch eine einfache Zählung schwarzer Pixel einer Spalte bzw. Zeile in einer rein schwarz-weißen Darstellung des Scans



618 7473. *rūtāre — 7483. sabūga.

„Hacke mit einem Griff“, prov. *redable*; abruzz. *retrapęlę, retrapęnę* „Egge“, „Ofenkrücke“ und dergl., waadtl. *rabye* „Egge des Weinbauern“; frz. *rouble* „Streichholz der Ziegelstreicher“; comask. *orabi* „Rührscheit“; it. *ritravio* „Rührkelle“, „Schaufel“. — Diez 663; Lorck 126; Salvioni, P.[1]; Salvioni, Zs. 26, 44; RIL. 44, 805; AGl. 15, 503; Thomas, Mél. 171. (Vegl. *grabya* „Rechen“ paßt begrifflich und formell vollständig zu serbokr. *grablje* id. Bartoli, Dalm. 1, 246, besonders da die Ackerbaugeräte im Ostvenez. zumeist slav. Namen tragen, kann also mit dem begrifflich und lautlich ferner stehenden venez. *rabyo* „Jäthacke“ Salvioni, RIL. 44, 805 nichts zu tun haben; frz. *râble* „Hinterstück des Hasen“ gehört als Ausdruck der Jägersprache vielleicht auch hierher Nigra, AGl. 14, 374.)

7473. ***rūtāre*** „werfen“, „schleudern“. Frz. *ruer* Förster, Zs. 2, 87.

7473a. **rüter** (ndd.) „Reiter“. Wallon. *rüt(e)* „Garbenhaufen“, „Art hohes Schilfrohr“ Haust 215.

7473b. **rŭtrum** „Schaufel“. Sanabr. *rod(r)o* „Gerät zum Reinigen der Tenne“, „des Backofens“, „der Wiesengräben“. — Ablt.: alav. *rodrillo*. — García de Diego 524; Krüger 240.

7474. ***rŭtŭlum*** „Gabel“, vgl. *ruculum* CGL. 2, 531, 58. Südfrz. *ruei*, gask. *arrul* „Gabel, mit der das Getreide auf der Tenne zusammengerafft wird“, poitev. *rŏi* „Rührhaken“. — Thomas, N. Ess. 329.

7475. **ryftbord** (ags.) „Bordbekleidung“. Frz. *ribord* (> pg. *ribordo, rubordo*).

S.

7476. **sabaja** „Art illyrisches Getränk“. (It. *zabajone, zabaglione* „Eierpunsch“ Caix 658 ist lautlich und begrifflich unmöglich.)

7476a. **šabaka** (arab.) „Netz“. It. *sciabica*, südfrz. *eisaugo* (> frz. *essaugue*), *savego*, kat. *xabega*, sp. *jábeca, jábega*; kalabr. *šábbaka* „Dirne“, siz. *šábbika* „Gelage“ Merlo, ID. 1, 256, 3. — Diez 498; Caix 124; Dozy-Engelmann 352; Eguilaz 311; Schuchardt, Zs. 30, 319.

7477. **šabal** (maghreb.) „Alose“, „Alse“. Sp. *sábalo* (> frz. *savalle*), pg. *savel*. — Baist, KrJber. 8, 1, 203; Thomas, Mél. 178.

7478. **sabănum** „Tuch“, „Handtuch“, „Leintuch“. San-Frat. *savu* „Leichentuch“, afrz. *savene*, prov. *savena* „Schleier“, „Segel“, sp. *sábana* (> siz. *sávana*) „Altartuch“, „Bettuch“. — Ablt.: apik. *savenel*, wallon. *savené*, norm. *saviñó* „Art Netz“ Haust 216, sp. *sabanilla* „Taschentuch“, galiz. *sabenlo* „Schürze“. — Diez 278.

7478a. **šabbāk** (arab.) „Schiff“. Kat. *xabec*, sp. *jabeque* (> it. *sciabecco*, frz. *chébec*), apg. *enxabeque*, npg. *xabeco* „ein zunächst maurisches Fischerfahrzeug“, heute ein „kleines dreimastiges Kriegsschiff“, it. *stambecco, zambecco* (> mfrz. *zambuche*). — Ablt.: ait. *zambecchino* (> kat. *xambequí*, sp. *chambequín*). — Dozy-Engelmann 352; Eguilaz 426; Schuchardt, Zs. 30, 318; 32, 44; Kemna 213.

7478b. **sabbāra** (arab.) „Aloe“. Siz. *tsammara, tsabbara*, asp. *azabara*; kat. *cevar* (> campid. *sebada*), asp. *acíbar*, apg. *azevre*; nsp. *sábila, sábida*. Der Wechsel von *-a-, -e-, -i-* ist im Arab. begründet. — Diez 414; Michaelis, R. 2, 91; Dozy-Engelmann 35; Eguilaz 29; Michaelis, RL. 13, 263; Wagner, Arch. 135, 116.

7479. **sabbătum** „Samstag“, 2. **sambătum.** 1. It. *sabato*, log. *sapadu*, prov., kat. *dissapte*, sp., pg. *sábado*; vegl. *sábata*, ladin. *sab(e)da*, friaul. *sabide*.

2. Rum. *sâmbătă*, frz. *samedi*, engad. *samda*; d. *Samstag*. — +SEPTIMUS *7835*: pik., wallon. *semdi* Förster, Aiol 600; Hofmann, RF. 2, 355. Die *-m-* Form ist schon orientalisch W. Schulze, ZVSp. 33, 366; G. Meyer, IF. 4, 326; Babad, Zs. 17, 564; die rum. schließt sich an entsprechende griech.-slav., die nordfrz., südostfrz., graubündn. an die d. Form an M.-L., ZDW. 1, 192; WS. 8, 9. — Diez 675; Jud, Kirchensprache 18.

7480. **sabel** (mhd.) „Säbel“. It. *sciabola*, frz. *sable, sabre*, sp. *sable*. — Diez 286.

7481. ***sabĕllum*** „Sand“. Südfrz. *savel* Gröber, ALLG. 5, 454.

7482. **sabīnā** „Sebenbaum“. Ait. *savina*, afrz. *savine*, [frz. *sabine*, südfrz. *sabino*, kat., sp., pg. *sabina*, pg. *savina*; d. *sebenbaum*.]

7483. **sabūga** (arab.) „Maifisch“, „Alose“, „Alse“. Kat. (> log.), sp., pg. *saboga*, arag. *saboca*, galiz. *samborca*. Das arab. Wort wird teils mit *š-*, teils mit *s-* gesprochen,

Abbildung 4.4: Erkannte Spaltengrenzen für Seite 618 aus dem REW

Dieses einfache Vorgehen führte zur korrekten Erkennung von 818 der 821 Seiten aus dem Hauptteil des REW, die restlichen drei Seiten wurden manuell nachbearbeitet. Die entstanden Seitensegmente wurden von links nach rechts und von oben nach unten nummeriert. Jedes Segment wird im folgenden über die Seitenzahl[6] und die Nummer des Segments referenziert.

4.2 Texterkennung

Auf Basis der im vorherigen Kapitel erstellten Seitensegmente wird nun die eigentliche Texterkennung durchgeführt. Prinzipiell war in den verwendeten Scans bereits Text eingebettet, der mit einem OCR-System erzeugt wurde. Dieser wurde jedoch offensichtlich auf Basis eines Zeichensatzes erzeugt, der nur das deutsche Alphabet verwendet. Dies ist bei einem Wörterbuch, das eine Vielzahl spezieller Buchstaben und Diakritika enthält (vgl. Abb. 4.5) so nicht zu gebrauchen. Auch Versuche mit den kombinierten Zeichensätzen verschiedener romanischer Sprachen führten nicht zu guten Ergebnisse (vgl. die Tabelle am Ende des Kapitels), da die häufig vorkommenden phonetischen Notationen nicht bzw. schlecht erkannt wurden.

[6] Mit Seitenzahlen sind hier (und auch in den weiteren) grundsätzlich die aufsteigend nummerierten Seiten des PDF-Scans gemeint, der Grundlage der OCR-Erkennung war. Sie entsprechen damit nicht den in der Textvorlage verwendeten Seitenzahlen. Dies hat technische Gründe, da so auch die mit römischen Ziffern nummerierten Seiten mit Vorwort und Abkürzungen bzw. nicht nummerierte Seiten konsistent referenziert werden können.

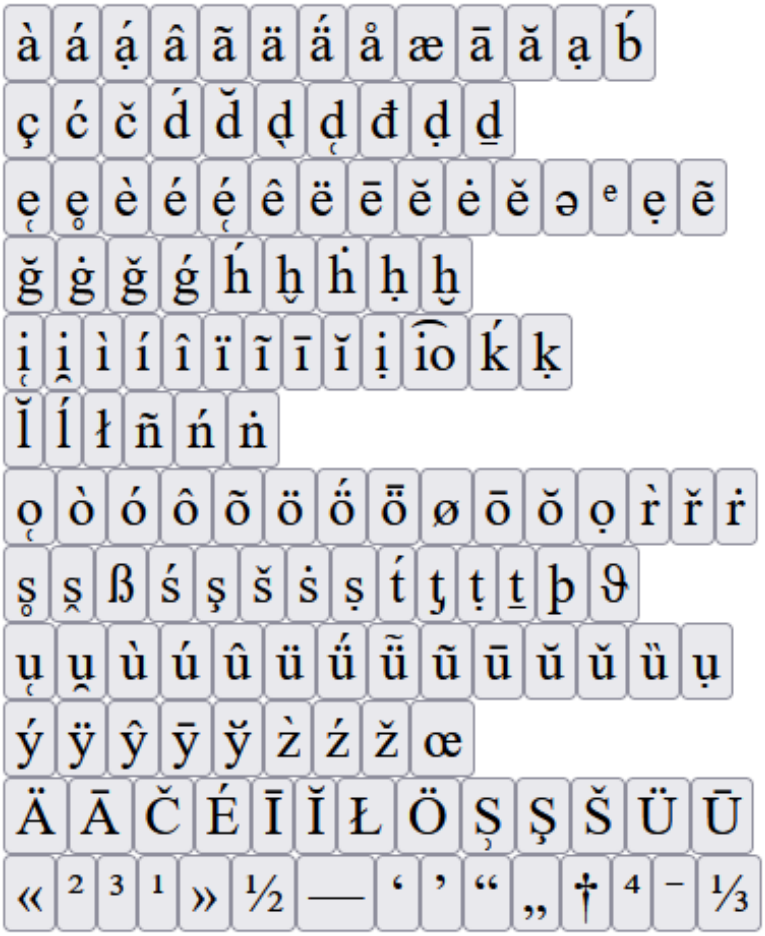


Abbildung 4.5: Alle Zeichen aus dem REW (Stand 30.05.2022), die nicht im ASCII-Standard[a] enthalten sind

[a] https://de.wikipedia.org/wiki/American_Standard_Code_for_Information_Interchange

Aus diesem Grund (und um eine Erkennung der Formatierung der Zeichen zu ermöglichen) wurde eine vollständiges Neutrainierung auf Basis von Ausschnitten aus dem REW durchgeführt. Dazu wurden für insgesamt 27 vollständige Seitensegmente und 8 weitere Einträge (insgesamt 1688 Zeilen Text) aus verschiedenen Teilen des REW mit *tesseract* sogenannte *Box Files* erzeugt und diese Ergebnisse manuell nachkorrigiert. Die *Box Files* markieren für jeden Buchstaben der Seite dessen (rechteckige) Begrenzung im Bild und (optional) seine Formatierung. Zur Verarbeitung der *Box Files* wurde die freie Software QT Box Editor verwendet.

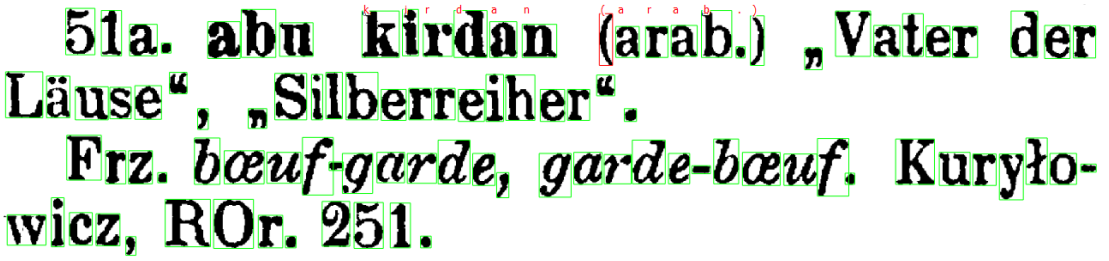


Abbildung 4.6: *Box File* für einen kurzen Eintrag aus dem REW in der Ansicht des *QT Box Editor*s

Für die Formatierung der Zeichen erlaubt *tesseract* (bis Version 3.x) verschiedene Formatangaben (italic, bold, fixed, serif, fraktur). Diese werden so verwendet, dass Zeichen in den Trainingsdaten verschiedenen Schriftarten zugeordnet werden und deren Formatierung entsprechend angegeben wird. Grundsätzlich sind dafür unterschiedliche *Box Files* für jede Schriftart nötig, im *QT Box Editor* ist es allerdings möglich die Formatierungen Kursivierung, Fettdruck und Unterstreichung innerhalb einer Datei anzugeben und diese nachträglich in die verschiedene Teile aufzuspalten, was den Arbeitsablauf deutlich vereinfacht. Da im REW Etyma zum Teil in Kapitälchen gesetzt werden, was für die Texterkennung nicht vorgesehen ist, wurden diese stattdessen als unterstrichen markiert. Trotz dieser Zuordnung (und Versuchen mit anderen Formatangaben) wurden diese nicht korrekt erkannt und vom OCR-System konsequent als normal formatierte Großbuchstaben interpretiert. Somit wurde dies in einem zusätzlichen Post-Processing-Schritt angeglichen (vgl. Kap. 4.3.3).

Um diese Trainingsdaten weiter zu korrigieren, wurden pro Zeichen und Formatierung außerdem einzelne Bilddateien generiert, die alle Vorkommen hintereinander setzen (vgl. Abb. 4.7. Diese erleichtern die Korrektur, da eventuell falsch markierte Zeichen im direkten Vergleich oftmals leichter erkannt werden können.

Abbildung 4.7: Sammlung aller Vorkommen für die (unformatierte) Ziffer 0, die fälschlich eine kursive Variante enthält

Der gesamte Prozess wurde dabei entsprechend der allgemeinen Prinzipien aus Kap. 3.1 in Teilschritten durchgeführt. Im erste Schritt wurden die Verzeichnisse zu Beginn des Wörterbuchs erfasst[7] und intensiv manuell nachkorrigiert. Die dort enthaltenen Abkürzungen wurden dann (zusammen mit einer regulären deutschen Wortliste) dem OCR-System zur Verfügung gestellt, sodass es diese bei unsicheren Fällen als Hinweis verwenden kann[8]. Im weiteren wurden nach und nach einzelne Abschnitte importiert und manuell nachkorrigiert. Insbesondere wurde auch eine Liste aller speziellen Zeichen geführt, die im Text vorkamen. Diese konnte einerseits für die

[7] Aufgrund der verhältnismäßig geringen Anzahl von speziellen Zeichen mit Standardparametern der Texterkennung

[8] Dies führte allerdings zu keiner deutlichen Erhöhung der Qualität der Ergebnisse. Somit wurde ein zusätzlicher Verbesserungsschritt ergänzt (vgl. Kap. 4.3.4).

Korrektur selbst verwendet werden (vgl. Kap. 12.5), diente aber auch zum Auffinden von Zeichen, die nicht oder nur sehr selten in den Trainingsdaten vorkamen. Somit konnte gezielt auf Basis kurzer Abschnitte, die solche Zeichen enthielten nachtrainiert werden, um die Genauigkeit zu verbessern (vgl. Abb. 4.8). Auch für die selten vorkommenden Fomatierungstypen Fettdruck und Kapitälchen wurden spezielle Bilder zusammengestellt, die zu großen Teilen aus Text mit dieser Formatierung bestanden. Trotzdem bleiben sehr selten auftretende Zeichen wie beispielsweise kursive Großbuchstaben[9] oder nicht lateinische Zeichen in den (fettgedruckten) Lemma-Formen problematisch. Den vollständigen Zeichensatz in allen vorkommenden Formatierungen in den Trainingsdaten abzubilden ist somit nicht praktikabel und fehlerhafte Erkennungen aus diesem Grund weiter möglich.

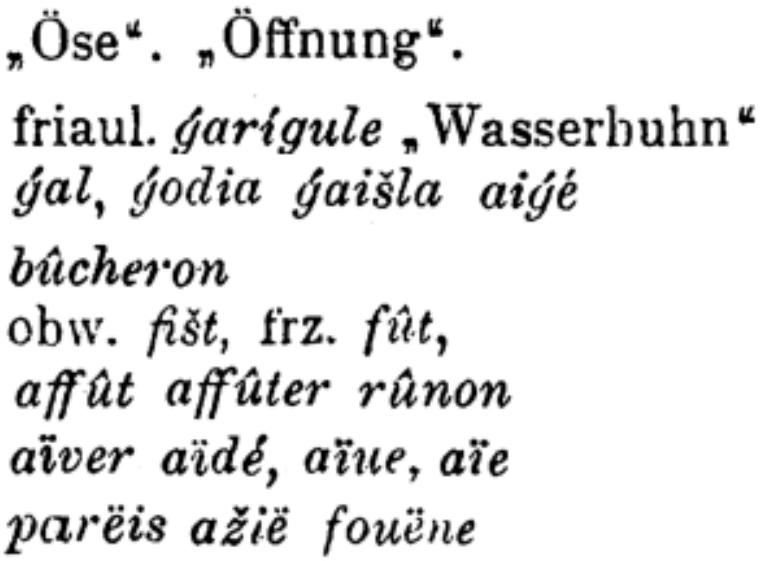


Abbildung 4.8: Ausschnitt aus einem speziellen Bild, dass aus Abschnitten mit selten vorkommenden Zeichen besteht

Die folgende Tabelle zeigt die Anzahl der Fehler[10] für die im Ursprungsscan eingebetteten textuellen Daten, eine Erkennung mit *tesseract* mit deutschen Trainingsdaten, die mit den großen romanischen Sprachen des REW kombiniert wurden, und der endgültigen Fassung der eigens aus dem REW erstellten Trainingsdaten für drei zufällig ausgewählte Seitensegmente[11]:

[9] Diese kommen vereinzelt vor allem in Ortsnamen vor

[10] Als Abstandsmaß wird hier die sogenannte Levenshtein-Distanz verwendet, die die minimale Anzahl an Einfügungen, Löschungen und Ersetzungen auf Zeichenebene zählt, um aus einer Zeichenkette eine andere zu erzeugen (vgl. auch Kap. 4.3.3). Die Formatierung wird nicht berücksichtigt, da sie nur mit den neu erzeugten Trainingsdaten erkannt wird.

[11] Alle Methoden, die im nächsten Kapitel vorgestellt werden, werden ebenfalls anhand dieser drei Beispiele verifiziert und ausgewertet.

Seitensegment	**Bestehendes OCR-Ergebnis**	***tesseract* in den Sprachen deutsch, italienisch, französisch, spanisch, portugiesisch und rumänisch**	**Training mit REW-Daten**
0141_2	159	96	26
0164_1	143	106	36
0823_2	172	221	81

Tabelle 4.1: Anzahl der Fehler für die verschieden OCR-Ergebnisse

Wie man sieht, kann eine deutliche Verbesserung der Ergebnisse erreicht werden, auch wenn die Fehlerquote weiterhin verhältnismäßig hoch ist. Dies hängt zum einen mit der grundsätzlich schwierigen Vorlage zusammen, die aus sehr schmalen Zeilen im Blocksatz mit einer sehr hohen Anzahl unterschiedlicher Zeichen besteht, aber auch mit teils schlechtem Druck, der bestimmte Abschnitte der Quelle besonders schwer erkennbar macht. Somit werden im nächsten Abschnitt Möglichkeiten besprochen, um die Fehlerquote weiter zu reduzieren.

Als Ausgabeformat des OCR-Prozesses wird das hOCR-Format[12] verwendet. Dieses basiert auf XML und gibt die Ergebnisse im Gegensatz zur reinen Textausgabe sehr detailliert wieder. Zur Veranschaulichung dient der folgende Ausschnitt:

```
  <div class='ocr_page' id='page_1' title='image "blocks_cut/content/p0345_1.png"; bbox 0 0 21
   <div class='ocr_carea' id='block_1_1' title="bbox 21 20 2133 7428">
    <p class='ocr_par' id='par_1_1' lang='rew' title="bbox 21 20 2064 373">
     <span class='ocr_line' id='line_1_1' title="bbox 21 20 2063 136; baseline -0 -23; x_size
     </span>
     <span class='ocr_line' id='line_1_2' title="bbox 21 140 2064 256; baseline -0.001 -24; x_
-</span> <span class='ocrx_word' id='word_1_9' title='bbox 684 140 971 230; x_wconf 85'>Ablt.:
...
```

Zusätzlich zum reinen Text enthält es alle Strukturelemente zusammen mit deren Position im Ausgangsbild und die Formatierung der Tokens unter Verwendung der Tags und . Außerdem ist für jedes Token ein Konfidenzwert durch *x_wconf* vorhanden, der angibt wie sicher das Texterkennungssystem in diesem Fall war[13]. Außer den Tags für die Zeichenformatierung sind hier vor allem die Pixelkoordinaten der einzelnen Zeilen entscheidend, die mit in die Datenbank importiert werden und später an verschiedenen Stellen im Webportal verwendet werden (vgl. Kap. 12.5, Kap. 12.2).

Der rein textuelle Inhalt wird aus diesem Format zeilenweise zusammengefügt. Die Formatierungen werden dabei in <i> und <b> Tags umgewandelt. Dies hat vor allem den Vorteil, dass diese kürzer sind und die einzelnen Zeilen für die manuelle Korrektur

[12] https://github.com/kba/hocr-spec

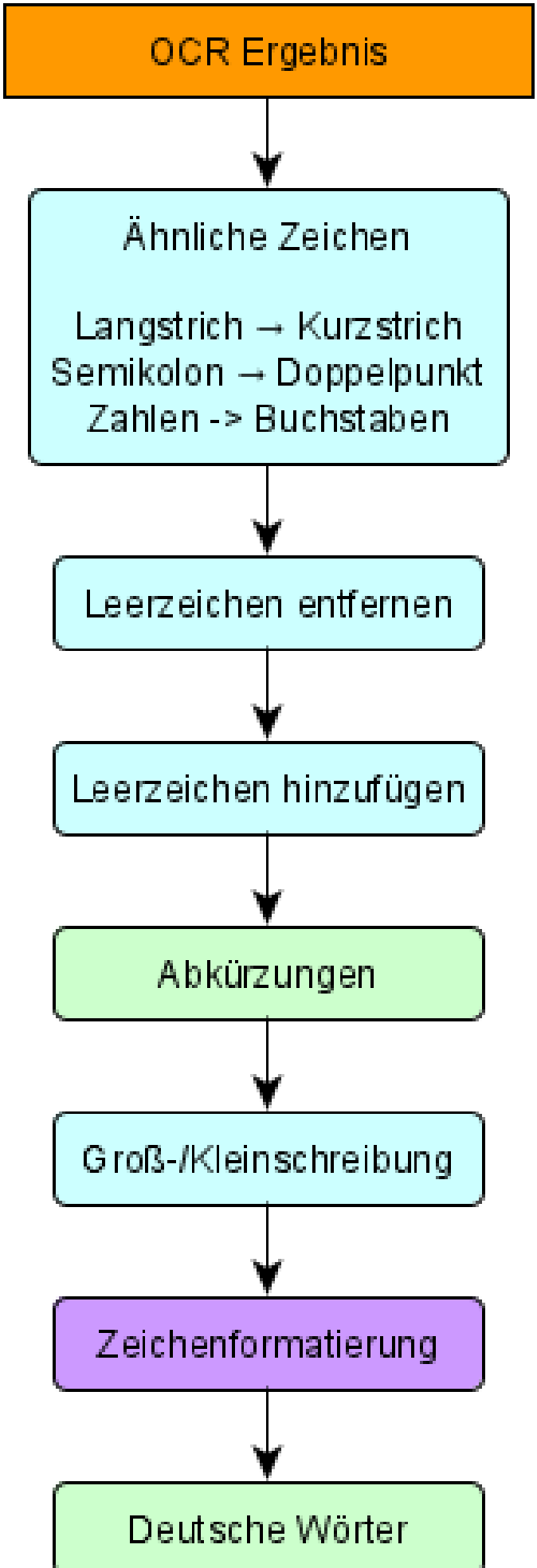
[13] Dieser könnte Basis für bestimmte Korrekturvorgänge sein, wird aber aktuell nicht verwendet.

lesbarer bleiben. Für die Tokens in Kapitälchen wird das HTML-Tag <u> verwendet, der eigentlich zur Unterstreichung dient. Dies hat den Grund, dass Kapitälchen in HTML nur über besondere CSS-Regeln dargestellt werden können. Um wiederum die Lesbarkeit zu erhöhen wird also das einzige weitere einbuchstabige Format-Tag verwendet. An der Oberfläche des Webportals werden diese Tags entsprechend als Kapitälchen dargestellt, in den Export-Formaten durch passende andere Möglichkeiten ersetzt (vgl. Kapitel 13). Wenn die Texterkennung einen Absatz (Klasse *ocr_par*) erkannt hat, wird an dieser Stelle eine Leerzeile eingefügt, die den Zeilenumbruch symbolisiert (vgl. auch Kap. 5.1.2). Somit kann eine möglichst quellentreue Darstellung des Ursprungstext erreicht werden, außerdem sind die Absätze zum Teil wichtig für die strukturelle Erkennung nummerierter Unterabschnitte (vgl. Kap. 5.3.2).

4.3 Post-Processing

Ein gewisser Anteil von Fehlern im OCR-Ergebnis ist sehr systematischer Natur. Eine einfache Art um solche zu korrigieren, können Ersetzungsregeln auf Basis von regulären Ausdrücken sein. Hiermit können häufig auftretende Probleme behoben werden wie beispielsweise die Umwandlung der Ziffer *1* innerhalb eines Wortes in den Buchstaben *l*. Für die ersten Textimporte wurde eine Reihe solcher Regeln aufgestellt[14], die Erfahrungen aus den manuell korrigierten bereits vorhandenen Abschnitten widerspiegeln. Dies kann in der Anfangsphase hilfreich sein, hat aber auch seine Limitierungen. So erfordert es oftmals eine Vielzahl sehr spezialisierter Regeln, um möglichst zu verhindern, dass eine zu allgemeine Regel wiederum zu neu eingeführten Fehlern führt. Deren Fomulierung und Anpassung ist arbeitsaufwendig und kann trotzdem immer nur unsystematisch Problemfälle behandeln, die als häufig wahrgenommen werden . Sobald eine gewisse Menge[15] an nachkorrigierten Abschnitten zur Verfügung steht, bietet es sich somit an auf Basis dieser Daten mit Hilfe u.a. von Verfahren des maschinellen Lernens bessere Korrekturroutinen zu erstellen. Deren genaue Ausgestaltung hängt vom Quellentext und den Anforderungen für die weitere Verarbeitung ab. Im Falle des REW ist vor allem strukturelle Integrität wichtig, d.h. besonders Leerzeichen, Satzzeichen und ähnliches sollten möglichst exakt erfasst werden. Inhaltliche Fehler werden deshalb nur auf Basis bestehender Abkürzungen und der deutschen natürlichsprachigen Abschnitte vorgenommen, die Korrektur von sprachlichen Formen ist hier kaum möglich und kann besser anhand der bereits verarbeiteten Daten vorgenommen werden (vgl. Kap. 8.2.2). Insgesamt wurde die folgende Prozesskette konzipiert:

[14] Der entsprechende Programmcode mit allen Ersetzungsregeln findet sich hier
[15] Im vorliegenden Fall lag dieser Wert bei 19,3% der Seitensegmente des REW.


OCR Ergebnis
Ähnliche Zeichen
Langstrich → Kurzstrich
Semikolon → Doppelpunkt
Zahlen -> Buchstaben
Leerzeichen entfernen
Leerzeichen hinzufügen
Abkürzungen
Groß-/Kleinschreibung
Zeichenformatierung
Deutsche Wörter

Die expliziten Ersetzungsregeln weichen hier abstrakteren Problembeschreibungen, d.h. anstatt beispielsweise Regeln aufzustellen an welchen Stellen Leerzeichen eingefügt oder entfernt werden müssen, wird hier nur das Problem als solches definiert. An welchen Stellen eine Einfügung oder Entfernung sinnvoll ist, wird aus den bestehenden Daten hergeleitet. Die einzelnen Prozessschritte können grob in drei Kategorien eingeteilt werden:

- Einfügungen, Löschungen und Ersetzungen einzelner Zeichen (hellblau)
- Korrektur der Formatierung (lila)
- Inhaltliche Korrektur auf Tokenebene (grün)

Die Reihenfolge der einzelnen Schritte ist hier in den meisten Fällen entscheidend, so sollten beispielsweise die Leerzeichen behandelt werden, bevor die Abkürzungen korrigiert werden, da im OCR-Ergebnis zum Teil Leerzeichen zwischen der eigentlichen Abkürzung und derem abschließenden Punkt sind. Die Abkürzungen sollten wiederum vor der Untersuchung von Stellen, an denen Kommata durch Punkte ersetzt werden, behandelt werden, da sonst in vielen Fällen der Punkt einer falsch erkannten Abkürzung durch ein Komma ersetzt wird.

Die einzelnen Teilschritte basieren auf zwei Modellen für das maschinelle Lernen, *Support Vector Machines* und *Conditional Random Fields*[16]. Diese werden in Kap. 4.3.1 kurz eingeführt. Die Details zu den einzelnen Problemstellungen werden darauf aufbauend in Kap. 4.3.2, Kap. 4.3.3 und Kap. 4.3.4 behandelt. Die entsprechenden Implementierungen in der Programmiersprache *Python* finden sich hier. Zusammen führen sie zu folgender Verbesserung der Ergebnisse[17]:

[16] Eine Ausnahme bildet die inhaltliche Korrektur, die nur auf Basis von Wortlisten arbeitet

[17] Die Formatierung der Zeichen wird hier wiederum nicht betrachtet, die Werte hierzu finden sich in Kap. 4.3.3

Seitensegment	**Bestehendes OCR-Ergebnis**	***tesseract* in den Sprachen deutsch, italienisch, französisch, spanisch, portugiesisch und rumänisch**	**Training mit REW-Daten**	**Regelbasiertes Post-Processing**	**Post-Processing auf Basis der korrigierten Daten**
0141_2	159	96	26	27	12
0164_1	143	106	36	29	16
0823_2	172	221	81	58	27

Tabelle 4.2: Anzahl der Fehler für die verschieden OCR-Ergebnisse inklusive der Post-Processing-Ergebnisse

Die Anzahl der Fehler konnte somit in allen Beispielen ungefähr halbiert werden, während die expliziten Ersetzungsregeln nur zu moderaten Verbesserungen führten[18].

Alle Korrekturen werden nicht direkt auf die Textzeilen angewendet, sondern zusammen mit den originalen Zeilen als „Korrekturdatensätze“ (vgl. Kap. 8.1) in die Datenbank importiert. Diese könnten somit grundsätzlich (nach einer manuellen Nachkorrektur) ebenfalls für das Training und somit eine höhere Genauigkeit bei späteren Importen verwendet werden. Richtige Korrekturen werden dabei gewissermaßen validiert, während falsche Korrekturen bei der manuellen Nachbearbeitung überschrieben werden und somit nicht mehr vorkommen.

4.3.1 Auswahl von passenden Machine Learning Algorithmen

Alle Korrekturschritte (mit Ausnahme der in Kap. 4.3.4 beschriebenen inhaltlichen Korrekturen) können als Klassifikationsprobleme aufgefasst werden. Hierbei wird einem bestimmten Datenpunkt je nach dessen Eigenschaften eine von mehreren Klassen zugewiesen. Im vorliegenden Fall sind das diese beiden Varianten:

[18] Der Grund, warum bei Seitensegment 141_2 das regelbasierte Post-Processing die Fehleranzahl sogar um eins erhöht ist, dass in diesem Fall an mehren Stellen ein kleines *o* innerhalb eines Wortes als Großbuchstabe erkannt wurde, wodurch vor diesem ein zusätzliches Leerzeichen eingefügt wurde. Aufgrund der verhältnismäßig geringen sonstigen Anzahl an Fehlern wirkt sich das deutlich aus.

- Soll in einem bestimmten Kontext eine Änderung durchgeführt werden (Klasse 1) oder nicht (Klasse 0)? Es handelt sich als um ein binäres Klassifikationsproblem.
- Welche Formatierung soll einem bestimmten Token zugewiesen werden? Hier gibt es vier verschiedene Klassen, die den Arten der Formatierung (regulär, kursiv, fett und Kapitälchen) entsprechen.

Da die zugrundeliegenden Modelle meist mathematischer Natur sind, werden als Eingabedaten im Normalfall Vektoren von Zahlen benötigt, d.h. nicht-numerische Daten müssen je nach Anwendungsfall in geeigneter Weise kodiert werden (vgl. z.B. Aggarwal 2018, S. 4–15). Ein gewisser Nachteil vieler Standardlösungen für Klassifikationsprobleme im Bereich von Texten ist, dass die Lösung kontextunabhängig ist, d.h. die Klassenzuweisung eines Datenpunkts hängt nicht von den Klassen der anderen Datenpunkte ab. Für die Probleme der ersten Variante ist es in den meisten Fällen kein Problem, da beispielsweise die Entscheidung ob an einer bestimmten Position ein Leerzeichen entfernt werden soll unabhängig davon ist, ob an anderer Stelle eines entfernt wurde. Es gibt im konkreten Fall allerdings eine Ausnahme bei der Korrektur der Groß- und Kleinschreibung. *tesseract* tendiert oftmals dazu in Fällen, bei denen sich große und kleine Varianten eines Buchstabens sehr ähnlich sehen, anstelle des kleinen einen Großbuchstaben zu erkennen[19]. Bei der Erkennung, ob ein Großbuchstabe durch die kleingeschriebene Variante ersetzt werden soll, kann es allerdings durchaus eine Rolle spielen, wie die umgebenden Buchstaben behandelt werden[20]. Weitaus wichtiger ist in dieser Hinsicht allerdings die zweite Variante von Problem, die sich mit dem Zeichenformat beschäftigt. In diesem Fall ist die Formatierung der Umgebung in vielen Fällen entscheidend für die Formatierung eines bestimmten Tokens. Somit fallen viele Standard-Modelle für das maschinelle Lernen weg.

Unter diesen Voraussetzungen wurden zwei verschiedene Klassifikatoren einmal für kontextunabhängige und einmal für kontextsensitive Probleme ausgewählt. Für erstere wurden sogenannte *Support Vector Machines* verwendet, ein weit verbreitetes Modell, das die einzelnen Datenpunkte mit Hilfe von Hyperebenen separiert, um so die Klassengrenzen darzustellen (für mathematische und algorithmische Details vgl. z.B. Steinwart und Christmann 2008). Da die meisten der Anwendungsfälle eher einfache und niedrigdimensionale Probleme sind (vgl. Kap. 4.3.3), die die Möglichkeiten des maschinellen Lernens nicht voll ausschöpfen, ist hier die genaue Auswahl der Methode (und deren Parametrisierung) allerdings weniger entscheidend.

Als zweites Konstrukt wurden sogenannte *Conditional Random Fields (CRFs)* (vgl. Lafferty, McCallum und Pereira 2001) verwendet. Bei diesen wird nicht einem Datenpunkt eine Klasse zugewiesen, sondern einer Liste von Datenpunkten entsprechend eine Liste von Klassen, wobei die Zuordnungen jeweils voneinander

[19] Der umgekehrte Fall kam in der Praxis nicht vor und wurde deshalb nicht behandelt.

[20] Insbesondere gilt dies natürlich für Tokens in Kapitälchen, die vollständig als Großbuchstaben interpretiert werden (vgl. Kap. 4.2) und somit komplett in Kleinbuchstaben umgewandelt werden sollten.

abhängig ist. Obwohl das Modell aus mathematischer Sicht sehr variabel ist, wird es in der Praxis in vielen Fällen auf sogenannte *Linear Chain CRFs* beschränkt, die nur den direkten Vorgänger eines Elements berücksichtigen. Somit spielt im Gesamtsystem jeweils nur der direkte Kontext eine Rolle. Für die einzelnen Elemente werden die Merkmale über *observation functions* dargestellt, die in realen Anwendungsfälle meist zu simplen booleschen Funktion vereinfacht werden, die nur wahr oder falsch zurückgeben (vgl. z.B. Sutton und McCallum 2010, S. 24–26). Somit lassen sich aus informatischer Sicht die Merkmale für ein bestimmtes Element leicht über eine Menge von *Labels* darstellen, wobei die Zuweisung eines solchen *Labels* einer booleschen Funktion entspricht, die für dieses Element wahr ist. Eine direkte Kodierung von textuellen Elementen als Zahlen ist somit nicht nötig, stattdessen werden die für den Anwendungsfall relevanten *Labels* erzeugt. In der verwendeten Bibliothek *sklearn-crfsuite*[21] können die Merkmale beispielsweise über Schlüssel-Wert Paare angelegt werden. Für ein einzelnes Token könnte ein sehr einfaches Beispiel diese Form haben:

```
{
   "is_lower": 1,
   "is_number": 0
}
```

In diesem Fall wären die einzigen beiden Merkmale, ob ein Token kleingeschrieben ist und ob es eine Zahl ist. Daraus würden entsprechend zwei *Labels* der Form „is_lower=1“ und „is_number=0“ erzeugt werden. Die weiteren Beispiele in den folgenden Kapiteln werden ebenfalls in der obigen Notation angegeben. Die grundlegenden Aufgabe bei einem *CRF*-basierten Ansatz ist die Festlegung von Merkmalen, die die im gegebenen Kontext relevante Information kodieren. CRFs wurden bereits in unterschiedlichen (computer-)linguistischen Fragestellungen verwendet (vgl. z.B. Ekbal, Haque und Bandyopadhyay 2007, Kudo, Yamamoto und Matsumoto 2004, Lüschow 2020).

4.3.2 Formatierung

Der wahrscheinlich wichtigste Fall, um die späteren Verarbeitungsresultate zur verbessern, ist die korrekte Erkennung der Formatierung der einzelnen Tokens, da vor allem die Kursivierung der sprachlichen Formen einen hohen Grad an Information transportiert. Deren Erkennung durch *tesseract* anhand der Trainingsdaten wie sie in Kap. 4.2 vorgestellt wurden, ist allerdings sehr schlecht[22]:

[21] https://sklearn-crfsuite.readthedocs.io/en/latest/

[22] Wörter in Kapitälchen werden hier nicht untersucht, da sie von der Texterkennung grundsätzlich nicht erkannt wurden (vgl. Kap. 4.2)

Seitensegment	Keine Formatierung	Kursiv	Fett
0141_2	1,9%	-30,1%	250,1%
0164_1	4,0%	-21,0%	-12,0%
0823_2	6,8%	-19,7%	-16,5%

Tabelle 4.3: Abweichungen der jeweiligen Formatierung im OCR-Ergebnis

Die Prozentwerte in der Tabelle geben dabei den relativen Anteil der Zeichen an, die falsch der jeweiligen Formatierung zugewiesen wurden:

$$\left(\frac{\text{Anzahl mit Fomatierung Ergebnis}}{\text{Gesamte Zeichenzahl Ergebnis}} - \frac{\text{Anzahl mit Fomatierung korrigiert}}{\text{Gesamte Zeichenzahl korrigiert}}\right) \div \frac{\text{Anzahl mit Fomatierung korrigiert}}{\text{Gesamte Zeichenzahl korrigiert}}$$

Abbildung 4.10: Formel für die Berechnung der Prozentwerte bei der Formatierung

Negative Vorzeichen bedeuten also, dass zu wenige Zeichen mit dieser Formatierung erkannt wurden, während positive umgekehrt angeben, dass zu viele Zeichen mit einer solchen erkannt wurden. Im ersten Beispiel wurden somit knapp 2% mehr Zeichen als unformatiert erkannt, als es richtig wäre, während ca. 30% weniger kursiviert sind als in der korrigierten Lösung. Der Grund für diese Angabe der Abweichungen ist, dass ein globaler Fehlerwert schwer zu definieren ist. Der wohl intuitivste Ansatz, nämlich die Zählung aller Tokens, denen die falsche Formatierung zugewiesen wurden, ist in der Praxis schwierig umzusetzen, da insbesondere durch die verhältnismäßig schmalen Seitensegmente, die im Blocksatz formatiert sind, des öfteren zusätzliche Leerzeichen erkannt werden bzw. Leerzeichen nicht erkannt werden (vgl. hierzu die Beispiele in Kap. 6.1). Somit kann sich die Anzahl und die Aufteilung der Tokens zwischen dem vollständig korrigierten Text und einem fehlerbehafteten Text durchaus unterscheiden, was einen direkten Vergleich zwischen Tokens aufwendig macht. Auch können beispielsweise bei einem falsch erkannten Leerzeichen ein Teiltoken als formatiert und das andere als unformatiert erkannt werden, wodurch unklar ist wie ein solcher Fall behandelt werden soll. Die Untersuchung der Formatierung auf Zeichenebene ist somit robuster, hat aber den Nachteil, dass es schwierig ist einen globalen Fehlerwert für einen bestimmten Textabschnitt anzugeben, da einzelne Zeichen der korrigierten Fassung oftmals nicht auf Zeichen in der fehlerhaften Fassung abgebildet werden können[23]. Ein weiterer Vorteil dieser Konvention ist allerdings, dass gerade bei kurzen Tokens, bei denen in den meisten Fällen keine Kursivierung erkannt wird, eine Untersuchung auf Tokenebene eine gewisse Verzerrung aufweisen würde, was so nicht der Fall ist.

Die Daten in der Tabelle zeigen, dass tendenziell zu wenige Zeichen als formatiert erkannt wurden, wobei es zu einem Ausreißer bei den fett formatierten Zeichen kommt.

[23] Ein Nachteil ist auch, dass keinerlei Aussage darüber getroffen wird, ob die korrekten Zeichen formatiert wurden, weil nur der Anteil der Zeichen mit einer gewissen Formatierung betrachtet wird. Somit wäre es theoretisch möglich, dass die Fehlerquote Null ist, obwohl genau die falschen Zeichen einem Format zugeordnet wurde, was allerdings in der Praxis wenig realistisch ist. Für eine grobe Qunatifizierung der Korrektheit sind diese Wert somit durchaus ausreichend.

In diesem Fall wird eine deutlich erhöhte Anzahl an Zeichen als fett markiert. Dieser Effekt tritt auf verschiedenen Seiten auf und hängt mit Stärke des Drucks der ursprünglichen Quelle zusammen. Während in manchen Fällen der Druck so dünn ist, dass einzelne Zeichenbestandteile vollständig fehlen (vgl. Abb. 4.11) und somit die Anzahl von falsch erkannten Zeichen tendenziell höher ist, kommen umgekehrt Seiten vor, auf denen der Druck so dick ist, dass zusätzliche fette markierte Wörter erkannt werden (vgl. Abb. 4.12).

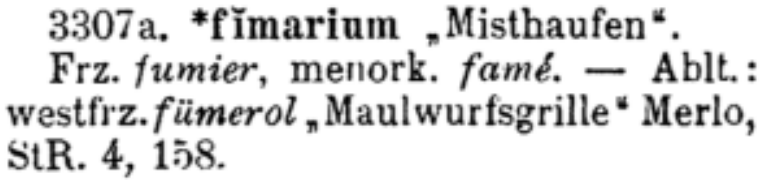

3307a. *fĭmarium „Misthaufen".
Frz. *fumier*, menork. *famé*. — Ablt.:
westfrz. *fümerol* „Maulwurfsgrille" Merlo,
StR. 4, 158.

Abbildung 4.11: Beispiel für Druck mit sehr wenig Tinte aus dem REW

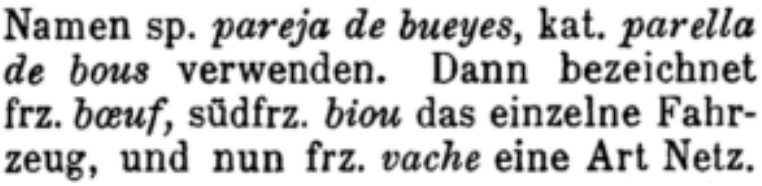

Namen sp. *pareja de bueyes*, kat. *parella de bous* verwenden. Dann bezeichnet frz. *bœuf*, südfrz. *biou* das einzelne Fahrzeug, und nun frz. *vache* eine Art Netz.

Abbildung 4.12: Beispiel für Druck mit sehr viel Tinte (REW, S. 1196). Die Tokens „bezeichnet", „einzelne" und „eine" wurden hier als fett formatiert erkannt.

Für die initialen regelbasierten Ersetzungen wurden insgesamt vier entsprechende Fälle behandelt:

- Kleingeschriebene Tokens nach einer Sprachabkürzung wurden immer kursiviert
- Wenn die vorherige Zeile mit einem Bindestrich innerhalb eines <i>-Tags beendet wurde, wurde das erste Token der nächsten Zeile ebenfalls kursiviert
- Wenn die vorherige Zeile auf eine Sprachabkürzung endete und die aktuelle mit einem Kleinbuchstaben begann, wurde das erste Token kursiviert
- Fett formatierte Tokens innerhalb von Anführungszeichen (also in Bedeutungsangaben) wurden unformatiert dargestellt.

Diese führen in den Beispielen zu folgenden neuen Werten[24]:

[24] Die Werte wurden nach Anwendung aller Ersetzungen berechnet, d.h. es wurden nicht nur solche ausgeführt, die sich auf die Formatierung beziehen. Das führt zu einer Veränderung der Textlänge und damit zu den leichten Abweichungen beim Fettdruck in den Zeilen 2 und 3, in denen sich an den eigentlichen als fett markierten Tokens nicht geändert hat.

Seitensegment	Keine Formatierung	Kursiv	Fett
0141_2	-2,8% (+ 0,9)	-12,4% (- 17,7)	243,3% (- 6,8)
0164_1	1,9% (- 2,1)	-9,4% (- 11,6)	-12,1% (+ 0,1)
0823_2	1,8% (- 5)	-5,5% (- 14,2)	-16,8% (+ 0,3)

Tabelle 4.4: Abweichungen der jeweiligen Formatierung nach den regelbasierten Ersetzungen. In Klammern wird die Änderung des Absolutbetrags in Prozentpunkten angegeben.

Die Tabelle zeigt, dass es gerade im Bereich der Kursivierungen zu deutlichen Verbesserungen kommt, wobei zum Teil auch durchaus falsche Positive eingeführt werden. Dies bezieht sich vor allem auf Wörter, die in natürlichsprachigen Abschnitten auf Sprachabkürzungen folgen und deshalb kursiviert werden:

```
umgedeutet sp., pg. zu <i>lom-</i> →
umgedeutet sp., pg. <i>zu</i> <i>lom-</i>
```

Gerade im Bereich des Fettdrucks, der eigentlich sehr einfachen Regeln folgt (fett gedruckt sind prinzipiell nur Lemmata in den Kopfzeilen der Einträge), stößt der Ansatz allerdings an seine Grenzen. Der Versuch einer Erkennung der Kopfzeilen (vgl. Kap. 5.3.2) ist aus verschiedene Gründen problematisch[25] und soll hier vermieden werden. Stattdessen wurde ein auf *Conditional Random Fields* basierender Ansatz verwendet. Dies bietet sich an, da die Verwendung der Formatierung extrem kontextabhängig ist und in den meisten Fällen anhand von diesem erkannt werden kann. Zu Zweck der Verarbeitung werden die einzelnen Zeilen zuerst tokenisiert[26]. Als Beispiel dient hier der folgende Eintrag:

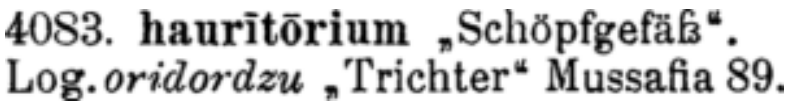

4083. **haurītōrium** „Schöpfgefäß“.
Log. *oridordzu* „Trichter“ Mussafia 89.

Abbildung 4.13: Eintrag 4083 im REW

Die einzelnen Tokens aus diesem Abschnitt können folgendermaßen dargestellt werden:

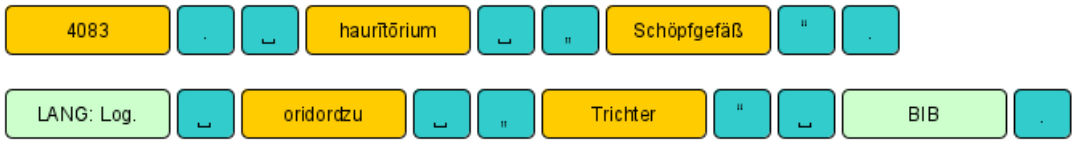


Abbildung 4.14: Schematische Darstellung der entsprechenden Tokenisierung

Dabei werden fünf Klassen von Tokens unterschieden:

[25] Das wäre u.a. deswegen schwierig, weil die Vorausetzung für eine robuste Erkennung der Kopfzeilen eine möglichst korrekte Erkennung der Formatierung ist.

[26] Es handelt sich um keine Tokenisierung im strengen Sinne, da auch Leerzeichen als Tokens behandelt werden.

- **Kontrollzeichen** (Leerzeichen, Punkte, Kommata etc.)
- **Sprachabkürzungen**
- **Literaturverweise**
- **Allgemeine Abkürzungen** (wie „Ablt.“)
- **Wörter** (alles übrige)

Zu beachten ist hierbei, dass nicht nur die bibliographischen Abkürzungen, sondern die vollständige Referenz inklusive Seitenzahlen etc. als ein Token aufgefasst wird, das nicht näher spezifiziert wird, also insbesondere auch nicht von anderen Literaturverweisen unterschieden werden kann. Sprachabkürzungen und allgemeine Abkürzungen hingegen werden einzeln unterschieden. Das hat den Hintergrund, dass diese zum Teil durchaus für die Erkennung der Formatierung relevant sein können. Zum Beispiel folgen auf die Abkürzung für *deutsch* zum Teil Wörter, die mit Großbuchstaben beginnen und trotzdem sprachliche Formen (also kursiv) sind. Einen andereren solchen Fall stellt eine kursivierte Lemmanummer nach der Abkürzung „vgl.“ dar. Das genaue Format eines Literaturverweises spielt hingegen in diesem Kontext keine Rolle. Die entsprechenden *Features* bestehen für alle Tokenklassen aus dem jeweiligen Typ und (mit Ausnahme der Literaturverweise) dem Text des Tokens. Beim Typ „Wörter“ (dem eigentlich entscheidenden Tokentyp für die Klassifizierung) werden zusätzlich weitere Angaben gemacht:

- Ob das Token eine Zahl ist
- Ob das Token mit einem Großbuchstaben beginnt
- Ob das Token Zeichen enthält, die nicht im deutschen Alphabet vorkommen
- Ob das Token Umlaute enthält
- Ob das Token mit einem Bindestrich startet oder endet
- Ob alle Zeichen dieses Tokens im Zuge des letzten Schrittes (Verbesserung von Fehlern bei der Groß-/Kleinschreibung) von Großbuchstaben in Kleinbuchstaben umgewandelt wurden[27]

Innerhalb eines gewissen Bereichs werden außerdem die *Features* der umgebenden Tokens miteinbezogen[28], da für die Formatierung eines Tokens in den meisten Fällen der Kontext eine größere Rolle spielt, als die Informationen, die sich auf das Token selbst beziehen. Somit können (fast unbegrenzt gültige) Regeln, wie diejenige, dass sich

[27] Diese Angabe ist speziell für Wörter in Kapitälchen nötig, die von der Texterkennung als Großbuchstaben erkannt werden.

[28] Wenn sich innerhalb dieses Bereichs am Ende bzw. am Anfang des Artikels nicht mehr genügend Tokens befinden, werden sie durch ein Pseudo-Token mit der Klasse *POST* bzw. *PRE* ersetzt.

zwischen Anführungszeichen keine Formatierungen befinden, durch das mathematische Modell bis zu einem gewissen Grad gelernt werden. Für das Token „ordidorzu“ im obigen Beispiel würde sich bei einem Bereich der Länge 2 folgendes *Feature* (im *Dictionary*-Format) ergeben:

```
{
    'type': 'word',
    'text': 'oridordzu',
    'digit': False,
    'capital': False,
    'non_german_letters': False,
    'umlauts': False,
    'dash_start': False,
    'dash_end': False,
    'made_lower': False,
    'type-1': 'control',
    'text-1': ' ',
    'type+1': 'control',
    'text+1': ' ',
    'type-2': 'lang',
    'text-2': 'Log.',
    'type+2': 'control',
    'text+2': '„'
}
```

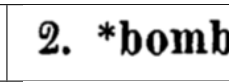
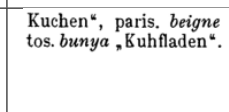
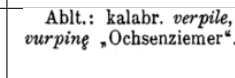
In der Praxis führen aufgrund der sehr feinteiligen Tokenisierung, bei der Leerzeichen, Satzzeichen und ähnliches eigene Tokens sind, größere Bereiche zu besseren Ergebnissen. Für die endgültige Verwendung wurde deshalb eine Länge von 10 verwendet. Die Traningsdaten selbst sind in diesem Fall sehr leicht zu erstellen: Die vorhandenen manuell korrigierten Zeilen werden tokenisiert und anhand ihrer Formatierung einer von vier Klassen (ohne Formatierung, fett, kursiv, Kapitälchen) zugewiesen. Die Eingangsdaten sind die ebenfalls tokenisierten Zeilen ohne Formatierung[29]. Ein *CRF*-Klassifikator, der mit den beschrieben Daten arbeitet führt zu folgenden Ergebnissen:

[29] Es ist möglich die Formatierung, die einem Token im Verlauf der Texterkennung zugewiesen wurde als zusätzliches Merkmal zu verwenden. Für die korrigierten Trainingsdaten wäre das allerdings aufwendig, da wieder das bereits oben erwähnte Problem auftritt, dass nach den Korrekturen die Tokens nicht unbedingt denen entsprechen, die im OCR-System erkannt wurden. Da ein solcher zusätzlicher Eintrag in einzelnen Versuchen außerdem nicht zu besseren Ergebnissen führte, wurde hier darauf verzichtet.

Seitensegment	Keine Formatierung	Kursiv	Fett
0141_2	-0,5% (- 1,4)	-1,7% (- 28,4)	2,1% (- 248,8)
0164_1	0,2% (- 3,8)	-2,4% (- 18,6)	18,4% (+ 6,3)
0823_2	0,1% (- 6,7)	-1,8% (- 17,9)	18,1% (+ 1,6)

Tabelle 4.5: Abweichungen der jeweiligen Formatierung nach den CRF-basierten Korrekturen. In Klammern wird die Änderung des Absolutbetrags gegenüber den Ergebnissen der Texterkennung in Prozentpunkten angegeben.

Zuerst ist ersichtlich, dass es in allen Spalten außer der des Fettdrucks deutliche Verbesserungen gibt. Besonders die sehr geringe Abweichung bei den nicht formatierten Zeichen zeigt, dass ein sehr großer Anteil an formatierten Zeichen auch als solche erkannt wurden. Bei der Kursivierung hat sich die Erkennungsquote ebenfalls sehr deutlich verbessert, sodass nur noch sehr vereinzelt kursive Tokens nicht als solche erkannt werden. Im Bereich des Fettdrucks sehen die Zahlen auf den ersten Blick eher mittelmäßig aus. Zum einen wurden (erwartungsgemäß) die Ergebnisse im ersten Beispiel, das einige falsche Positive enthielt, deutlich verbessert, andererseits haben sich die Werte bei den beiden anderen Beispielen verschlechtert. Hierbei sollte allerdings beachtet werden, dass alle Prozentwerte relativ zum Vorkommen der jeweiligen Formatierung im Text sind. Da fett gedruckte Tokens nur einen sehr geringen Anteil am Gesamttext haben, wirken diese Werte deutlich extremer als sie in der Praxis sind. Konkret ist es in allen drei Beispielen so, dass genau ein Token fehlerhaft als fett formatiert erkannt wurde. Obwohl dies somit keine allzu hohe Relevanz hat, zeigen die Beispiele deutlich, in welchen Fällen der Ansatz zu falschen Ergebnissen führt, und werden deshalb noch kurz im Detail betrachtet. Konkret geht es um die folgenden Textpassagen:

0141_2		2. *bombax. **X**
0164_1		Kuchen“, paris. beigne „Ohrfeige“, tor-tos. **bunya** „Kuhfladen“. Die Sippe ist auf
0823_2		Ablt.: kalabr. verpile, ourpíle, neap.**.**vurpine „Ochsenziemer“.

Im ersten und dritten Beispiel wurden jeweils sehr kleine Artefakte als Zeichen „X“ bzw „.“ interpretiert, während im zweiten eine Sprachabkürzung (aufgrund der Worttrennung) nicht als solche erkannt wurde, was letztendlich dazu führt, dass „tos.“ von der Tokenisierung in zwei Tokens „tos“ und „.“ aufgeteilt wird. In allen Fällen sind also auf tokenisierter Ebene Fehler vorhanden, die in den Trainingsdaten so nicht vorkommen und aufgrund ihrer erratischen Natur auch nicht in den vorherigen Schritten behoben werden konnten. Somit sind diese falschen Markierungen grundsätzlich nachvollziehbar.

Das Problem der Artefakte könnte prinzipiell auf Basis einer entsprechenden Verarbeitung der Ausgangsscans auf Bildebene gelöst werden (vgl. z.B. team 2014) und stellt somit keinen grundsätzlichen Defekt des vorliegenden Ansatzes dar. Der Fall der nicht erkannten Abkürzung liegt dabei etwas anders, da er auftritt, weil die Tokenisierung auf Basis der einzelnen Zeilen stattfindet. Eine Zusammenführung der Zeilen (vgl. Kap. 5.1.2) könnte in diesem Fall eine Lösung sein. Es ist aber auch unabhängig von der Worttrennung das Problem möglich, dass Abkürzungen nicht erkannt werden, da sie in den jeweiligen Listen (noch) nicht vorkommen. Eine eventuelle Lösung könnte der Versuch einer Erkennung der jeweiligen Abkürzungen anstatt der Verwendung einer statischen Liste sein (vgl. hierzu auch Kapitel 14)

4.3.3 Einfügungen, Löschungen und Ersetzungen

Im folgenden werden einzelne Korrekturen auf Einzelzeichenebene betrachtet[30]. Die Grundidee dabei ist es einfache Regeln der Form „Zwischen einem Anführungszeichen und einem Buchstaben steht kein Leerzeichen“ durchzusetzen ohne dass diese explizit formuliert werden. Stattdessen wird nur das abstrakte Problem beispielsweise des Entfernen eines Leerzeichens bestimmt. Dazu werden zuerst auf Basis der bestehenden manuellen Korrekturen alle Positionen gesucht, an denen ein Leerzeichen eingefügt wurde (und umgekehrt auch alle, an denen kein Leerzeichen eingefügt wurde). Diese Daten werden zum Trainieren eines binären Klassifikators benutzt, der im folgenden verwendet werden kann, um bei neuen Textzeilen zu bestimmen an welchen Stellen ein Leerzeichen entfernt werden muss. Der erste Schritt ist somit das Auffinden aller Änderungen eines bestimmten Typs auf Basis der vorhandenen Korrekturdatensätze.

In der Datenbank liegen alle vorgenommen Korrekturen in strukturierter Form vor (vgl. Kap. 8.1). Somit könnte ein Klassifikator prinzipiell direkt auf Basis von diesen trainiert werden. Aufgrund der Tatsache, dass Änderungen prinzipiell in späteren Korrekturen (teilweise) wieder rückgängig gemacht werden können, wurde darauf allerdings verzichtet und die Trainingsdaten beruhen nur auf dem Unterschied zwischen einer importierten Zeile und der korrigierten Fassung. Als Basis für den Vergleich der beiden Zeilen bietet sich die sogenannte Levenshtein-Distanz an, die erstmals in Levenshtein u. a. 1966 vorgestellt wurde. Sie gibt die minimale Anzahl an Ersetzungen, Einfügungen und Löschungen auf Einzelzeichenebene an, die nötig ist, um eine Zeichenkette in eine andere zu überführen. Insbesondere können auch konkrete Folgen von Operationen erzeugt werden, die diese Überführung beschreiben. Klassischerweise wird dies über die Erstellung einer Tabelle berechnet, die die Unterschiede für alle Teilabschnitte der beiden Zeichenketten enthält. Begonnen wird mit zwei leeren Zeichenketten ( ), dann wird pro Spalte ein Zeichen der Ursprungszeile hinzugefügt und entsprechend pro Spalte ein Zeichen der korrigierten Zeile. Die Abstände in der ersten Spalte bzw. Zeile steigen dabei immer um eins an, da sie den Unterschied

[30] Eine Erweiterung auf längere Zeichenketten wäre möglich, wurde hier allerdings nicht benötigt.

zwischen einem Leerstring und dem jeweiligen Teilabschnitt angeben. Alle weiteren Werte werden jeweils aus den drei vorherigen Feldern (links, oben und links oben) berechnet, indem die minimale Operation verwendet wird. Das diagonale Feld steht dabei für Ersetzungen, die den Abstand um eins erhöhen, wenn die Zeichen unterschiedlich sind und ihn ansonsten gleich lassen. Die Felder links und oben stehen für Einfügungen bzw. Löschungen eines Zeichens und erhöhen somit den Abstand immer um eins. Die Tabelle unten symbolisiert dies anhand der Zeichenketten **bumb KnOpf “** und **bumb „Knopf“**:

		b	**u**	**m**	**b**	**␣**	**„**	**K**	**n**	**o**	**p**	**f**	**“**
	0	1	2	3	4	5	6	7	8	9	10	11	12
b	1	0	1	2	3	4	5	6	7	8	9	10	11
u	2	1	0	1	2	3	4	5	6	7	8	9	10
m	3	2	1	0	1	2	3	4	5	6	7	8	9
b	4	3	2	1	0	1	2	3	4	5	6	7	8
␣	5	4	3	2	1	0	1	2	3	4	5	6	7
K	6	5	4	3	2	1	1	1	2	3	4	5	6
n	7	6	5	4	3	2	2	2	1	2	3	4	5
O	8	7	6	5	4	3	3	3	2	2	3	4	5
p	9	8	7	6	5	4	4	4	3	3	2	3	4
f	10	9	8	7	6	5	5	5	4	4	3	2	3
␣	11	10	9	8	7	6	6	6	5	5	4	3	3
“	12	11	10	9	8	7	7	7	6	6	5	4	3

Tabelle 4.6: Tabelle für die Levenshtein-Distanz zwischen zwei Zeichenketten

Es wurden insgesamt drei Korrekturen vorgenommen (Einfügen eines einführenden Anführungszeichens, Ersetzung von *O* durch *o* und Entfernung eines Leerzeichens). Die Levenshtein-Distanz für die beiden vollen Zeichenkette (in Feld ganz rechts unten) ist somit entsprechend drei. Die (in diesem Fall einzige) Liste von Operationen wäre

```
[('add', 5, '„'), ('sub', 8, 'O', 'o'), ('del', 11, ' ')]
```

Diese Pfade sind allerdings nicht immer eindeutig. Für **a c** und **ab** gibt es beispielsweise zwei verschiedene Möglichkeiten:

```
[('sub', 1, ' ', 'b'), ('del', 2, 'c')]
[('del', 1, ' '), ('sub', 1, 'c', 'b')]
```

Im ersten Fall wird das Leerzeichen durch ein *b* ersetzt und das *c* entfernt, während im zweiten Fall das Leerzeichen entfernt wird und *c* durch ein *b* ersetzt wird. Beide Pfade bestehen aus zwei Operationen und sind somit gültig. Wenn nun beispielsweise Stellen gefunden werden sollten, an denen ein Leerzeichen entfernt wurde und nur der erste

Pfad betrachtet wird, würde dieser Fall nicht aufgefunden. Somit wird das Vorhandensein einer bestimmten Korrektur an einer bestimmten Stelle so definiert, dass es mindestens einen Pfad gibt, der eine solche enthält. Das explizite Berechnen aller Pfade ist dabei weniger realistisch, da gerade bei sehr unterschiedlichen Zeichenketten deren Anzahl exponentiell ansteigt[31]. Dies ist allerdings auch nicht nötig, da die Korrekturen auch anhand der Tabelle nach dem obigen Vorbild gefunden werden können. Dazu werden zuerst mögliche Kandidaten für einem bestimmten Typ von Korrektur gesucht und dann überprüft, ob diese Stellen auch auf einem gültigen Pfad liegen. Die Details werden hier am Beispiel einer Löschung besprochen, die anderen Fälle funktionieren analog.

In der obigen Tabelle werden Löschungen durch den Übergang von einem Feld der Tabelle in das darunterliegende dargestellt. Potentielle Löschungen von Leerzeichen können also also an allen solchen Übergängen gefunden werden, die zu einer Zeile führen, die einem Leerzeichen im Originaltext entspricht:

		b	**u**	**m**	**b**	␣	**„**	**K**	**n**	**o**	**p**	**f**	**“**
	0	1	2	3	4	5	6	7	8	9	10	11	12
b	1	0	1	2	3	4	5	6	7	8	9	10	11
u	2	1	0	1	2	3	4	5	6	7	8	9	10
m	3	2	1	0	1	2	3	4	5	6	7	8	9
b	4	3	2	1	0	1	2	3	4	5	6	7	8
␣	5	4	3	2	1	0	1	2	3	4	5	6	7
K	6	5	4	3	2	1	1	1	2	3	4	5	6
n	7	6	5	4	3	2	2	2	1	2	3	4	5
O	8	7	6	5	4	3	3	3	2	2	3	4	5
p	9	8	7	6	5	4	4	4	3	3	2	3	4
f	10	9	8	7	6	5	5	5	4	4	3	2	3
␣	11	10	9	8	7	6	6	6	5	5	4	3	3
“	12	11	10	9	8	7	7	7	6	6	5	4	3

Tabelle 4.7: Alle Stellen, an denen potentiell ein Leerzeichen gelöscht wird. In gelb sind die Ausgangsfelder in grün die jeweiligen Zielfelder markiert.

Eine gültige Levenshtein-Operation hat allerdings nur stattgefunden, wenn der jeweilige Wert der Zelle sich um genau 1 erhöht hat. Das schränkt die Kandidaten auf folgende Fälle ein:

[31] Ein Beispiel aus dem REW ist die Korrektur von **EST PORTANDUM** zu **<u>est</u> <u>portandum</u>**. Der Abstand ist dabei 26, ist gibt aber insgesamt 1144000 verschiedene Pfade

		b	**u**	**m**	**b**	␣	„	**K**	**n**	**o**	**p**	**f**	“
	0	1	2	3	4	5	6	7	8	9	10	11	12
b	1	0	1	2	3	4	5	6	7	8	9	10	11
u	2	1	0	1	2	3	4	5	6	7	8	9	10
m	3	2	1	0	1	2	3	4	5	6	7	8	9
b	4	3	2	1	0	1	2	3	4	5	6	7	8
␣	5	4	3	2	1	0	1	2	3	4	5	6	7
K	6	5	4	3	2	1	1	1	2	3	4	5	6
n	7	6	5	4	3	2	2	2	1	2	3	4	5
O	8	7	6	5	4	3	3	3	2	2	3	4	5
p	9	8	7	6	5	4	4	4	3	3	2	3	4
f	10	9	8	7	6	5	5	5	4	4	3	2	3
␣	11	10	9	8	7	6	6	6	5	5	4	3	3
“	12	11	10	9	8	7	7	7	6	6	5	4	3

Tabelle 4.8: Alle Stellen einer Leerzeichen-Löschung, die in einen gültigen aber potentiell nicht minimalen Pfad enthalten sind. In gelb sind die Ausgangsfelder in grün die jeweiligen Zielfelder markiert.

Alle diese Operationen liegen nur prinzipiell auf gültigen Pfaden, für die Distanz werden allerdings nur jene verwendet, die minimale Länge haben. Alle Fälle, in denen die Distanz in den grünen Feldern bereit größer ist als die Gesamtdistanz drei ist, können bereits vorab aussortiert werden. Für die übrigen Fälle wird die „restliche“ Levenshtein-Distanz ausgerechnet, also die Kosten der Überführung der restlichen Zeichenketten, die nach der aktuellen Position kommen, ineinander.

		b	**u**	**m**	**b**	␣	„	**K**	**n**	**o**	**p**	**f**	“
	0	1	2	3	4	5	6	7	8	9	10	11	12
b	1	0	1	2	3	4	5	6	7	8	9	10	11
u	2	1	0	1	2	3	4	5	6	7	8	9	10
m	3	2	1	0	1	2	3	4	5	6	7	8	9
b	4	3	2	1	0	1	2	3	4	5	6	7	8
␣	5	4	3	2	1	0	1	2	3	4	5	6	7
K	6	5	4	3	2	1	1	1	2	3	4	5	6
n	7	6	5	4	3	2	2	2	1	2	3	4	5
O	8	7	6	5	4	3	3	3	2	2	3	4	5
p	9	8	7	6	5	4	4	4	3	3	2	3	4
f	10	9	8	7	6	5	5	5	4	4	3	2	3
␣	11	10	9	8	7	6	6	6	5	5	4	3	3
“	12	11	10	9	8	7	7	7	6	6	5	4	3

Tabelle 4.9: Alle Stellen einer Leerzeichen-Löschung, die in einen minimalen Pfad enthalten sind. In gelb sind die Ausgangsfelder in grün die jeweiligen Zielfelder markiert. In rot ist ein nicht gültiger Übergang markiert.

Die einzige Stelle, die übrig bleibt ist in der obigen Tabelle markiert und entspricht der tatsächlichen Entfernung eines Leerzeichens an dieser Stelle. Der Abstand ist an dieser Stelle bereits drei, entspricht also dem Ergebnis, allerdings ist die restliche Distanz null, da sowohl der restliche Teil des Originals als auch der des Ergebnisses die Länge eins hat und nur aus einem schließenden Anführungszeichen besteht. Exemplarisch soll noch ein zweiter Übergang betrachtet werden, der in der Tabelle rot markiert ist. Dieser hat den Wert zwei und die beiden Restbestandteile sind **KnOpf “** für das Original und **b „Knopf“** für das Ergebnis. Deren Abstand ist allerdings fünf, somit ist die Summe sieben. Da dieser Wert größer als der endgültige Abstand drei ist, liegt dieser Übergang nicht auf einem minimalen Pfad.

Anmerkung zu Einfügungen und Löschungen: So wie die Levenshtein-Distanz definiert ist, können in seltenen Fällen auch Operationen, die intuitiv eine Löschung bzw. Einfügung sind nicht als solche dargestellt werden. Das ist beispielsweise der Fall, wenn in der direkten Umgebung, in der ein Leerzeichen entfernt wurde, auch vom OCR-System ein einzelner Buchstabe als zwei Zeichen erkannt wurde Das führt dazu, dass Original (z.B. **a m)** und Ergebnis (z.B. **arr**) die gleiche Länge haben. Dadurch ist der kürzeste Pfad eindeutig und besteht aus zwei Ersetzungen:
`[('sub', 1, ' ', 'r'), ('sub', 2, 'm', 'r')]`. Um auch diese Fälle zu berücksichtigen wurden, wurden bei den Einfügungen und Löschungen von Leerzeichen auch Fälle berücksichtigt, in denen ein Leerzeichen durch ein anderes Zeichen ersetzt wurden bzw. ein Zeichen durch ein Leerzeichen ersetzt wurde. Dies ist im Fall von Leerzeichen unproblematisch, da es kein anderes Zeichen gibt, das mit einem Leerzeichen „verwechselt“ werden könnte, es also keine Ersetzung in diesem Sinne gegeben haben kann. Für andere Zeichen ist das aber nicht unbedingt der Fall.

Auf Basis dieser Form der Korrekturerkennung wurden insgesamt fünf Klassifikatoren für verschiedene Anwendungsfälle erstellt:

Klassifikator	**Granularität**	**Methode**
Deletion	Zeichen	SVM
Insertion	Zeichen	SVM
Replacement	Zeichen	SVM
ReplacementTokens	Tokens	SVM
CaseReplacement	Zeichen	CRF

Tabelle 4.10: Implementierte Klassifikatoren für Einfügungen, Löschungen und Ersetzungen

Zu beachten ist dabei, dass sich die Granularität nur auf die Umgebung bezieht. Es wird auch im Fall ReplacementTokens ein einzelnes Zeichen ersetzt, in diesem Fall werden allerdings die umgebenden Tokens betrachtet und nicht die umgebenden Zeichen.

Die ersten drei Klassifikatoren sind sich dabei sehr ähnlich. Sie suchen nach Stellen in den korrigierten Daten, an denen ein bestimmtes Zeichen eingefügt oder entfernt bzw.

ein Zeichen durch ein bestimmtes anderes ersetzt wurde und verwenden die Umgebung dieser Stelle innerhalb eines gewissen Rahmens *n*. Dabei werden verschiedene Zeichentypen unterschieden:

Zeichen-typ	**Zeichen**
1	Textanfang bzw. -ende
2	Buchstaben
3	Ziffern
4 - n	Sonderzeichen wie Punkte, Leerzeichen, Kommata etc. (Jedes stellt einen eigenen Typ dar)

Tabelle 4.11: Verschiedene Zeichentypen für zeichenbasierte Ersetzungen

Es werden also vor allem Sonderzeichen voneinander unterschieden. Das hat den Hintergrund, dass entscheidend vor allem strukturelle Verbesserungen sind, wobei die Struktur aus genau diesen Elementen aufgebaut ist. Für das Einfügen und Entfernen von Leerzeichen werden sogar ausschließlich Umgebungen betrachtet, die mindestens ein Sonderzeichen enthalten. Das ist der Fall, da Leerzeichen die zwischen Buchstaben eingefügt oder entfernt werden müssen, auf dieser Basis nicht sinnvoll erkannt werden können. In diesem Fall führen auch nur minimale Rahmengrößen der Länge eins zu guten Ergebnissen. Ein Datenpunkt symbolisiert so eine Aussage der Form „vor dem Zeichen kommt ein Leerzeichen, nach dem Zeichen kommt ein Buchtstabe“.

Auf Basis der Zeichentypen werden numerische Vektoren erzeugt, die eine SVM verwenden kann. Als Klassen werden wie bereits erwähnt die Werte 0 und 1 verwendet, die beschreiben, ob die Änderung an dieser Stelle durchgeführt werden soll oder nicht. Für die Trainingsdaten werden also alle Positionen, an denen eine entsprechende Korrektur stattfand mit eins markiert, alle anderen mit Null.

Angewendet werden kann dieser Klassifikator (außer für die Behandlung der Leerzeichen) für explizite Ersetzungen von beispielsweise Langstrichen durch Kurzstrichen, aber auch für allgemeinere Aufgaben wie die Ersetzung von Ziffern durch Buchstaben. Hierzu wurden die Ersetzungen zusätzlich auf Fälle untersucht, in denen Ziffern durch Buchstaben ersetzt wurden und wenn deren Anzahl einen gewissen Schwellenwert überschritt, wurde ein Klassifikator damit trainiert. In der Praxis kamen in den vorliegenden Daten allerdings nur die erwartbaren Fälle von *0* -> *o* und *1* -> l in entsprechender Häufigkeit vor.

Der Klassifikator des Typs *ReplacementTokens* arbeitet prinzipiell analog, es werden

allerdings statt der umgebenden Zeichen die umgegebenden Tokens verwendet. Die Tokenisierung unterscheidet sich dabei zum Teil von der in Kap. 4.3.3 vorgestellten und entspricht eher den oben besprochenen Zeichenklassen. Alle Sonderzeichen werden voneinander unterschieden die restlichen Tokens werden einer der folgenden Typen zugeordnet:

Tokentyp	**Token**
upper	Alle Tokens, die mit einem Großbuchstaben beginnen. Das schließt auch sprachliche Formen und großgeschriebene Sprachabkürzungen ein.
abbr	Andere Abkürzungen und Literaturverweise
lower	Token, die mit einem Kleinbuchstaben beginnen
digit	Zahlen

Tabelle 4.12: Verschieden Tokentypen für die Ersetzungen

Der Anfang bzw. das Ende des Texts werden hier wie das Sonderzeichen für den Zeilenumbruch behandelt. Ansonsten werden wieder anhand einer gewissen Rahmengröße[32] die numerischen Daten erzeugt und wie oben beschrieben verwendet.

Im Unterschied zu den vorherigen eignet sich dieser Klassifikator vor allem für Fälle, die mehr von den Struktur des umgebenden Satzes abhängen und weniger von der direkten Umgebung. In der Praxis wird er aktuell nur für die Ersetzung von Punkten durch Kommata eingesetzt, da hier gerade die großräumigere Umgebung und die Groß- und Kleinschreibung relevant sind.

Der letzte Klassifikator wurde schließlich speziell für die Ersetzung von Großbuchstaben durch Kleinbuchstaben konzipiert und kann weniger allgemein verwendet werden, als die oben beschriebenen. Auch hier wurden für alle Zeichen in einer bestimmten Umgebung[33] betrachtet, wobei in diesem Fall die folgenden Merkmale für ein Zeichen verwendet wurden:

char	Das Zeichen selbst
is_letter	Ob das Zeichen ein Buchstabe ist
is_upper	Ob das Zeichen ein Großbuchstabe ist
latin_letter	Ob das Zeichen aus dem lateinischen Alphabet (inlusive Längen- und Kürzenzeichen) ist
ger-man_letter	Ob das Zeichen aus dem deutschen Alphabet ist

Auf Basis dieser Merkmale wurde entsprechend ein binärere *CRF*-Klassifikator

[32] Als guter Wert hat sich hier eine Größe von drei herausgestellt.
[33] Drei erwies sich hier ebenfalls als guter Wert

trainiert, der bestimmt, ob ein Großbuchstabe durch seine Kleinschreibung ersetzt werden soll oder nicht.

4.3.4 Wortlistenbasierte Korrekturen

Der letzte Typ von Korrektur bezieht sich auf bekannte Wörter, in denen einzelne Zeichen falsch erkannt wurden. Grundsätzlich können solche Fälle bereits im Laufe des OCR-Prozesses korrigiert werden, wenn entsprechende Wortlisten als *Dictionary* übergeben werden. Wie bereits erwähnt führt dies allerdings in vielen Fällen weiterhin zu Fehlerkennungen, obwohl die entsprechenden Wörter im *Dictionary* enthalten sind[34]. Somit wurde hierfür ein zusätzliche Korrekturmöglichkeit eingerichtet.

Das Vorgehen basiert wiederum auf der Levenshtein-Distanz und entspricht prinzipiell bekannten Ansätzen, die früh auf deren Basis definiert wurden (vgl. z.B. Damerau 1964)[35]. In diesem Fall werden allerdings prinzipiell nur Fälle behandelt, in denen Tokens gefunden wurden, die sich in genau einem Zeichen von einem bekannten Token unterscheiden, da eine gewisse Unvollständigkeit der entsprechenden Listen zu erwarten war. Außerdem werden nur dann Ersetzungen vorgenommen, wenn sich der neue und der alte Buchstabe ähnlich sind. Ähnliche Buchstaben waren dabei solche, die eine der folgenden Bedingungen erfüllten:

- Die beiden Buchstaben unterscheiden sich nur hinsichtlich ihrer Diakritika
- In den korrigierten Zeilen kommen mindestens 10 Fälle vor, in denen der eine Buchstabe durch den anderen ersetzt wurde

Durchgeführt wurde dieser Korrekturschritt zum einen für Sprach- sowie bibliographische Abkürzungen und zum anderen für deutsche Wörter[36]. Die Ersetzungen kommen allerdings an verschiedenen Stellen im Prozessablauf (vgl. Abb. 4.9) vor. Das hat den Hintergrund, dass die Korrektur der Abkürzungen möglichst vor der Korrektur von Groß- und Kleinschreibung stattfinden sollte, da viele Abkürzungen mehrere Großbuchstaben enthalten, die falls die Abkürzung nicht als solche erkannt wird, oftmals fälschlich durch Kleinbuchstaben ersetzt werden. Die Korrektur der deutschen Wörter muss andererseits nach der Erkennung der Formatierung stattfinden, damit sichergestellt werden kann, dass möglichst keine sprachlichen Formen betrachtet werden. Die Formaterkennung muss wiederum nach

[34] Der Grund hierfür konnte trotz verschiedenen Untersuchungen nicht gefunden werden und hängt wohl mit der inneren Arbeitsweise des OCR-Systems zusammen. Ein möglicher Faktor könnte hierfür sein, dass ein großer Teil des Quellentexts nicht aus deutschen Wörtern besteht.

[35] Eine zusätzliche Verbesserung könnte evtl. mit komplexeren Ansätzen erreicht werden (vgl. z.B. Jean-Caurant u. a. 2017)

[36] Im Fall des REW sind damit alle Tokens gemeint, die weder speziell formatiert sind noch eine bestehende Abkürzung darstellen. Diese sind in den meisten Fällen deutsche Wörter aus natürlichsprachlichen Passagen.

der Behandlung der Groß- und Kleinschreibung stattfinden, da ein Merkmal für die Behandlung der Formatierung eine eventuelle Änderung von Großbuchstaben zu Kleinbuchstaben ist (vgl. Kap. 4.3.3). Dies ergibt zusammen die zu Beginn von Kap. 4.3 angegebene Reihenfolge.

Um falsche Positive möglichst auszuschließen wurden weiterhin bei den bibliographischen Abkürzungen nur solche verwendet, die aus mindestens drei Zeichen bestehen und zumindest ein Sonderzeichen (im Normalfall einen Punkt oder Bindestrich) enthalten. Bei den Sprachabkürzungen wurden nur solche verwendet, die auf einen Punkt enden.

Das grundsätzliche Vorgehen kann damit folgendermaßen zusammengefasst werden:

- Finde jeweils alle Tokens, die den oben genannten Bedingungen genügen
- Falls das Token mit einem Großbuchstaben beginnt, an einer Stelle im Text auftaucht, nach der eine Großschreibung erwartet wird (beispielsweise nach einem Punkt und einem Leerzeichen oder einer öffnenden Klammer) und es ein bekanntes Wort gibt, dass sich nur in der Kleinschreibung des ersten Buchstabens unterscheidet, ignoriere das Wort. Diese Regel wird nur für die deutschen Wörter verwendet und verhindert, dass ein am Satzanfang groß geschriebenes Wort durch die eigentliche kleingeschriebene Variante ersetzt wird.
- Falls es ein Wort gibt, das identisch bis auf die Verwendung eines *ss* anstatt eines *ß* ist, ignoriere das Wort (Wort ist in alter Rechtschreibung)
- Finde alle Wörter aus der Wortliste, die sich in maximal einem Zeichen unterscheiden[37]
- Falls es ein identisches Wort in der Wortliste gibt, ist das Wort korrekt und wird nicht weiterbehandelt.
- Falls es genau einen Kandidaten gibt, der sich nur durch die Groß-/Kleinschreibung von dem Wort unterscheidet, ersetze es durch diesen.
- Ansonsten, falls es genau einen Kandidaten gibt, ersetze das Wort durch diesen.

[37] Für das effizientere Auffinden dieser Kandidaten wurde die Bibliothek RapidFuzz verwendet.

5 Strukturelle Erkennung

Dieses Kapitel behandelt den zweiten Schritt der in Kap. 3.3 gegebenen Prozesskette, also die Umwandlung von Text in eine formelle digitale Repräsentation, die dessen inhaltliche Zusammenhänge abbildet. Entscheidend ist hierbei eine detaillierte Analyse des Aufbaus, der Formatierung und der Notationen der entsprechenden Quelle. Dabei kann der Quellentext zuerst in Abschnitte unterteilt werden, die zumeist den groben Aufbau des Werks widerspiegeln. Ein solcher Abschnitt wird formell über die entsprechenden Seitensegmente (vgl. hierzu Kap. 4.1) definiert und umfasst inhaltlich Passagen die aus Wiederholungen von gleich strukturierten Textabschnitten bestehen. Im Falle eines Wörterbuchs sind das beispielsweise der Hauptteil, der die eigentlichen Artikel enthält, oder verschiedene vorhandene Verzeichnisse. Die folgende Tabelle zeigt diese Einteilung für das REW:

Seite Start	**Segment Start**	**Seite Ende**	**Segment Ende**	**Inhalt**	**Datenbank-tabelle**
6	1	13	1	Vorwort[4]	
14	1	26	2	Bibliographie	bibliography
26	3	31	2	Sprachabkürzungen	lang_abbreviations
32	1	32	2	Andere Abkürzungen	abbreviations
32	3	32	4	Zeichen[5]	
36	1	860	2	Artikel	entries
861	1	871	2	Nachträge	entries + entry_supps
872	1	1270	3	Wortverzeichnisse[6]	
1271	1	1274	3	Verbesserungen	source_corrections

Tabelle 5.1: Einteilung des REW in Sinnabschnitte und die Tabellen für deren Inhalte in der in Kapitel 6 besprochenen Datenbank

Insgesamt kann der Ablauf der strukturierten Erfassung dann in drei Phasen beschrieben werden, die hierarchisch von oben nach unten angeordnet sind:

- Einteilung der Quelle in Abschnitte
- Gruppierung der zusammengehörigen Zeilen, die zu einem Bestandteil innerhalb des Abschnitts gehören
- Strukturelle Erkennung der einzelnen Bestandteile

Die textuellen Daten liegen als einzelne Textzeilen[7] vor. In der Gruppierungsphase werden dann alle Zeilen innerhalb eines Abschnitts entsprechend ihrer Zugehörigkeit zu einer bestimmten Entität gruppiert (Kap. 5.1), es werden also beispielsweise alle Zeilen, die zu einem speziellen Wörterbuchartikel gehören, einer Gruppe hinzugefügt. Dies beinhaltet ebenfalls die Transformation von Textzeilen in Fließtext, also insbesondere die Auflösung von Zeilentrennungen. Der letzte (und mit Abstand komplexeste) Schritt ist dann die strukturelle Erfassung der im vorherigen Schritt entstandenen Textpassagen. Diese Arbeit schlägt dazu die Verwendung einer formellen Grammatik vor. Kap. 5.3 analysiert dazu die speziellen Eigenheiten von Quellen, die sowohl stark formalisierte Bestandteile als auch natürlichsprachige Einlassungen enthalten, und welche Anforderungen das an die Abbildung derer hierarchischer Struktur mit Hilfe einer formellen Grammatik stellt. Im letzten Kapitel werden diese Prinzipien auf das konkrete Beispiel eines Wörterbuchartikels angewandt (Kap. 5.3). In allen Teilabschnitten liegt dabei der Fokus darauf ein tragfähiges und generalisierbares Modell zu entwickeln, welches einen möglichst großen Teil der (in hohem Maße inkonsistenten) Bestandteile eines solchen Texts logisch abbilden kann.

5.1 Bündelung der Zeilen

Die Rohdaten nach dem Import der OCR-Ergebnisse liegen als einzelne Zeilen vor, deren Datensätze zusätzlich zum eigentlichen Inhalt drei Felder mit kontextueller Information beinhalten, nämlich die Seitenzahl, das Seitensegment und die Zeilennummer innerhalb des Segments. Von der Texterkennung gefundene Paragraphenenden werden als leere Zeilen abgebildet; sie werden also nicht gesondert indiziert (vgl. die folgende Tabelle). Das hat den Grund, dass die Zuordnung zu Abschnitten fehlerbehaftet ist und bei diesem Modell die Indizierung nach dem initialen Import konstant bleibt, also sämtliche Korrekturen auf inhaltlicher und nicht auf struktureller Ebene stattfinden.

[7] Für eine ausführliche Diskussion der Granularität der textuellen Eingangsdaten vgl. Kap. 6.1

id_line	content	page	section	line_number
7850	47. absque „fern von“.	40	1	1
7852		40	1	2
7854	Lomb. aska „außer“ Diez 353.	40	1	3
7856		40	1	4
7858	48. abstĕrgĕre „abwischen“.	40	1	5
7860		40	1	6
7862	Rum. şterge, it. astergere, campid.	40	1	7
7864	stréžiri, afrz. esterdre, prov. esterzer.	40	1	8
7866		40	1	9

Tabelle 5.2: Textzeilen in der relationalen Datenbank (verkürzte Darstellung)

Die Umwandlung dieses Formats in den Fließtext, der einen bestimmten Bestandteil der Quelle (also z.B. einen einzelnen Wörterbuchartikel) darstellt besteht aus zwei Schritten: Die Erkennung des Beginns eines Bestandteils und das Zusammenführen von der jeweiligen Zeilen.

5.1.1 Erkennung von Artikelanfängen

Eine Methode zur Durchführung des ersten Schritts ist sehr stark von der Formatierung der jeweiligen Quelle abhängig, sollte aber im Normalfall kein größeres Problem darstellen. Das grundsätzliche Kriterium für den Artikelanfang[8] im REW betrachtet, ob eine Zeile mit einer Nummer beginnt, die von einem Punkt, einer Leerzeile und einem fett formatierten Wort[9] gefolgt wird (die zusätzliche Einrückung, die im Ursprungstext vorhanden ist, wird bei der Texterkennung nicht abgebildet). Dies deckt bereits die meisten Fälle ab, verursacht aber noch Probleme bei Einträgen wie in der folgenden Abbildung, bei der die Liste der nummerierten Lemmata über eine Zeile hinausgeht.

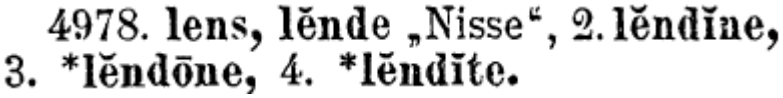


Abbildung 5.1: Nummerierte Lemmata über zwei Zeilen (REW, S. 4978)

Somit wird als zweites Kriterium verwendet, dass die jeweilige Nummer größer als die

[8] Hier wird die Bündelung der Zeilen speziell am Beispiel eines Wörterbuchartikels besprochen. Für andere Bestandteile (z.B. die Bibliographie) funktioniert diese ähnlich.

[9] Entscheidend ist in diesem Fall natürlich eine gute Erkennung der Formatierung, die aber gerade im Bereich des Fettdrucks relativ sicher ist (vgl. Kap. 4.3)

des letzten Zeilenanfangs ist. Damit können bis auf 15 (Stand 01.09.2023) Spezialfälle, die durch entsprechende Ausnahmeregeln behandelt werden (siehe Kap. 5.1.3), die Anfänge von allen 10701 (Stand 01.09.2023) Einträgen korrekt bestimmt werden.

5.1.2 Zusammenführung von Zeilen

Die Erstellung von Fließtext aus den erkannten Zeilengruppen ist dann im Gegensatz zum bisherigen Vorgehen nicht von vom konkreten Textabschnitt abhängig, sondern nur von den in der Quelle verwendeten typographischen Regeln. Die gruppierten Zeilen werden nacheinander verkettet, sodass schließlich der Fließtext entsteht. Leerzeilen werden als Zeilenumbruch also als HTML-Element
 in die Verkettung aufgenommen. Grundsätzlich können drei verschiedene Arten des Zusammenfügens von Zeilen unterschieden werden:

1. Die Zeilen werden verbunden, wie sie sind.
2. Am Ende der ersten Zeile wird ein Trennstrich entfernt. (Dies beinhaltet im Fall des REW noch den Spezialfall, dass die veraltete deutsche Trennung des *ck* mit *k-k* vorhanden ist).
3. Zwischen den Zeilen wird ein Leerzeichen eingefügt.

Der erste Fall tritt auf, wenn die vorherige Zeile nur aus einem Zeilenumbruch bestand, aber auch falls eine Abkürzung[10], die einen Bindestrich enthält, an diesem getrennt wurde:

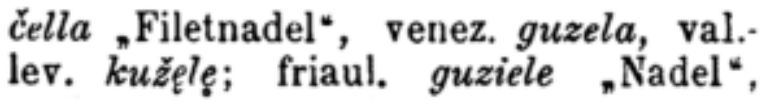


Abbildung 5.2: Beispiel für eine Trennung, bei der der Bindestrich erhalten bleibt (REW, S. 118).Es wird also „... val.-lev. ...“ (Abkürzung für „Mundart der Val Leventina“) und nicht „... val.lev. ...“ erstellt.

Der zweite Fall tritt auf, wenn eine Zeilentrennung stattgefunden hat und kann strukturell daran erkannt werden, dass die erste Zeile auf einen Trennstrich endet und die zweite mit einem Kleinbuchstaben beginnt oder dass die erste Zeile auf Großbuchstaben endet und die zweite mit Großbuchstaben beginnt[11]. Die dritte

[10] Hierfür ist die Importreihenfolge Allgemeine Abkürzungen / Sprachabkürzungen -> Bibliographische Abkürzungen -> Textuelle Elemente (vgl. Kap. 3.1) entscheidend, um an dieser Stelle auf eine Liste von Abkürzungen zurückgreifen zu können.

[11] Worte in Großbuchstaben kommen so im REW nicht vor, allerdings kann dieser Fall z.T. bei nicht erkannten Kapitälchen, die als Großbuchstaben dargestellt werden, auftreten.

Variante ist schließlich der Normalfall, in dem die zweite Zeile mit einem neuen Wort beginnt. Dies bedeutet allerdings nicht zwingend, dass die erste nicht trotzdem mit einem Trennstrich endet:

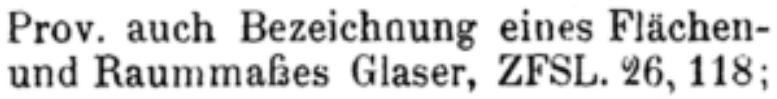


Abbildung 5.3: Beispiel für einen Trennstrich, nach dem ein Leerzeichen eingefügt werden muss (REW, S. 1881)

Das folgende Diagramm symbolisiert die Behandlung der drei Fälle (ohne Berücksichtigung von HTML-Tags für die Formatierung):

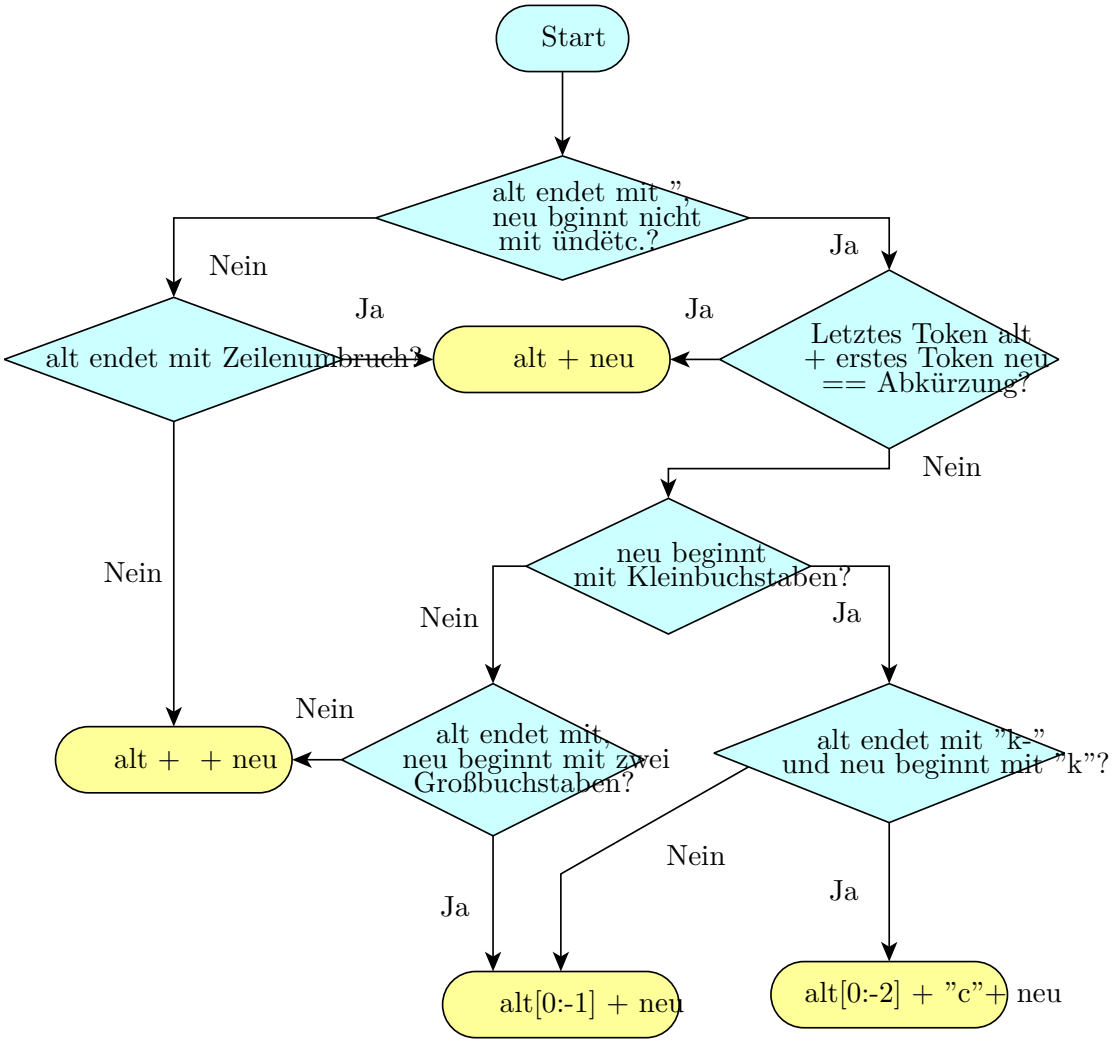


Abbildung 5.4: Vereinfachtes Flußdiagramm für die Zusammenführung von Zeilen ohne Berücksichtigung von Formatierungstags und Ausnahmen

5.1.3 Ausnahmen auf Zeilenebene

Die Ansätze aus den vorhergehenden Kapiteln sorgen dafür, dass weite Teile des Texts korrekt bearbeitet werden können. Trotzdem ist es nicht auszuschließen, dass sie an einzelnen Stellen versagen, was eine Korrekturmöglichkeit erfordert. Diese wird mit den in Kap. 3.2 besprochenen Ausnahmen umgesetzt. Wie alle Klassen von Ausnahmen[12]

[12]Der hier präsentierte Ansatz unterscheidet drei Klassen von Ausnahmen, je nachdem in welchem Kontext sie vorkommen und in welcher Struktur sie in der Datenbank abgelegt werden. Die beiden anderen werden in Kap. 5.2.6 und Kap. 7.3 besprochen.

beziehen sie sich nicht auf konkrete Elemente in der Datenbank (d.h. deren Identifikatoren) sondern allgemeiner auf den Kontext. In diesem Fall werden sie über einen Typ und die bekannte Positionsindizierung über Seite, Sektion und Zeilennummer[13] definiert. Die verschiedenen Typen, die in diesem Bereich notwendig sind, werden in der folgenden Tabelle illustriert. Wie man an den jeweiligen Mengen erkennen kann, handelt es sich beim ganz überwiegenden Teil um die Korrektur von Zeilenumbrüchen (siehe auch Kap. 12.5), die im Zuge der Texterkennung fehlerhaft behandelt wurde, die weiteren Typen beziehen sich eher auf seltene Spezialfälle, die algorithmisch nicht explizit berücksichtigt wurden (vgl. hierzu auch Kap. 3.2).

[13] Das hat in diesem Fall den Vorteil gegenüber der Verwendung der ID der entsprechenden Zeile, dass bei einem eventuellen Neuimport ganzer Sektionen (z.B. bei gravierenden Mängeln in der Texterkennung o.ä., vgl. Kap. 6.1), der zu neuen Zeilen-IDs führt, die Ausnahmen gültig bleiben (vorausgesetzt die Anzahl der Zeilen verändert sich nicht).

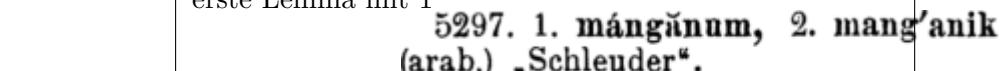
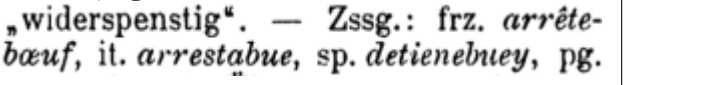
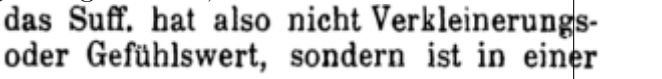
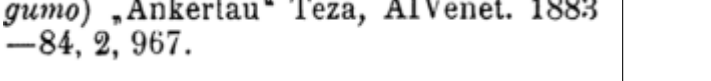
Typ	Anzahl	Beschreibung
is_new_entry	14 (Stand 01.09.2023)	Gibt an, dass diese Zeile der Beginn eines neuen Eintrags ist. Ein Beispiel für eine solche Ausnahme ist (REW, S. 5297), der (als einziger Artikel im REW) das erste Lemma mit 1  5297. 1. mángănum, 2. mang'anik (arab.) „Schleuder". nummeriert. Abbildung 5.9: Beispiel für einen unkonventionell beginnenden Artikel
no_new_entry	3 (Stand 01.09.2023)	Gibt umgekehrt an, dass diese Zeile nicht der Beginn eines neuen Eintrags ist.
is_new_line	351 (Stand 01.09.2023)	Gibt an, dass vor dieser Zeile ein Absatz vorhanden ist, der von der Texterkennung nicht als solcher erkannt wurde.
no_new_line	716 (Stand 01.09.2023)	Komplementär zu „is_new_line", falls die Texterkennung einen zusätzlichen falschen Absatz erkannt hat.
keep_hyphen	12 (Stand 01.09.2023)	Gibt an, dass der Trennstrich am Ende der vorherigen Zeile behalten werden soll. Beispiele hierfür sind vor allem sprachliche Konstrukte, die einen Bindestrich enthalten und genau an dieser Stelle getrennt werden. Dies kann rein syntaktisch ohne Zusatzwissen nicht erkannt werden, vgl.  „widerspenstig". — Zssg.: frz. *arrête-bœuf*, it. *arrestabue*, sp. *detienebuey*, pg. z.B. Abbildung 5.10: Beispiel für einen Bindestrich, für den ohne Zusatzinformation nicht entschieden werden kann, dass er beim Zusammenführen der Zeilen behalten werden muss (REW, S. 673)
keep_hyphen_space	1 (Stand 01.09.2023)	Gibt an, dass der Bindestrich behalten werden soll, aber trotzdem ein Leerzeichen eingefügt werden soll, z.B. bei Ergänzungsstrichen, die nicht erkannt  das Suff. hat also nicht Verkleinerungs- oder Gefühlswert, sondern ist in einer werden: Abbildung 5.11: Beispiel für einen Ergänzungsstrich am Ende einer Zeile (REW, S. 8059)
no_space	1 (Stand 01.09.2023)	Gibt an, dass an dieser Stelle kein Leerzeichen eingefügt werden  *gumo*) „Ankertau" Teza, AIVenet. 1883 —84, 2, 967. soll. Abbildung 5.12: Halbgeviertstrich am Zeilenanfang. Ein Spezialfall, der in der Routine zum Zusammenfügen der Zeilen nicht berücksichtigt wurde (REW, 3952a)

Tabelle 5.3: Die verschieden Typen von Ausnahmen auf Zeilenebene

5.1.4 Indizierung der Zeilen im Fließtext

Bei der Umwandlung von Zeilen in Fließtext kann es außerdem hilfreich sein die Position der Zeilen im neu erstellen Text explizit zu speichern. Das ist für alle Operationen entscheidend, die auf Basis des Fließtexts durchgeführt werden, deren Ergebnisse sich aber auf die zugrunde liegenden Zeile auswirken (vgl. hierzu die Durchführung von Massenkorrekturen, wie sie in Kap. 8.2 vorgestellt werden, und das Erstellen von Ausnahmen für die Grammatik in Kap. 5.2.6 und Kap. 12.5). Eingefügte Leerzeichen werden dabei der Zeile vor dem Leerzeichen zugerechnet, vgl. das folgende Beispiel aus (REW, 1a):

id_line	**content**	**index_start**	**index_end**
6416	1a. aanmarren (ndl.) „ein Schiff an-	0	42
6418	binden“.	42	50
6420		50	56
6422	Frz. amarrer (> it. amarrare, sp.,	56	105
6424	pg. amarrar). — Diez 15; Baist, ZDWF.	105	150
6426	4, 272.	150	157
6428		157	157

Tabelle 5.4: Zeilen eines Artikels und deren Indizes im erstellten Fließtext

1a. aanmarren (ndl.) „ein Schiff an binden“.
 Frz. amarrer (> it. amarrare, sp., pg. amarrar). Diez 15; Baist, ZDWF. 4, 272.

Die Indexe sind dabei als halboffene Intervalle [start, end[definiert, was dazu führt, dass bei Zeilen, die nicht verwendet werden, der Start-Index dem End-Index entspricht. Das ist dann der Fall wenn ein Absatz von der Texterkennung falsch erkannt wurde bzw. ein Artefakt als zusätzliche Zeile.

5.2 Abbildung von semi-strukturierten Texten

Nach der Erstellung des Fließtexts, der einem bestimmten Element zugeordnet ist, kann nun auf dessen Basis die eigentliche Tiefenerschließung durchgeführt werden. Dazu ist es hilfreich zuerst die besondere Struktur von Wörterbüchern bzw.

Nachschlagwerken im Allgemeinen zu betrachten. Der Duden beschreibt ein solches als „Buch (besonders Lexikon, Wörterbuch), das in übersichtlicher, meist alphabetischer Anordnung der schnellen Orientierung über etwas dient“ (Duden „Nachschlagwerk“). Als bezeichnendes Merkmal wird hier also vor allem die übersichtliche Anordnung genannt, um es von anderen Publikationsformen abzugrenzen. Daraus kann eine starke Formalisierung der Makrostruktur des Werkes abgeleitet werden, wie sie ja auch schon im vorherigen Kapitel für die Unterteilung verwendet wurde. Über die Strukturierung der einzelnen Bestandteile wird keine Aussage getroffen, aber auch dafür wird über den Zweck der „schnellen Orientierung“ zumindest angedeutet, dass eine gewisse klare Form vorhanden sein könnte. Deren Ausgestaltung ist von Gattung zu Gattung unterschiedlich (eine Enzyklopädie wird tendenziell eine weniger starke Strukturierung aufweisen als ein Wörterbuch), auch innerhalb einer Gattung oder auch eines einzelnen Werkes können große Unterschiede vorhanden sein. Exemplarisch sollen hierfür die beiden Einträge 6755a und 6796 des REW stehen:

6755a. *principiāre „anfangen“.
[It. *principiare*] dringt von der Toskana und Emilia mehr und mehr in die städtischen Zentren ein und verschmilzt mit *cominciare* namentlich in der Emilia zu *kmintsipiar* Jaberg, RLiR. 1, 129.

Abbildung 5.13: Artikel mit hauptsächlich natürlichsprachlichem Inhalt

6769. prŏfĕctus „Vorteil“.
Apiem. *profet*, frz. *profit* (> it. *profitto*), prov. *profech*, sp. *provecho* (> ait. *proveccio*), pg. *proveito*.

Abbildung 5.14: Stark strukturierter Artikel aus dem REW

Ersterer besteht fast vollständig aus natürlichsprachlichem Text, während zweiterer extrem formalisiert ist. In der Praxis befinden sich die Artikelinhalte meist zwischen diesen beiden Extremen, sind also ein Hybrid zwischen einem vollständig strukturierten Text (z.B. Programmcode) und rein natürlichsprachlichen Texten (z.B. wissenschaftliche Aufsätze, literarische Texte). Dabei ist es durchaus nicht immer der Fall, dass beides klar von einander getrennt auftritt. In vielen Fällen enthalten auch strukturierte Passagen natürlichsprachliche Einschübe verschiedener Länge (vgl. hierzu auch Kap. 5.3.3).

Dass die stark formalisierten Passagen für die Extraktion von Information anhand des Modells aus Kap. 2.2 genutzt werden sollten, steht außer Frage, da sie den Großteil der dafür relevanten Information enthalten. Der Umgang mit natürlichsprachlichen Abschnitten steht auf einem anderen Blatt. Ein gewisser Teil enthält sehr

einzelfallspezifische weiterführende Informationen, die nicht mit realistischen Aufwand formalisiert abgebildet werden und bei denen es auch durchaus streitbar ist, ob dies überhaupt sinnvoll wäre. Diese Arbeit schlägt deshalb die folgende grundlegende zweistufige Herangehensweise an dieses Problem vor:

- Im ersten Schritt werden die formalisierten Anteile des Quellentexts erfasst und deren Hierarchie und Anordnung abgebildet. Nicht strukturell erkennbare Elemente[14] werden als als textuelle Bestandteile behandelt.

- Im zweiten Schritt werden alle textuellen Elemente auf einzelne strukturell erkennbare Bestandteile untersucht und diese entsprechend annotiert. Es findet hierbei also eine Form von *Named-entity recognition* im informatischen Sinne statt. Der Informationsgehalt, der aus diesen Abschnitten gewonnen werden kann, ist im Gegensatz zu den vollständig erschlossenen Textbestandteilen aus dem ersten Schritt deutlich geringer, da keine Zusammenhänge zwischen den einzelnen Bestandteilen verwendet werden können (vgl. Kapitel 7).

Die folgenden Abbildung illustriert die Ergebnisse der beiden Arbeitsschritte[15]. In Phase 1 werden nur Entitäten markiert, die strukturiert vorliegen; die übrigen Passagen sind rot markiert. Im zweiten Schritt werden innerhalb von diesen bestimmte relevante Bestandteile wie Literaturangaben und Sprachbelege aufgefunden.

[14] Dies beinhaltet beispielsweise auch Abschnitte, bei denen fehlerhafte Zeichen zum Zeitpunkt der Erkennung die Struktur stören.

[15] Die vorliegende Art der Illustration, die auch innerhalb des Webportals verwendet wird (vgl. Kap. 12.5) macht sehr deutlich welche Teile des Texts erfasst wurden. Sie kann allerdings im Bezug auf die Unterschiede der beiden Phasen auch irreführend sein, da sie für den ersten Schritt die jeweils tiefsten Einträge einer Baumstruktur markiert, die den entsprechenden Bereich vollständig hierarchisch abbildet, während in der zweiten Phase diese Hierarchien nicht (oder nur in sehr begrenztem Umfang) vorhanden sind. Eine Visualisierung wie Abb. 5.30 in Kap. 5.3.2 verdeutlicht das besser, kann aber aufgrund der sehr komplexen Baumstrukturen nur bei sehr kleinen Teilausschnitten sinnvoll verwendet werden.

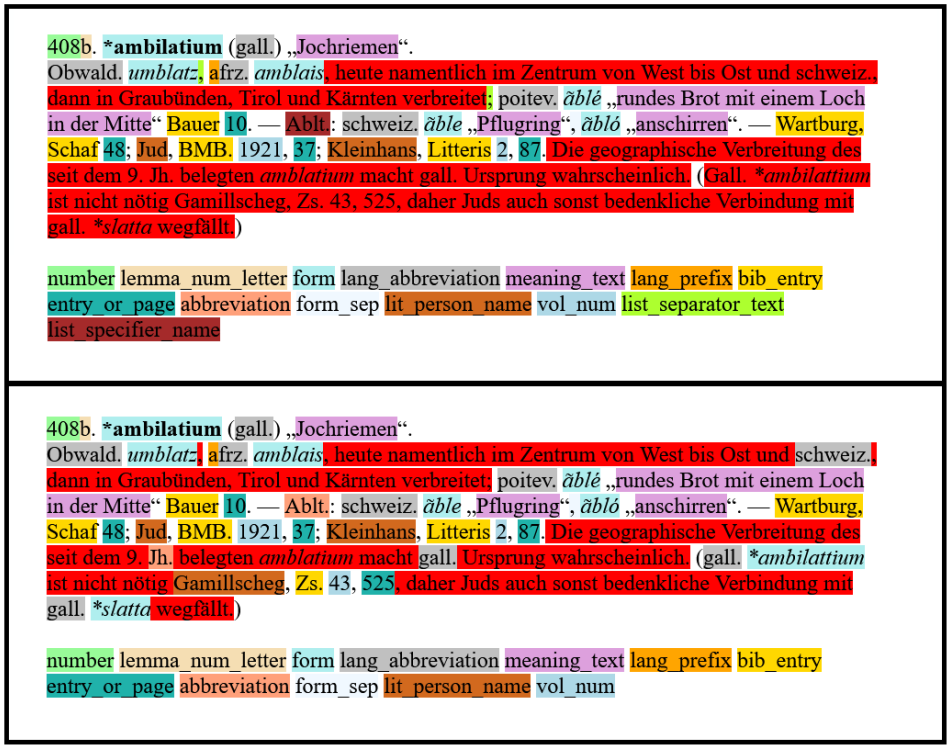


Abbildung 5.15: Ergebnis der zwei Arbeitsschritte der strukturellen Erkennung

Zur Umsetzung von insbesondere Phase 1 bietet sich die Verwendung einer formellen Grammatik an, die den Artikelinhalt beschreibt[16]. Diese wurden ursprünglich als Mittel zur mathematischen Beschreibung der englischen Sprache entwickelt (vgl. Chomsky 1956), sind aber heute in der Computerlinguistik und Informatik weit verbreitet. Vor allem die Grammatiken der Stufen 2 und 3, kontextfreie Grammatiken und reguläre Grammatiken, finden eine breite Anwendung, letztere in vielen Fällen zur Umsetzung von Textersetzungsanwendungen in der Programmierung und erstere unter anderem im Compilerbau zur Verarbeitung von Quellcode (vgl. hierzu z.B. Vossen und Witt 2002). Kontext-freie Grammatiken (oder Abwandlungen davon) wurden auch bereits für die elektronische Verarbeitung von Print-Wörterbüchern eingesetzt (vgl. Kunze und Lemnitzer 2007, S. 94–107, Hauser und Storrer 2017), die resultierenden Ergebnisse aber aufgrund der Inkonsistenzen in traditionellen Wörterbüchern auch durchaus kritisch betrachtet (vgl. Heyn 1992, S. 187–188, Crist 2011, S. 23–25) und werden daher vor allem als „integraler Bestandteil zukünftiger Instruktionsbücher“ (Heyn 1992, S. 192) gesehen, um strukturelle Inkonsistenz künftiger Wörterbuchtexte zu verhindern. Kunze und Lemnitzer 2007 beschreibt in diesem Sinne eine Wörterbuchgrammatik folgendermaßen:

> Wörterbuchartikelgrammatiken definieren Wohlgeformtheitsbedingungen

[16] Phase 2 kann allerdings ebenfalls mit einer solchen umgesetzt werden, vgl. Kap. 5.3.5

> für Wörterbucheinträge und deren Konstituenten. Wörterbuchstrukturen, die nicht durch die Grammatik lizenziert sind, werden als nicht wohlgeformt ausgemustert und markiert.(Kunze und Lemnitzer 2007, S. 94)

So hilfreich eine solche Verwendung auf redaktioneller Ebene ist (und so wünschenswert strukturelle Konsistenz innerhalb eines Wörterbuchs ist) steht eine solche strikte Formulierung einer statischen Grammatik der Verwendung mit gerade älteren, weniger formalisieren Werken eher im Weg. Das bereits beschriebene zweistufige Verfahren hat aus dieser Perspektive gesehen schon einmal den Vorteil, dass eine „Ausmusterung“ nicht auf Artikelebene stattfinden muss, sondern sich maximal auf einzelne Artikelbestandteile bezieht. Des weiteren unterscheidet sich aber auch die zugrundeliegende Konzeption, die die Datenerstellung nicht als einmaligen Prozess auffasst, sondern als inkrementelles und dynamisches System, in dem die Datenqualität (und -quantität) ständig verbessert werden kann. Die Verwendungsweise einer formellen Grammatik, die in den folgenden Abschnitten konzipiert wird, orientiert sich somit stark an den in Kap. 3.1 festgelegten Grundsätzen.

5.2.1 Parsing Expression Grammars

Ein Parser dekonsturiert einen Text anhand der definierten Regeln in seine Bestandteile. In der Praxis erweist sich dabei das generative Modell der Chomsky-Grammatiken, das als Möglichkeit zur Beschreibung von formellen Sprachen und weniger als Methodik zum Parsen von bestehenden Texten konzipiert wurde, zum Teil als unpraktisch. Vor allem die Tatsache, dass explizit Mehrdeutigkeiten vorgesehen sind, steht einer Anwendung im Weg. Traditionell wird dies zum großen Teil durch die Verwendung von sogenannten *LR-Grammatiken* (vgl. z.B. Chapman 1987) gelöst, die eine Untermenge der kontext-freien Sprachen bilden. Da diese bestimmte Anforderungen an die Regeln stellen und deren Einhaltung gerade bei komplexen Grammatiken nicht immer trivial ist, wurden in Ford 2004 die sogenannten *Parsing Expression Grammars (PEG)* entworfen. Diese ersetzten die gleichwertigen Alternativen der klassischen kontextfreien Grammatiken durch eine sogenannte „prioritized choice“ (Ford 2004, S. 1), die die Varianten von links nach rechts abarbeitet und die erste gültige auswählt. Gerade für die einfache automatisierte Integration von Listen von Literalen ist dieses Verhalten hilfreich (vgl. Kap. 5.2.3).

Ein weiterer Unterschied ist die Einführung von sogenannten *Lookahead assertions*, wie sie in der praktischen Umsetzung von Parsern für reguläre und kontextfreie (vgl. z.B. Parr und Quong 1995) Sprachen häufig vorkommen. Syntaktisch wird dies durch die Verwendung zweier Operatoren *AND (&)* und *NOT (!)* dargestellt. Sie stellen sicher, dass der Text nach der aktuellen Position einem bestimmten Muster entspricht bzw. nicht entspricht, ohne die Position selbst zu verändern. Umgekehrte *Lookbehind Assertions*, die die Zeichen vor der aktuellen Position überprüfen, sind in der Definition der PEG nicht enthalten s. Kap. 5.2.2).

Zum Verständnis der Beispiele in den folgenden Kapiteln wird hier kurz die grundsätzliche Syntax zusammengefasst. Sie entspricht bis auf die beiden eben erwähnten Operatoren den klassischen Konventionen. Jede Regel wird durch ein ein Nichtterminal[17] auf der linken Seite und einen Ausdruck aus Nichtterminalen und Terminalen[18] auf der rechten Seite definiert. Dabei gibt es die folgenden Operatoren

Leerzeichen	Sequenz
/	Alternative
*	Beliebige Anzahl [0,n]
+	Mindestens einmal [1,n]
?	Optional [0,1]
{m,n}	Wiederholung mit festen Grenzen [m,n]
&	Positiver Lookahead
!	Negativer Lookahead

Tabelle 5.5: Syntax der Operatoren in der PEG

und folgende Möglichkeiten Zeichen zu beschreiben

” ”	Zeichenkette, z.B. ”Beispiel”
[ ]	Zeichenklasse, z.B. [a-z] für Kleinbuchstaben oder [^a-z] für Zeichen, die keine Kleinbuchstaben sind
.	Beliebiges Zeichen

Tabelle 5.6: Notationen für die Beschreibung von Zeichen in der PEG

Eine einfache Beschreibung einer Liste von einer oder mehreren Bedeutungen (in der Formatierung des REW) könnte somit folgendermaßen dargestellt werden:

```
meaning_list   <-   meaning (", " meaning)*
meaning        <-   "„" [A-Za-z ] """
```

5.2.2 Erweiterungen der PEG

Auch wenn der Formalismus der PEG sich sehr gut für die intuitive Erstellung von Parsern für komplexe Grammatiken eignet, gibt es bestimmte Spezialfälle, deren Darstellung sehr aufwendig ist und zu fehleranfälligen Regeln führt. Deshalb werden hier zwei Erweiterungen vorgeschlagen, die solche Fälle vereinfachen.

Wie bereits erwähnt enthalten PEG zwar die Möglichkeit sogenannte *Lookahead Assertions* einzufügen aber nicht den umgekehrten Fall der *Lookbehind Assertions*, die

[17] Ein Nichtterminal kann gerade im erkennungsbasierten System (vgl. Ford 2004, S. 1) der PEG prinzipiell als Identifikator für eine bestimmte Regel aufgefasst werden.

[18] Terminale sind Literale, also beispielsweise Zeichenketten

zusätzlich ergänzt wurden[19]. Dies ist entscheidend für das in Kap. 5.2.6 vorgeschlagene Ausnahmensystem, hat aber auch sonst Vorteile in der Abbildung bestimmter Spezialfälle. So lassen sich manche typographische Konventionen ohne solche nur sehr schwer umsetzen. Ein typisches Beispiel hierfür ist die Verwendung von Punkten zum Abschluss von Sätzen oder ähnlichem. Gerade in älteren Wörterbüchern findet aufgrund der Platzbeschränkungen und der Kosten des Drucks eine häufige Verwendung von Abkürzungen statt. Wenn nun beispielsweise ein Satz auf eine Abkürzung endet, wird aus optischen Gründen kein zweiter Punkt gesetzt. Für die strukturelle Erfassung mit Hilfe einer Grammatik ist diese Konvention allerdings problematisch. Beispielsweise könnte ein Satz stark vereinfacht durch folgende Regel beschrieben werden[20]:

```
sentence   <-   token (" " token)* "."
token      <-   word / abbreviation
```

Für den folgenden Beispielsatz würde eine solche Regel allerdings fehlschlagen:

> Rum. crap ist slav.

Die Verarbeitung würde korrekt bis zur Abkürzung „slav.“ verlaufen, dann wären allerdings bereits alle Zeichen konsumiert, sodass der Satz als ungültig behandelt würde, da der abschließende Punkt fehlt. Ein eleganter Lösungsansatz lässt sich durch die erwähnten *Lookbehind Assertions* erstellen[21]:

```
sentence <- token (" " token)* pot_dot
token    <- word / abbreviation
pot_dot  <- <&"." / "."
```

Die Regel *pot_dot* definiert somit also ein Zeilenende damit, dass entweder das vorherige Zeichen schon ein Punkt war oder, falls dies nicht der Fall ist, als expliziten Punkt. Sie entspricht somit exakt dem Vorgehen, welches auch eine menschliche Person intuitiv verwenden würde.

Die zweite Erweiterung ist die Einführung von sogenannten *Makros*. Grundsätzlich beschreiben diese in der Programmierung Code-Bestandteile, die an verschiedenen Stellen eingefügt und damit wiederverwendet werden können. Eine Anwendung mit formellen Grammatiken findet sich beispielsweise in (Thiemann und Neubauer 2008).

[19] Um den Implementierungsaufwand im Rahmen zu halten, wurden diese nur für Literale und Optionslisten von Literalen implementiert

[20] Die Regel ist aus verschiedenen Gründen so nicht praxistauglich, vor allem weil so keine sinnvolle Erkennung von Satzenden möglich ist, und dient nur zur Illustration.

[21] Hier wird die Darstellung <& für eine positive *Lookbehind Assertion* verwendet. Entsprechend beschreibt <! eine negative *Lookbehind Assertion*.

An dieser Stelle werden sie zur einfachen Parametrisierung von Grammatik-Regeln verwendet. Ein Beispiel ist die Konvention im REW bestimmte Bedeutungen für Lemmata mit einem Sternchen (vor oder nach dem einführenden Anführungszeichen) als *rekonstruiert* zu markieren. Um diesen Spezialfall abzubilden, müssten zwei Sätze von Regeln für Bedeutungen für Lemmata und für andere sprachliche Formen konstruiert werden:

```
meaning_list_head    <-    meaning_head (", " meaning_head)*
meaning_head         <-    "*"? "„" "*"? german_sentence """

meaning_list_body    <-    meaning_body (", " meaning_body )*
meaning_body         <-    "„" german_sentence """
```

Diese Darstellung ist in hohem Maße redundant und somit fehleranfällig, da eventuelle Änderungen an mehreren Stellen vorgenommen werden müssten. Mit Hilfe der parametrisierten Regeln kann dies ohne Redundanz dargestellt werden[22]:

```
meaning_list($recon)    <-   meaning($recon) (", " meaning($recon))*
meaning($recon)         <-   $recon? "„" $recon? german_sentence """
```

Somit können je nach Kontext die Regeln *meaning_list(„*“)* oder *meaning_list(nothing)*[23] verwendet werden.

Anmerkung zur Implementierung: Im konkreten Fall wurde die Bibliothek *PHP PEG Parser*[24] für die Umsetzung eines PEG-Parsers verwendet. Deren Umsetzung ist sehr einfach, was den Vorteil hat, dass Erweiterungen sehr leicht hinzugefügt werden können, anderseits aber den Nachteil, dass das Parsen der Texte nicht besonders effizient ist. In diesem Fall wurde die Makros somit direkt in den Parser eingebaut und innerhalb des Parse-Vorgangs umgesetzt. Bei einer effizienteren Implementierung (vgl. z.B. Ford 2002, Moss 2017) wäre es wohl grundsätzlich sinnvoller die Makros mehr im eigentlichen Sinne zu verwenden, d.h. aus der erweiterten Grammatik wird eine reguläre PEG erstellt und dann geparset. In diesem Fall würde es sich dann auch streng genommen nicht um eine Erweiterung des eigentlichen Grammatikformalismus handeln.

[22] Die Parameter werden hier in Anlehnung an Variablen in der Programmiersprache PHP mit einem $-Präfix markiert.
[23] Die Regel *nothing* ist eine spezielle Regel, die niemals gültig ist, vgl. auch Kap. 5.2.6
[24] https://github.com/wouterj/peg

5.2.3 Dynamische formelle Grammatiken

Zu Beginn dieses Kapitels wurde bereits die Problematik einer statischen Grammatik im Bezug auf das hohe Maß an Inkonsistenzen in traditionellen Werken angesprochen. Diese wirft vor allem drei Fragestellungen auf:

- Wie detailliert kann/soll eine Grammatik sein, d.h. welche Notationen oder Sonderfälle werden noch umgesetzt und welche nicht?
- Wie können die Fälle, die nicht abgebildet werden in das Gesamtsystem integriert werden, so dass trotzdem ein Maximum der restlichen Information erfasst werden kann?
- Wie können vereinzelte Fälle von falsch positiven oder falsch negativen Erkennungen der Grammatik behandelt werden, ohne die Grundstruktur der Grammatik zu verändern

Eine Antwort auf die erste Frage liefert vor allem das in Kap. 3.1 vorgestellte iterative Vorgehen, in dem ein gewisser Anteil des Quellenmaterials sehr intensiv analysiert und korrigiert wird. Zu Beginn kann mit einer „Basisgrammatik“ begonnen werden, die aus der grundsätzlichen Analyse der Quelle entstanden ist. Im weiteren Vorgehen können dann Einträge, die Notationen enthalten, die durch die aktuelle Grammatik nicht abgebildet werden können, übersprungen werden. Dabei sind vor allem die Auswahl der Stichprobe und die Markierung des Grunds für die Überspringung entscheidend.

Eine über verschiedene Abschnitte ausgewählte Stichprobe ist dabei grundsätzlich sinnvoller als ein fortlaufender Abschnitt, da die Wahrscheinlichkeit höher ist, dass alle relevanten Notationen entdeckt werden. Gerade bei Werken wie dem REW, die von Einzelpersonen verfasst wurden und bei denen es keine redaktionellen Vorgaben oder ähnliches gab, kann es durchaus vorkommen, dass bestimmte Notationen erst im hinteren Teil der Quelle genutzt werden bzw. bestimmte Notationen ab einem gewissen Punkt aufgegeben oder durch andere ersetzt werden[25].

Die Angabe eines Grunds für das Überspringen eines Artikels dient dazu eine quantitative Analyse auf Basis von diesen durchführen zu können. Denkbar dafür sind verschiedene Kriterien, beispielsweise die explizite Behandlung einer Konvention in der Grammatik, wenn ein bestimmter Grenzwert oder ein bestimmter Anteil überschritten wird. Bei Fällen, in denen dies nicht bereits im Verlauf der Korrekturphase der Fall ist, kann an deren Ende ein Überblick über die Anzahlen bestehender Spezialfälle gewonnen werden und auf Basis dieser Daten eine Entscheidung getroffen werden, welche zusätzlichen Regeln noch hinzugefügt werden sollten.

[25] Zum Beispiel tritt im REW die spezielle Notation, dass verschiedene anderssprachige Formen aus verschiedenen Bedeutungen einer Ursprungsform entlehnt werden, zum ersten Mal in Eintrag 1297 (von 9721) auf.

Die Beantwortung der zweiten Frage nach dem Umgang mit Notationen, die nicht durch die Grammatik abgebildet werden, wurde bereits in den vorherigen Kapiteln angerissen. Der Lösungsansatz ist hierbei die Trennung zwischen strukturierten Passagen und unstrukturieren bzw. natürlichsprachlichen Abschnitten (vgl. hierzu auch die allgemeinen Betrachtungen in Crist 2011, S. 25, 26). Ein Abschnitt der aufgrund eines Spezialfalls nicht abgebildet werden kann, landet somit automatisch in der zweiten Kategorie[26]. Was noch nicht besprochen wurde, ist die Umsetzung einer solchen Trennung innerhalb einer formellen Grammatik. Diese hängt eng mit der dritten Frage nach dem Umgang mit falsch erkannten Passagen zusammen. Gerade beim Vorhandensein von unstrukturierten Bestandteilen innerhalb strukturierter Abschnitte ist die Wahrscheinlichkeit inkorrekter Erkennungen besonders hoch. Das ist vor allem deswegen der Fall, weil es kaum möglich ist gerade unstruktrierte Einschübe so zu modellieren, dass sie sicher erkannt werden können, weil sie auf syntaktischer Ebene oftmals nicht von anderen Konstrukten unterschieden werden können. Eine fortlaufende Abwandlung der Grammatik, um auf neu auftretende Spezialfälle zu reagieren, ist im Normalfall wenig sinnvoll, da eine solche Vorgehensweise zu sehr komplizierten Regeln führt, die kaum überschaubar sind und wiederum an anderen Stellen zu falschen Erkennungen führen können. Entscheidend ist eher eine grundsätzliche Variabilität der Grammatik, um mit dem ebenfalls sehr variablen und unterschiedlichen Aufbau von Wörterbuchartikeln umzugehen. Eine Grammatik, die als eine Menge von Regeln aufgefasst werden kann, wird um diese zu erreichen aus vier Teilmengen[27] zusammengesetzt:

- Die grundsätzlichen Strukturregeln, die nach der intellektuellen Analyse der Quelle manuell erstellt werden;
- Listen von Literalen, die für die Erkennung der Struktur notwendig sind, die allerdings wegen ihrer Sprach- und Quellenabhängigkeit nicht Teil der Kernregeln sein sollten;
- Listen von Literalen, die (Teil-)Ergebnisse von vorherigen Schritten im Importprozess (vgl. Kap. 3.1) sind (z.B. bibliographische Abkürzungen);
- Lokale und globale Ausnahmen auf Grammatikebene (vgl. Kap. 5.2.6).

Alle diese Regeln sind als einzelne Datensätze vorhanden, was den Vorteil hat, dass aus ihnen entsprechend des jeweiligen Artikelkontexts und der Verarbeitungsphase (vgl. Kap. 5.2.4) die notwendige Grammatik zusammengesetzt werden kann.

Die erste Kategorie bildet das „Skelett“ der Formalisierung des Texts und besteht aus abstrakten Entitäten, die die grundsätzlichen Strukturelemente und deren Anordnung

[26] Es ist natürlich immer möglich zu einem späteren Zeitpunkt die Grammatik zu erweitern und diese Fälle strukturiert zu erfassen.

[27] Zusätzlich zu diesen wird ebenfalls eine Liste aller Zeichen verwendet, die in sprachlichen Formen vorkommen können.

darstellen, ohne konkret deren Ausgestaltung zu definieren In dieser wird also auf Literale verzichtet, die über sehr simple Strukturbestandteile wie Punkte oder Leerzeichen hinausgehen. Regeln wie

```
meaning_list <-
     "id."
     / "ebenso"
     / meaning ((", " / ", eigentlich " / ", auch " / " und ") meaning)*
```

werden also auch in einfach Fällen konsequent umstrukturiert zu

```
meaning_list <-
     special_meaning_literal
     / meaning (meaning_sep_literal meaning)*

special_meaning_literal  <- "id." / "ebenso"
meaning_sep_literal      <- ", " / ", eigentlich " / ", auch " / " und "
```

Die letzten beiden Regeln werden nicht explizit in dieser Form in der Datenbank abgelegt, sondern aus einer eigenen Tabelle generiert, die die Literale aus Kategorie 2 enthält. Das hat einerseits den Vorteil, dass die Literallisten leicht erweitert werden können, ohne die eigentlichen Strukturregeln verändern zu müssen, andererseits kann so eine eventuelle Nachnutzung deutlich erleichtert werden. Auch wenn nur eine einziges Literal innerhalb von einer bestimmten Regel verwendet wird, wird für dieses eine eigene Regel erstellt.

Bemerkung zur Nachnutzung: In der Praxis weichen wohl auch auf den ersten Blick ähnlich wirkende Quellen zu stark voneinander ab, als dass realistisch davon ausgegangen werden kann, dass selbst die Strukturregeln aus Kategorie 1 in dieser Form direkt wiederverwendet werden können. Trotzdem hat dieses Modell den Vorteil, dass zumindest deutlich einfacher aus einer bestehenden Menge von Regeln durch bestimmte Hinzufügungen, Auslassungen und Abwandlungen Grammatiken für weitere Quellen erstellt werden können, da die konkrete Formatierung der Quelle fast komplett aus dem Modell entfernt wird.

Ein Beispiel für eine solche Liste soll hier mit dem Präfix für den Herausgeber eines Bandes in der Bibliographie des REW genannt werden. Da gerade diese besonders inkonsistent formatiert ist, treten eine Vielzahl von sprachlichen und notationellen Varianten auf:

id_literal	value
390	hg. von
405	jetzt hg. von
408	hgg. von
411	jetzt hgg. von
414	herausgeg. von
417	ordinati da
420	e ordinati da
423	p. de
426	p. d.
429	p. p.
432	ed.
435	ed. by
438	dirigée par
441	editors:
444	d. p.
447	diretti da
450	direttori:
453	par MM.

Diese Tabelle kann dann beim Auffinden eventueller zusätzlicher Varianten leicht erweitert werden, ohne die strukturelle Grammatik anpassen zu müssen.

Die Regeln der Kategorie 3 unterscheiden sich formell nicht von denen der zweiten Kategorie, der einzige Unterschied ist deren Herkunft aus vorherigen Importprozessen. Sie sind damit noch quellenspezifischer als die anderen, da es sich hierbei im Normalfall um konkrete Abkürzungen handelt, die in der Quelle genannt werden[28]. Alle solchen Literallisten müssen grundsätzlich absteigend sortiert werden, da es häufig Überschneidungen zwischen Abkürzungen bzw. verschiedenen Varianten von Abkürzungen gibt. So würde aufgrund der „priotized choice“ der PEG bei einer anderen Sortierung unter Umständen den Bestandteil „bress.“ der Abkürzung „bress.-louh.“ bereits als vollständige Abkürzung erkannt werden, da diese auch existiert.

Die Elemente der letzten Kategorie sind keine eigentlichen Regeln, sondern Anpassungen bestehender Regeln in konkreten Kontexten (im Normalfall auf Artikelebene) und werden in Kap. 5.2.6 ausführlich beschrieben.

[28] Die Listen sind allerdings nach dem initialen Import nicht annähernd vollständig und müssen im Laufe der Verarbeitung ständig aktualisiert werden (vgl. Kap. 12.5)

5.2.4 Modularer Grammatikaufbau

Ein weiterer Vorteil der im letzten Kapitel vorgestellten teilautomatisierten Grammatikerstellung aus verschiedenen Datenbanktabellen ist eine erleichterte Modularisierungbarkeit der strukturellen Erfassung. Damit ist die Verwendung von verschiedenen Grammatiken gemeint, die nacheinander angewandt werden, anstatt den vollständigen Text durch eine globale Grammatik zu parsen. Da die Regeln als einzelne Einträge in der Datenbank vorliegen, können sie sehr einfach in verschiedenen Grammatiken wiederverwendet werden. Insbesondere muss eine Grammatik nicht über eine fixe Menge von Regel definiert werden, sondern kann aus der Hauptregel über die jeweils referenzierten Regeln rekursiv erstellt werden. Die Modularisierung an sich hat verschiedene Vorteile.

Einerseits verringert es die Komplexität der einzelnen Teilgrammatiken, so dass diese übersichtlicher bleiben und eventuelle Fehler leichter aufgefunden werden können. Zum anderen kann es aber auch Vorteile im Prozessablauf der Quellenerschließung mit sich bringen. Trotz aller im vorherigen Kapitel vorgestellten Maßnahmen kann zum Beispiel nicht ausgeschlossen werden, dass einzelne Artikel komplett ausgemustert werden und nicht einmal in Teilen von der Grammatik erfasst werden können. Das ist vor allem dann der Fall wenn an ungünstigen Stellen[29] im Text bestimmte entscheidende Strukturelemente fehlerhaft sind (beispielsweise öffnende Klammern oder ähnliches). Die Basisinformation, die für die Erstellung eines Artikels im REW benötigt wird, ist aber zumindest die Nummer des jeweiligen Eintrags. Falls dieser nicht geparst werden kann, kann im Importvorgang also nicht einmal ein Platzhalter erstellt werden. Wenn die Kopfzeile unabhängig vom restlichen Artikeltext zuerst strukturiert erfasst wird, kann das Auftreten dieses Problems zumindest stark reduziert werden. Das heißt das Kriterium ist nur noch, dass die Kopfzeile des Artikels erfasst werden können muss, die strukturell sehr viel einfacher ist. Der Rest des Artikeltexts kann dann im Extremfall als reiner unverarbeiteter Textbestandteil für einer erste Artikelversion importiert werden. Auch die grundlegende Anwendung des zweistufigen Erfassungsmodells (vgl. Kap. 5.2) gestaltet sich so einfacher.

Regeln können, wie bereits erwähnt, leicht wiederverwendet werden (so kommen z.B. Bedeutungsangaben im Artikelkopf und im Artikelinhalt vor), sie können aber auch je nach Kontext „überladen“ werden, d.h. es wird je nach Grammatik für eine bestimmte Regel ein anderer Ausdruck verwendet[30]. Besonders nützlich ist dies bei der Trennung

[29] Diese „ungünstige Stellen“ hängen stark vom konkreten Aufbau der Grammatik ab. Das Modell, das in Kap. 5.3.3 beschrieben wird, vermeidet solche Fälle größtenteils, indem nicht erkannte Bestandteile sich im Normalfall nur auf Satzebene auswirken und nicht auf den vollständigen Text. Trotzdem können sie nie vollständig ausgeschlossen werden, wenn beispielsweise schon die Erkennung der Satzgrenzen scheitert.

[30] Dies würde prinzipiell auch mit dem in Kap. 5.2.2 vorgestellten Makrosystem möglich sein, aber gerade bei weitreichenden Abwandlungen kann die Erstellung eigener Regeln nützlich sein. In diesem Fall wird natürlich wieder eine gewisse Redundanz erzeugt. Oftmals muss man deshalb im Einzelfall die Vor- und Nachteile beider Methoden abwägen.

der beiden Phasen der strukturierten Erfassung und der *Entity Recognition*, wobei sich zwar viele Bestandteile überschneiden, aber nicht genau gleich behandelt werden. Insbesondere werden einige Einzelbestandteile durch die immer ungültige Regel *nothing* ersetzt, da sie nicht gebraucht werden (vgl. Kap. 5.3.5).

Eine andere Möglichkeit, die sich durch die Modularisierung ergibt, ist bestimmte Bestandteile nicht von links nach rechts, sondern umgekehrt von rechts nach links zu verarbeiten. In manchen Fällen macht es die rein vorwärtsgerichtete Arbeitsweise eine formellen Grammatik schwierig bestimmte Notationen korrekt zu erkennen. Das ist beispielsweise bei der Bibliographie des REW der Fall. Die Einträge haben dabei einen klaren Rahmen: Sie beginnen mit einer Abkürzung und einem = und enden mit einem Ort und einer Jahreszahl, die durch einen Punkt getrennt werden (jeweils bis auf wenige Ausnahmen):

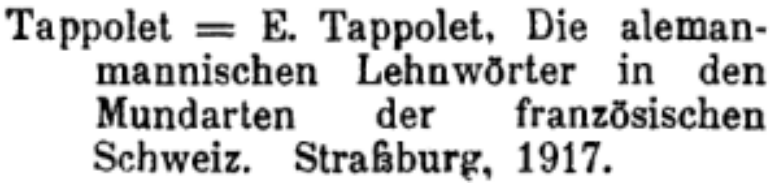

Tappolet = E. Tappolet, Die alemannischen Lehnwörter in den Mundarten der französischen Schweiz. Straßburg, 1917.

Abbildung 5.16: Einfacher Eintrag aus der Bibliographie (REW, S. XXVI)

Da die bibliographischen Angaben aber selbst des öfteren Punkte enthalten, ist es zum Teil bei einer Reihenfolge von links nach rechts schwer zu entscheiden, ob die Angabe des Ortes erreicht ist oder nach dem Punkt nur ein (optionaler) Untertitel des Werks folgt[31]. Eine einfache Lösung des Problems ist eine Aufteilung des Erkennungsvorgangs auf drei verschiedene Grammatiken. Zuerst wird der vordere Anteil bis zum Trennzeichen erfasst und der Rest als unstrukturierter Text aufgefasst, im zweiten Schritt wird der hintere Anteil entsprechend erkannt und im letzten Schritt der restliche Text in der Mitte:

Abbildung 5.17: Schematische Verarbeitungsreihenfolge bei der Erfassung der Bibliographie

Die Angabe der andersgerichteten Grammatik kann dabei in der üblichen Form erfolgen, wobei die Regeln automatisiert „gespiegelt“ und auf den ebenfalls umgekehrten Text angewandt werden.

[31] Eine Behandlung in Links-Rechts-Reihenfolge ist nicht unmöglich, nur aufwendig und potentiell unübersichtlich. Es wäre z.B. eine Lösung über *Lookahead Assertions* oder sogar über die Führung einer Liste von Ortsnamen grundsätzlich vorstellbar.

5.2.5 Ergebnisformat der strukturellen Erkennung

Ein Parser für eine formelle Grammatik verarbeitet den Text und zerlegt ihn anhand der festgelegten Regeln in die einzelnen Bestandteile. Diese können im Verlauf des Vorgangs je nach Anwendungsfall weiterverarbeitet werden. Um die Komplexität des zugrundeliegenden Programmcodes zu verringern, wird an dieser Stelle lediglich eine abstrakte Repräsentation in einem festgelegten Austauschformat erzeugt, welche als Basis der nächsten Prozessphase Kapitel 7 verwendet wird. Dieses enthält für jeden Knoten der Baumstruktur die folgenden Informationen:

- **string**: Die vollständige Zeichenkette, aus der dieser Knoten erzeugt wurde
- **id**: Der Name der angewandten Regel
- **start**: Beginn der Zeichenkette im Gesamttext
- **end**: Ende der Zeichenkette im Gesamttext
- **data**: Die jeweiligen Unterelemente. Für die Endknoten des Baumes fehlt dieses Element.

Das folgende Beispiel zeigt dies für Artikel 4363 des REW[32] im JSON-Format:


```
{
   "string": "4363. <b>incrĕscĕre<\/b> ...",
   "id": "entry",
   "start": 0,
   "end": 324,
   "data": {
      "head": {
         "string": "4363. <b>incrĕscĕre<\/b> ...",
         "id": "head",
         "start": 0,
         "end": 70,
         "data": [...]
      },
      "parts": {
         "string": "1. Mazed. <i>ncrişteare<\/i> ...",
         "id": "parts",
         "start": 76,
         "end": 324,
         "data": [{
```


[32]Für die zugrundeliegende Modellierung s. Kap. 5.3.2

```
            "string": "1. Mazed. <i>ncrişteare<\/i> ...",
            "id": "numbered_part_first",
            "start": 76,
            "end": 155,
            "data": [...]
         }, {
            "string": "2. It. <i>rincrescere<\/i> ...",
            "id": "numbered_part",
            "start": 161,
            "end": 324,
            "data": [...]
         }]
      }
   }
}
```

Für das *data*-Attribut gibt es dabei zwei Möglichkeiten. Im ersten Fall enthält es eine Liste mit Namen und Informationen der jeweiligen Unterelemente. Zum Beispiel besteht ein Eintrag aus den Unterelementen *head* und *parts*. Die Unterelemente werden dabei wieder im gleichen Format angegeben. Die zweite Variante ist eine einfache Liste von Unterelementen (im Beispiel mit eckigen Klammern angegeben). Dies wird für rein sequentielle Elemente verwendet, wie sie beispielsweise mit Hilfe des *-Operator erzeugt werden können. Diese Regeln haben in der Praxis meist die folgenden Form:

```
rule   <-   element (separator element)*
```

So ist es auch in diesem Beispiel, in dem die Regel *parts* aus einer Liste von Abschnitten besteht (vgl. Kap. 5.3.2).

Zu beachten ist hierbei, dass dieses Resultatformat leichte Einschränkungen an die Formulierung der Regeln der Grammatik stellt. So sollten Listen mit beliebiger Elementzahl als eigene Regel modelliert werden. Der Vorteil ist hierbei, dass sowohl die Grammatik selbst als auch das Ergebnis der Erfassung eine sehr klare Struktur aufweisen.

Die Verwendung der Start- und Endindexe hat den Grund, dass diese Information vor allem bei der Korrektur von Fehlern hilfreich ist. So kann beispielsweise für alle Sprachbelege ihre Position innerhalb des Ursprungstext in der Datenbank gespeichert werden. Wird beispielsweise mit der Methodik in Kap. 8.2.2 eine inkorrekte Form aufgefunden, kann mit Hilfe diese Information die genaue Zeile (bzw. die Zeilen) angezeigt werden, aus der die Form ursprünglich stammt, was die Korrektur im Gegensatz zur Alternative, bei der alle Zeilen des Artikels angezeigt werden müssten, deutlich vereinfacht.

5.2.6 Ausnahmen auf Grammatikebene

Die Ausnahmen für die strukturelle Erfassung werden wie in Kap. 5.2.3 erwähnt in die aktuelle Grammatik integriert, d.h. es werden je nach Kontext zusätzliche Regeln eingebaut. Dabei verwenden die Ausnahmen ausschließlich Zeichenketten, d.h. es finden keine komplexeren Regelveränderungen statt, sondern einzelne Regeln werden durch Literale ergänzt.

Unterschieden werden positive und negative Ausnahmen, die entsprechend die beiden Arten von Fehlern abdecken. Eine positive Ausnahme wird als Optionsliste konstruiert, die um die eigentliche Regel herum eingefügt wird. Wenn die ursprüngliche Regel beispielsweise diese Form hätte, die (stark vereinfacht) eine kursive Form definiert

```
form_it    <-   "<i>" [a-z] "</i>"
```

und als Ausnahme die nicht diesem Muster entsprechende Form Olivar hinzugefügt wird, würde folgende neue Regel entstehen.

```
form_it    <-    "<i>Olivar</i>" / ("<i>" [a-z] "</i>")
```

Aufgrund der Erkennungsreihenfolge einer PEG wird die Ausnahme prinzipiell zuerst überprüft und erst danach die eigentliche Regel. Werden mehrere Ausnahmen auf einmal eingefügt, werden diese absteigend alphabetisch gelistet eingefügt. Eine weitere Ausnahme Olival würde somit diese Regel erzeugen:

```
form_it    <-    "<i>Olivar</i>" / "<i>Olival</i>" / ("<i>" [a-z] "</i>")
```

Die negative Ausnahme hat eine etwas kompliziertere Gestalt und basiert auf den bereits beschriebenen *Lookahead* und *Lookbehind Assertions*. Das Einfügen einer negativen Ausnahme für die Zeichenkette „a“ in die gleiche Regel würde folgende Veränderungen bewirken:

```
form_it    <-    "<i>" [a-z] "</i>" !<"<i>a</i>"
           /     !"<i>a</i>" "<i>" [a-z] "</i>"
```

Die Regel wird also in zwei Optionen aufgeteilt, die zusammen die logische Aussage bilden, dass entweder der durch die Regel abgedeckte Text nicht mit der Ausnahme beginnt oder nicht mit der Ausnahme endet. Bei Fälle sind notwendig, da bei ausschließlich der ersten Option alle längeren Zeichenketten ausgeschlossen würden, die

mit der Ausnahme beginnen und bei ausschließlich der zweiten Option alle solchen, die mit der Ausnahme enden. Mehrere negative Ausnahmen werden sequentiell nacheinander dargestellt, was in diesem Fall einem logischen *Und* entspricht

```
form_it    <-    "<i>" [a-z] "</i>" !"<i>b></i>" !"<i>a</i>"
           /     !"<i>b></i>" !"<i>a</i>" "<i>" [a-z] "</i>"
```

Beim Einfügen der Ausnahmen wird die Reihenfolge angewendet, dass zuerst die positiven Ausnahmen und erst danach die negativen Ausnahmen eingefügt werden:

```
form_it    <-    (<i>Olivar</i> / "<i>" [a-z] "</i>") !"<i>a</i>"
            /    !"<i>a</i>" (<i>Olivar</i>  / "<i>" [a-z] "</i>")
```

Die negative Regel hat somit Vorrang und kann gewissermaßen eine gleichlautende Positivausnahme ungültig machen. In der Praxis ist diese Reihenfolge allerdings eher nicht entscheidend, da eine Überschneidung von positiven und negativen Ausnahmen für den gleichen Textbereich im Normalfall nicht sinnvoll ist.

Ein grundlegender Unterschied zwischen positiven und negativen Ausnahmen, ist, dass erstere ausschließlich auf Regeln angewandt werden können, die als Resultat eine eine einfache Zeichenkette haben (vgl. Kap. 5.2.5), während letztere uneingeschränkt genutzt werden können. Dies ist so weil bei positiven Ergebnissen ein Eintrag im Resultatbaum angelegt wird. Wenn dieses allerdings auf Basis mehren Teilregeln zusammengesetzt werden muss, können diese aus der Zeichenkette der Ausnahme nicht bestimmt werden, da diese eben nicht durch die reguläre Regel dekonstruiert werden kann. Durch die Aufteilung der Regeln in Strukturbestandteile und Listen von Literalen (vgl. Kap. 5.2.3) spielt diese Einschränkung in der Praxis allerdings keine große Rolle, da Positivausnahmen im Normalfall sowieso nur bei Literalen sinnvoll sind und diese grundsätzliche in eine eigene Regel eingebettet sind.

Der Gültigkeitsbereich einer jeden Ausnahme kann unterschiedlich festgelegt werden. Grundsätzlich werden zwei Hauptgruppen unterschieden: Globale und lokale Ausnahmen. Erstere gelten grundsätzlich immer innerhalb einer bestimmten Gruppe von Teilgrammatiken, während letztere an eine konkrete Zeile gebunden sind und nur für den Eintrag verwendet werden, in dem diese Zeile vorkommt[33].

Globale positive Ausnahmen sind streng genommen nicht notwendig und könnten auch direkt in die jeweiligen Grammatikregeln als Liste von Literalen eingebaut werden. Zum Teil bleiben die Strukturregeln allerdings übersichtlicher, wenn abweichende Fälle in dieser Weise modelliert werden. Tendenziell sind globalen Ausnahmen aber negativ und geben beispielsweise an, dass bestimmte Abkürzungen im Textkontext nicht als

[33] Wenn die Ausnahme über mehrere Zeilen geht, wird die erste verwendet.

solche interpretiert werden soll. So gibt es im REW beispielsweise die literarischen Abkürzungen „Zweifel“ oder „Einführung“, die allerdings allein (d.h. ohne Zeilennummer oder ähnliches) im Textkontext nicht als bibliographische Angabe interpretiert werden sollen, sondern als das deutsche Wort.

Lokale Ausnahmen nehmen den ganz überwiegenden Teil der Gesamtzahl der Grammatikausnahmen ein: 3123 (Stand 01.09.2023) von 3147 (Stand 01.09.2023) Einträgen. Diese können größtenteils ohne Probleme auf den gesamten Artikel angewandt werden, dem sie zugeordnet sind. In wenigen Spezialfällen kommen bestimmte Zeichenketten allerdings mehrmals vor, wohingegen sich die Ausnahme nur auf eines des Vorkommen beziehen soll[34]. In diesem Fall kann zusätzlich ein Kontext, der auf die entsprechende Ausnahme folge, angegeben werden, um diese genauer einzugrenzen. Er wird als zusätzlicher positiver *Lookahead* bei positven Ausnahmen und als negativer *Lookahead* bei negativen Ausnahmen in die Regel eingebaut:

```
form_it    <-    "<i>Olivar</i>" &"Kontext" / ("<i>" [a-z] "</i>")
```

bzw.

```
form_it    <-    "<i>" [a-z] "</i>" (!<"<i>a</i>" / !"Kontext")
            /    !"<i>a</i>Kontext" "<i>" [a-z] "</i>"
```

Beim Einfügen von globalen und lokalen Ausnahmen wird ebenfalls eine feste Reihenfolge eingehalten, die im Gegensatz zu derjenigen zwischen positiven und negativen Ausnahmen weniger arbiträr ist: Es werden zuerst globale und dann lokale Ausnahmen eingefügt, sodass beispielsweise eine globale Negativausnahme im Einzelfall durch eine lokale Positivausnahme überschrieben werden kann.

5.3 Darstellung von Wörterbuchartikeln

Nachdem im vorherigen Kapitel Werkzeuge und Mechanismen für die Abbildung von teilstrukturierten Texten besprochen wurden, beschäftigt sich dieses Kapitel mit einer konkreten Umsetzung. Der Fokus liegt dabei auch auf der inhaltlichen Analyse eines Wörterbuchtexts und einer Betrachtung der grundlegenden Elemente und Strukturen, die eine formelle Grammatik aufweisen muss, um einen solchen abzubilden. Obwohl dies auf Basis des REW betrachtet wird, soll ein möglichst allgemeingültiges Modell entworfen werden, so dass es mit einer geringen Anzahl an Änderungen auf andere Wörterbücher übertragen werden kann. Diese Allgemeingültigkeit kann allerdings

[34] Das ist vor allem bei sehr kurzen Ausnahmen der Fall, z.B. einzelnen Zahlen.

vielfach nur auf Ebene der „höheren“ Regeln, die mehr die allgemeinen Grundelemente beschreiben, sinnvollerweise in Anspruch genommen werden, weshalb im Allgemeinen nur diese betrachtet werden. Spezialisiertere Regeln, die deren genauere Ausgestaltung beschreiben sind oft so quellenspezifisch, dass sie maximal exemplarisch erwähnt, aber nicht in der Tiefe analysiert werden.

Zu Beginn werden einige grundlegenden Problemtypen mit entsprechenden Lösungen besprochen, die auf verschieden Ebenen auftreten, während im weiteren die verschiedenen hierarchischen Ebenen eines Wörterbuchs besprochen und eine Abbildung mit einer formellen Grammatik konzipiert wird.

Anmerkung: Die Ausschnitte aus den verschiedenen Grammatiken, die in diesem Kapitel angegeben werden, sind größtenteils deutlich vereinfacht und dienen zur Veranschaulichung der vorgestellten Konzepte. Die vollständigen im Webportal verwendeten Fassungen können in der aktuellsten Version hier abgerufen werden: https://www.rew-online.gwi.uni-muenchen.de/index.php/grammars/

5.3.1 Allgemeine Modellierungsgrundlagen

Dieses Kapitel behandelt zunächst einige prinzipielle Überlegungen und Erfahrungen, wie Grundtypen von Problemen behandelt werden können, die an verschiedenen Stellen (und auf verschiedenen Ebenen) auftreten. Konkrete Anwendungsfälle werden in den späteren spezifischeren Kapiteln behandelt.

Ein regelmäßiges Vorkommen sind mehrere optionale Bestandteile, die einem bestimmten Element zugeordnet sind. Hierbei ist es meist sinnvoll keine feste Reihenfolge in der Grammatik zu verlangen, auch wenn der Großteil der Vorkommen eine solche aufweist. Im Normalfall kann nicht davon ausgegangen werden, dass keine abweichenden Varianten auftreten, vgl. z.B. die Reihenfolge von Bedeutung und Literaturangabe in der folgende Abbildung.

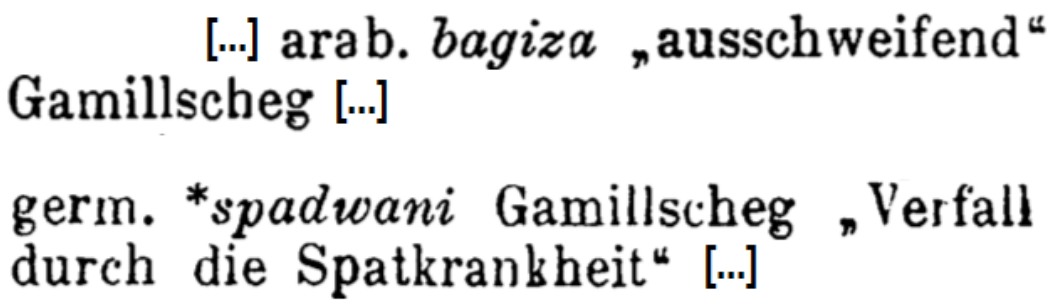
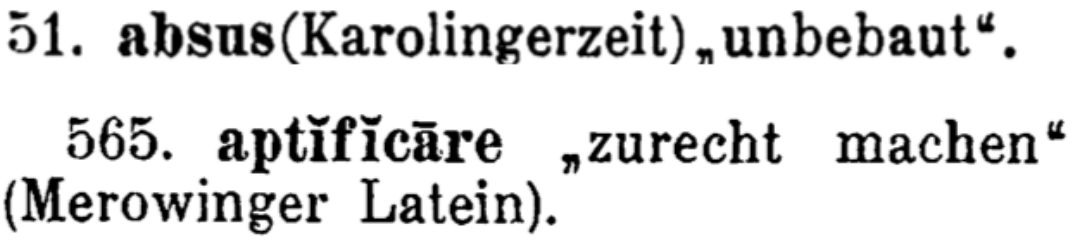


Abbildung 5.18: Unterschiedliche Reihenfolgen von Bedeutungen und Literaturangaben in REW, S. 861 (oben) und REW, S. 8125 (unten)

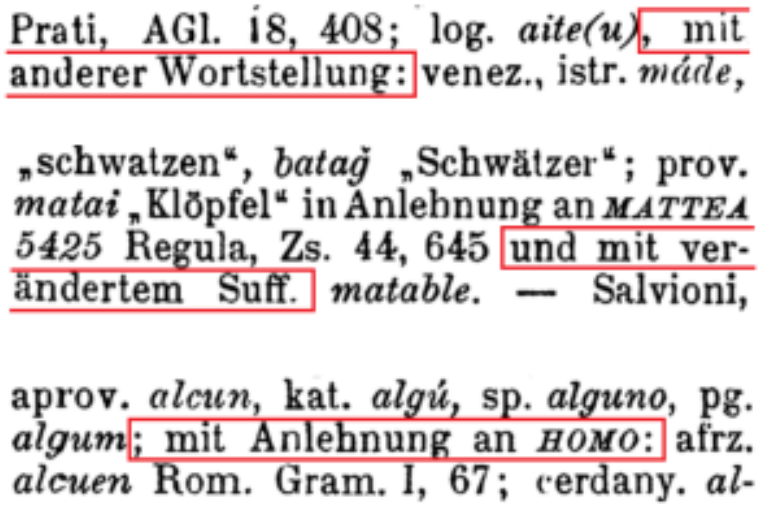
Die Reihenfolge an sich kann unter Umständen durchaus relevant sein. Wenn hier eine unterschiedliche Behandlung sinnvoll ist, kann diese Information allerdings besser im nächsten Arbeitsschritt entsprechend verwendet werden, solle aber nicht auf Ebene der Grammatik unterschieden werden. In den meisten Fällen gibt es allerdings keinen erkennbaren Grund für die Wahl der Anordnung, vgl. z.B. auch folgende Varianten in der Kopfzeile:


51. absus (Karolingerzeit) „unbebaut“.

565. aptificāre „zurecht machen“ (Merowinger Latein).

Abbildung 5.19: Verschiedene Reihenfolge bei der Angabe einer zeitlichen Eingrenzung im REW

Ein wichtige Klasse von Grundelementen, die an vielen Stellen vorkommen, sind Trennzeichen. Im Fall des REW werden Formvarianten bzw. Sprachbelege in den meisten Fällen durch Kommata und Teilsätze durch Semikola getrennt. In den Beispielen werden deshalb aus Vereinfachungsgründen meist nur diese angegeben. In der Praxis kommen allerdings oftmals auch längere Varianten vor, deren Haupttypen als eigene Regeln Eingang in die Grammatik finden, während seltenere Formen nur über Ausnahmen erkannt werden können. Abb. 5.20 zeigt einige Beispiele für solche längeren Vorkommen. Alle Separatoren (auch wenn sie meist nur aus einem Zeichen bestehen) werden aus diesem Grund als diskursive Elemente aufgefasst. In der zweiten Phase der Strukturerkennung werden diese somit erneut bearbeitet. In diesem Fall werden die Abkürzung „Suff.“ und das lateinische Etymon im letzten Beispiel erkannt und entsprechend markiert.


Prati, AGl. 18, 408; log. *aite(u)*, mit anderer Wortstellung: venez., istr. *máde*,

„schwatzen“, *batağ* „Schwätzer“; prov. *matai* „Klöpfel“ in Anlehnung an *MATTEA* 5425 Regula, Zs. 44, 645 und mit verändertem Suff. *matable*. — Salvioni,

aprov. *alcun*, kat. *algú*, sp. *alguno*, pg. *algum*; mit Anlehnung an *HOMO*: afrz. *alcuen* Rom. Gram. I, 67; cerdany. *al-*

Abbildung 5.20: Beispiele für längere Separatoren im REW (oben: Lemma 172, Mitte: 994, unten: 339)

Texteinschübe innerhalb der strukturellen Elemente können mit sehr restriktiven Basisregeln initial definiert werden, wobei im iterativen Prozess häufig vorkommende Varianten ergänzt werden[35]. Die Definition von sehr „toleranten“ Regeln führt oftmals zu falschen Positiven. Ein Beispiel ist folgender Ausschnitt:

„Siebenb. *kocie* stammt aus dem Magy. () Tiktin; Tagliavini, RHongr., 1, 21.“ (REW, S. 4729)

Wird die Erkennung von Zusatzinformationen zu einem *Sprachbeleg* wenig restriktiv definiert, wird der Abschnitt „stammt aus dem Magy. ()“ unter Umständen als solche interpretiert, wodurch der Satz als Belegliste aufgefasst wird und die Form mit dem Lemma verknüpft, obwohl dies hier offensichtlich gerade nicht der Fall ist.

Nützlich sind in vielen Fällen auch *Platzhalterregeln*, die nie gültig sind, aber die Grundlage für Ausnahmen sein können. Besonders wichtig sind diese, wenn bestimmte Elemente unterschiedlich verarbeitet werden müssen, aber syntaktisch nicht unterschieden werden können. Ein Beispiel hierfür die Unterscheidung von Zusatzinformationen, die sich auf eine Bedeutung beziehen von solchen, die sich auf den gesamten Beleg beziehen. Die Details hierzu finden sich in Kap. 5.3.4. Auch sehr selten auftretende Fälle, für die eine allgemeine Regel schwierig ist, könnten so vorgesehen werden. Deren Behandlung ist zwar in der aktuellen Konzeption nur durch manuelle Nachbearbeitung möglich, sie erhöhen aber die Mächtigkeit des zugrundeliegenden Modells. Formell kann eine solche Regel beispielsweise mit der Formulierung `!. .` umgesetzt werden.

[35] Die Häufigkeit kann wie in Kap. 3.1 beschrieben im Verlauf durch Sammlung und Etikettierung bestimmt werden. In diesem Fall kann sie auf einer quantitativen Analyse der entsprechenden Ausnahmen beruhen.

5.3.2 Makrostruktur

Jeder Artikel im REW besteht aus einer Kopfzeile und einem Artikelinhalt[36], was der Struktur der meisten Wörterbücher entspricht. Die Kopfzeile enthält dabei die Artikelnummer und das Lemma oder die Lemmata samt deren Bedeutungen und eventueller zusätzlicher Informationen. Der Artikelinhalt besteht vor allem aus einer oder mehren Listen von Formen in den Einzelsprachen bzw. Dialekten und/oder diskursiven Elementen, die unstrukturiert weiterführende Angaben enthalten. Die folgende Abbildung veranschaulicht diese Grobstruktur:

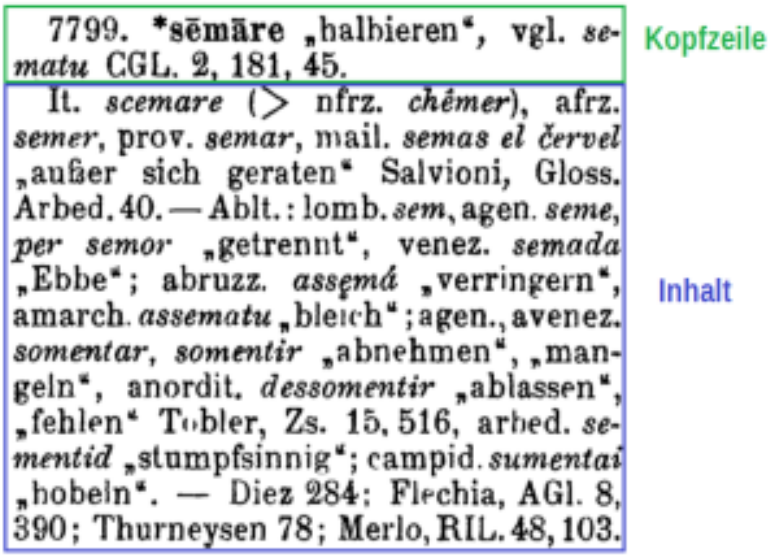


Abbildung 5.21: Grundstruktur eines REW-Artikels

Selbst die auf den ersten Blick einfach strukturiert wirkende Kopfzeile kann dabei in den unterschiedlichsten Ausprägungen vorkommen. Die folgende Abbildung zeigt Beispiele, die den grundlegenden Variationen entsprechen.

[36] Die einzige Ausnahme bildet Nummer 247, die nur angibt, dass eine bestimmte Form nicht besteht. Dieser Fall wird durch eine spezielle Ausnahmeregel abgefangen. Falls solche Fälle in anderen Quellen systematisch vorkommen, müsste das entsprechend in der Grammatik berücksichtigt werden.

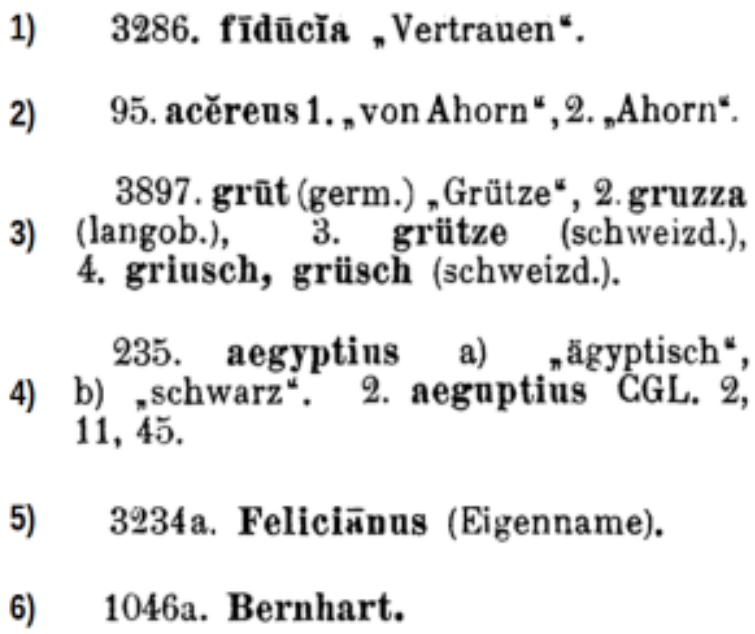


Abbildung 5.22: Verschiedene Typen von Kopfzeilen im REW: 1) Einfache Kopfzeile, 2) Nummerierung verschiedener Bedeutungen, 3) Nummerierung verschiedener Lemmata, 4) Mischform, die 2 und 3 kombiniert, 5) Eigenname mit Spezifikation, 6) Eigenname ohne Spezifikation

Eine Grammatik, die allgemeine Einträge abbilden soll, muss diese verschiedenen Formen und ihre logische Struktur entsprechend berücksichtigen. Sie ist für die spätere Erzeugung der Daten von entscheidender Bedeutung, da aus dieser Informationen über die Bedeutungen der sprachlichen Formen und deren etymologischer Herleitung inferiert werden können (vgl. Kap. 7.2.3). Aufbauend auf den verschiedenen Varianten kann folgender Aufbau einer Kopfzeile konstruiert werden:

```
head  <- lemma_num '. ' (given_name_info / standard_lemma) lemma_sub_list? num_lemm

standard_lemma <- lemma_list (meanings / head_comment)*

given_name_info <-
    given_name_const
    / given_name_list (" (" small_lang_list ")")? " " (
                "(" given_name_type_literal ")"
              / "„" given_name_type_literal """
              / given_name_type_literal
              )
```

Die elementaren Bestandteile sind somit eine Lemmanummer (bestehend aus einer Zahl und optional einem Buchstaben) und entweder ein „Standard-Lemma“ oder ein Eigenname. Darauf folgen eine Reihe von optionalen Elementen. Ersteres kann dabei als Liste eines oder mehrerer Lemmata und optional einer Liste von Bedeutungen und eines Kommentars[37] dargestellt werden. Da die Reihenfolge der beiden optionalen Elemente nicht konsequent ist, sind beide Varianten möglich[38]. Für Eigennamen gibt es wiederum im REW verschiedenste Notationskonventionen, die durch die Regel *given_name_info* abgebildet werden[39]. Dabei ist *given_name_const* eine reine Platzhalterregel für Ausnahmen (vgl. Kap. 5.3), die Fälle mit Eigennamen ohne jegliche Auszeichnung (vgl. Fall 6 in der obigen Grafik) abdeckt. Die restliche Regel beschreibt den Normalfall, in dem auf einen (oder mehrere) Eigennamen eine optionale Sprachabkürzung und eine von drei Notationsvarianten, für die Markierung folgt:

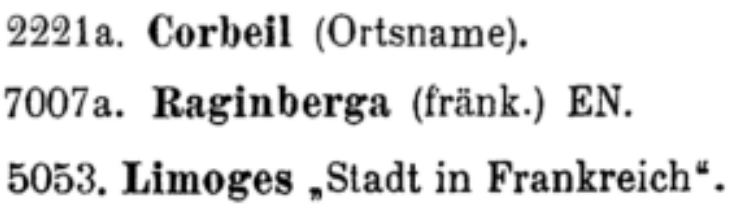


Abbildung 5.23: Drei unterschiedliche Notationen für die Markierung eines Eigennamens

Besonders letztere ist dabei ungünstig, da die Notation in den Anführungszeichen gewissermaßen impliziert, dass hier eine reguläre Bedeutungsangabe vorliegt, was aber nicht der Fall ist. Das entsprechende Lemma ist eben nicht eine Bezeichnung für das Konzept STADT IN FRANKREICH, sondern für eine spezifische Instanz einer solchen. Bei der menschlichen Betrachtung ist dies intuitiv verständlich, für die algorithmische Verarbeitung ist ein solcher Fall allerdings schwierig, da die konsistente Notation eine Klammerung (wie im ersten Beispiel) wäre. In Kap. 10.1 wird dieses Thema näher betrachtet.

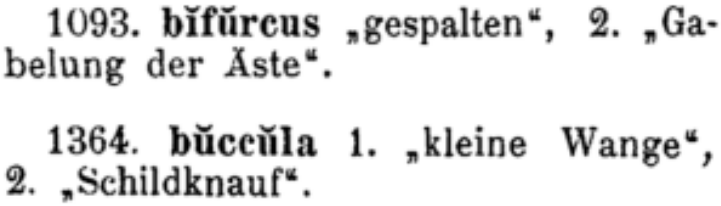
Auf die notwendigen Elemente in der Kopfzeile folgen dann (abgesehen vom Kommentar) zwei weitere optionale Elemente *lemma_sub_list* und *num_lemma_list*. Beide werden für zusätzliche Lemmata bzw. Bedeutungen verwendet, stehen aber hierarchisch auf verschiedenen Ebenen. Während *lemma_sub_list* für eine genauere Aufteilung des Lemmas steht, auf das sie folgen, werden über *num_lemma_list* weitere zusätzliche Lemmavarianten bzw. ähnliche Lemmata angegeben. Im REW werden folgende grundlegenden Regeln verwendet:

[37] Die Regel *head_comment* taucht absichtlich sowohl am Ende der Kopfzeile, als auch am Ende von *standard_lemma* auf, da an beiden Stellen Texteinschübe möglich sind.

[38] Die Regel würde in dieser Form auch mehrere Bedeutungslisten bzw. Kommentare erlauben, was aber nicht vorkommt und deshalb unproblematisch ist.

[39] An dieser Stelle ist schwer zu beurteilen, wie allgemeingültig eine solche Regel sein kann, sie wird hier trotzdem zumindest als Beispiel betrachtet

- Wenn ein Lemma mehrere Bedeutungen hat, denen jeweils auch unterschiedliche romanische Formen im Artikeltext zugeordnet sind, werden diese mit arabischen Zahlen nummeriert. Es gibt allerdings zwei strukturell leicht unterschiedliche Variationen:


1093. **bĭfūrcus** „gespalten“, 2. „Gabelung der Äste“.

1364. **bŭccŭla** 1. „kleine Wange“, 2. „Schildknauf“.

Abbildung 5.24: Zwei Varianten der Nummerierung mehrere Bedeutungen für ein Lemma im REW

- Wenn mehrere Lemmata genannt werden, werden diese ebenfalls mit römischen Zahlen nummeriert. Hier tritt (mit einer Ausnahme, vgl. Kap. 5.1.3) nur die erste Variante auf:

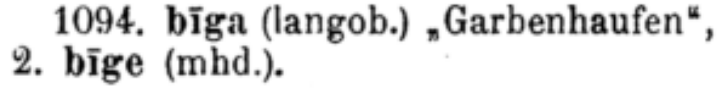

1094. **bīga** (langob.) „Garbenhaufen“, 2. **bīge** (mhd.).

Abbildung 5.25: Nummerierung mehrerer Lemmata im REW

- Treten mehrere Lemmata auf und mindestens eins davon hat eine weitere Unterteilung der Bedeutungen, werden diese mit Buchstaben bzw. ebenfalls mit arabischen Zahlen nummeriert. In manchen Fällen werden sie allerdings auch einfach auf eine Ebene durchnummeriert:


261. **afflāre** 1. „zuwehen“, 2. „finden“, 3. **arflare** GGr. 1², 486, 67.

785. **augŭrium** „Vogelflug“, 2. ***agŭrium** Einführung 159: 1) „Vogelflug“, 2) „Vorbedeutung“, 3) „Glück“.

1984. **clavus** „Nagel“, 2. **claus** Einf. 131 a) „Nagel“, b) „Furunkel“.

Abbildung 5.26: Zweistufige Nummerierung im REW

All diese Fälle können nun mit Hilfe der beiden erwähnten Regeln behandelt werden. Dazu wird eine *num_lemma_list* als Liste definiert, deren Elemente entweder die bereits erwähnten „Standard-Lemmas“ oder Bedeutungen sind, während eine *lemma_sub_list* nur Bedeutungen enthalten kann:

```
lemma_sub_list        <-  lemma_sub_list_entry (', ' lemma_sub_list_entry)+ '.'
lemma_sub_list_entry  <-  sub_list_enum ') ' meaning_list

num_lemma_list        <-  num_lemma (', ' num_lemma)* pot_dot
num_lemma             <-  part_number (standard_lemma / meaning_lemma) lemma_sub_list?
meaning_lemma         <-  meanings head_comment?
standard_lemma        <-  lemma_list (meanings / head_comment)*
```

Die Beispiele aus Abb. 5.24, 5.25 und 5.26 werden damit folgendermaßen erfasst:

1093 . **bĭfŭrcus** „gespalten , 2. „Gabelung der Aste“.
1364 . **bŭccŭla** 1. „kleine Wange“, 2. „Schildknauf“.
1094 . **bĭga** (langob.) „Garbenhaufen , 2. **bīge** (mhd.).
261 . **afflāre** a) „zuwehen“, b) „finden“. 2. **arflare** GGr. 1², 486, 67.
785 . **augŭrium** „Vogelflug , 2. ***agŭrium** Einführung 159: 1) „Vogelflug“, 2) „Vorbedeutung“, 3) „Glück“.
1984 . **clavus** „Nagel , 2. **claus** Einf. 131 a) „Nagel“, b) „Furunkel“.
lemma_num standard_lemma num_lemma_list lemma_sub_list

Das Konstrukt der *lemma_sub_list* wird also ausschließlich im Falle der doppelten Unterteilung der Beispiele aus Abb. 5.26 auf. In anderen Fällen werden die nummerierten Elemente als eigene Lemmata aufgefasst, auch wenn nur eine Bedeutung gegeben wird. Die eigentliche sprachliche Form, die zu dieser Bedeutung gehört wird dann im Zuge der Erzeugung relationaler Daten (vgl. Kapitel 7) ergänzt. Eine Eigenheit dieser Modellierung ist außerdem, dass die ersten beiden Beispiele unterschiedlich aufgefasst werden, obwohl diese grundsätzlich die gleichen Strukturelemente enthalten. Im ersten Fall werden zwei Lemma-Elemente erzeugt, im zweiten Fall drei:

1093 . **bĭfŭrcus** „gespalten , 2 . „Gabelung der Aste .
1364 . **bŭccŭla** 1 . „kleine Wange , 2 . „Schildknauf .
lemma_num standard_lemma part_number meaning_lemma

Diese Abweichung wird im Zuge der weiteren Verarbeitung angeglichen, sodass jeweils strukturell identische Daten erzeugt werden. Grundsätzlich könnte man dies auch auf Gramamtikebene lösen, es würde allerdings zu einer deutlich unintuitiveren Formalisierung führen, die potentiell schlechter nachnutzbar ist, als das sehr generalisierte Modell, das hier vorgestellt wurde.

Der Hauptteil des Artikels weist bezogen auf die Makrostruktur zwei Varianten auf, nämlich einen einzelnen Textblock oder mehrere nummerierte. Der erste Block wird im

letzteren Fall meist auch nummeriert, dies ist allerdings nicht immer der Fall.

1983. **clavŭla** „Pfropfreis“.
Judik. *ťaula* „kleiner, dürrer Zweig“. — Ablt.: not. *čauluni* „Pfropfreis“ Salvioni, Zs. 23, 517; campodolc. *kanawla*, tosk. *chiola* „Knöchel“ Salvioni, Zs. 34, 389. (Friaul. *kenole*, *kanole*, grödn. *kunodla* „Puls“ Salvioni, Zs. 34, 388; 404 ist kaum möglich Jud, BDR. 4, 59.)

---

243. **aĕrūgo, -ĭne** „Rost, Grünspan“, 2. ***aerīgo** Rom. Gram. 2, 359.
Rum. *ruginĕ*, it. *ruggine*, veron. *maruzene* „Groll“, log. *ruindzu*, camp.

[...]

2. Romagn. *redzna*, sp. *orin* Mask. — Diez 278; García de Diego 25. (Obw.

[...]

1984. **clavus** „Nagel“, 2. **claus** Einf. 131 a) „Nagel“, b) „Furunkel“.
1. Log. *ğau*.
2a. It. *chiodo*, *chiovo*, avenez. *clold*, parm., regg. *čold*, piac. *čod*, friaul. *klaud*,

[...]

Abbildung 5.27: Drei Artikel aus dem REW: Einfacher Artikel, Artikel mit unnummeriertem ersten Textblock und Artikel mit nummeriertem ersten Textblock

Da diese Fälle grundsätzlich deutlich weniger Variation aufweisen als die Struktur der Kopfzeile, können sie auf die folgende einfache Weise beschrieben werden:

```
parts                <-   numbered_part_first (new_line numbered_part)*
numbered_part_first  <-   part_prefix? part_content
numbered_part        <-   part_prefix part_content
```

Das Resultat ist in allen Fällen eine Liste von Textblöcken. Falls nur ein einzelner vorhanden ist, hat dieser Liste die Länge eins. Während beim ersten Textblock die Nummerierung optional ist, müssen alle weiteren eine solche aufweisen.

Der Artikel als Ganzes kann nun durch folgende einfache Regel abgebildet werden:

```
entry   <-   head new_line parts
```

Erwähnenswert ist hier, dass kein Zusammenhang zwischen dem Format der Kopfzeile und dem Format des Hauptteils hergestellt wird. Das hat den Grund, dass zwar in den meisten Fällen auf eine Kopfzeile mit mehren nummerierten Lemmata oder Bedeutungen ein ebenfalls nummerierte Hauptteil folgt, es aber auch Abweichungen auf beiden Seiten gibt (d.h. Nummerierungen in der Kopfzeile und nicht im Hauptteil, s. Abb. 5.28, und umgekehrt, s. Abb. 5.29). Somit ist strengere Formulierung der Grammatik an dieser Stelle nicht möglich. Die Zuordnung der Bestandteile in Kopfzeile und Hauptteil findet dann (soweit möglich) im nächsten Prozessschritt statt (vgl. Kap. 7.2.3).

24. abies, -iēte „Tanne“. **2. *abēte** Einführ. 111.

Ligur. *ave(o)*, valses. *aveid*, piac. *aved*, bellun., pad., venez. *albeo*, z. T. speziell die „Weißtanne“, daher das *l*; it. *abeto*, *-e*, apul., kalabr. *apitu*, siz. *abbitu*, march. *ubbeta*, gask., toulous., langued., kat. *abet*, mallork. *vet*, sp., pg. *abeto*. Formen in erbwörtlicher Entwicklung scheinen auf den Alpengürtel beschränkt zu sein, nur hier finden sich denn auch die Ablt.: venez. *avedin*, friaul. *lavadiṅ*, enneb. *aidiṅ*, comel. *vdiṅ* „Weißtanne“. — Mussafia 25; Salvioni, RIL. 39, 621; AGl. 16, 286; Wartburg; Moll 13.

Abbildung 5.28: Eintrag aus dem REW, der eine nummerierte Kopfzeile und einen nicht nummerierten Hauptteil aufweist

248. aestīvus „sommerlich“.

1. It. *stio* „Sommerleinen“ Diez 404.

2. Aapul. *stibo*, log. *istiu*, prov., kat., valenc. *estiu*, sp., pg. *estío* „Sommer“. — Ablt.: aprov. *estival* „sommerlich“, afrz. *estivailles* „Sommergetreide“, H.-Loire *etyivalo* „Brachland“; mallork. *estivar* „dürr werden“, „die Kräfte wegen zu großer Hitze verlieren“; pg. *estiar* „aufhören zu regnen“, „aufheitern“, „abkühlen“. — Merlo, Stag. mes. 31; Wartburg. (Das Verbum ist kaum lat. *AESTIVARE* „den Sommer verbringen“, eher das Adj. Fortsetzung von *AESTIVALIS*; it. *stivale* s. *8264*.)

Abbildung 5.29: Eintrag aus dem REW, der eine einfache Kopfzeile und einen nummerierten Hauptteil aufweist

Die folgende Grafik illustriert das gewünschte Ergebnis anhand eines sehr einfachen

Wörterbuchartikels:

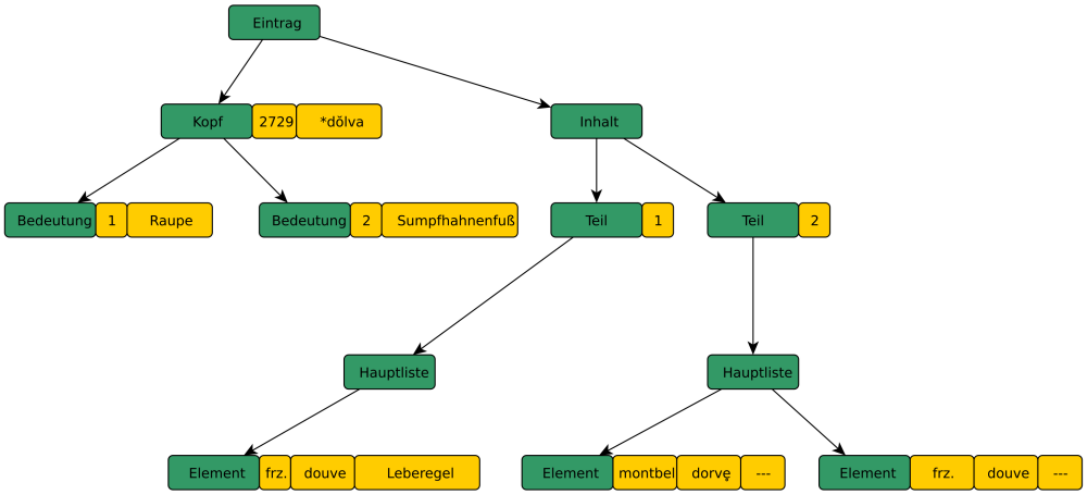


Abbildung 5.30: Darstellung der grundsätzlichen hierarchischen Struktur eines Wörterbuchartikels

5.3.3 Beleglisten und Sätze

Für die weitere Erfassung der Artikeltexte stellt sich zuerst die Frage, welche weiteren hierarchischen Gliederungselemente verwendet werden sollen. Im REW gibt es beispielsweise das Trennzeichen —, mit dem u.a. eine Liste von abgeleiteten Formen von den Erbwörten zu Beginn eines Artikels getrennt wird. Auch ein Zeilenumbruch, der im Normalfall nur verwendet wird, um einzelne nummerierte Textblöcke voneinander zu trennen, kommt in seltenen Fällen auch innerhalb eines Blocks als Trennzeichen vor. Beide Konventionen werden allerdings in hohem Maße inkonsistent verwendet, vgl. z.B. folgende Beispiele:

friaul. *anzile*, frz. *anguille*, béarn. *añele*.
— Ablt.: kat. *anguiler* „Taucher“ *(mer-*

Piazz. *anguli* Salvioni, MIL. 21, 259.
Ablt.: afrz. *soi angler* „anbeißen von

Abbildung 5.31: Ableitungslisten mit und ohne Trennzeichen. Beispiel 1 stammt aus REW, S. 461, Beipiel 2 aus REW, S. 463.

Damit haben diese für eine hierarchische Strukturierung des Artikeltexts wenig Relevanz und die weitere hierarchische Aufteilung des Artikels findet ausschließlich auf Ebene des Satzes statt[40]. Die grundsätzliche Modellierung eines Textabschnitts kann damit folgendermaßen dargestellt werden:

```
part_content   <-   main_list extra* / extra+
```

Er besteht somit entweder aus einer Hauptliste[41] und optional weiteren Zusatzbestandteilen oder nur aus mindestens einem diese Zusatzbestandteile. Diese können vier verschiedene Ausprägungen haben. Falls keine davon zutrifft, werden sie einer Rest-Kategorie zugeordnet (also als diskursives Element behandelt):

```
extra   <-   (pot_space / new_line_sep)? unsure_extra_list / extra_list / general_lit_ref / no
```

Die verschiedenen Unterregeln haben dabei folgende Bedeutung:

- **unsure_extra_list**: Eine Liste von Belegen, die als unsicher markiert sind (vgl. hierzu Kap. 6.5.2)
- **extra_list**: Eine zusätzliche Belegliste (im Fall des REW z.B. Ableitungen oder Zusammensetzungen)
- **general_lit_ref:** Eine Liste von Literaturverweisen, die nicht direkt einem oder mehreren Sprachbelegen zugeordnet ist, sondern als eigener „Satz“ gegeben wird.
- **not_accepted:** Ein geklammerter Textbestandteil, der im Falle des REW Ausführungen zu sprachlichen Formen enthält, die nicht dem jeweiligen Lemma zugeordnet werden (obwohl dies beispielsweise andere Quellen getan hatten). Der geklammerte Text kann dabei durchaus aus mehreren Sätzen bestehen, wird aber nicht weiter unterteilt, da er ausschließlich natürlichsprachigen Inhalt hat und nicht in der ersten Phase der Strukturerfassung behandelt werden kann.

Die Regel für die Restkategorie ist dabei entscheidend für die Menge an Daten, die strukturiert aus einem Artikel extrahiert werden kann. Im einfachsten Fall könnte diese beliebige Zeichen und somit immer den vollständigen restlichen Textabschnitt umfassen:

[40] Mit Satz ist hier grundsätzlich jeder Textbestandteil gemeint, der mit einem Punkt abgeschlossen wird. Das kann eine Liste von Formen, ein natürlichsprachiger Satz oder auch eine Mischform aus beidem sein.

[41] Eine „Hauptliste“ ist eine Liste von Sprachbelegen (entsprechend der Definition in Kap. 2.2) zu Artikelbeginn, die nicht genauer spezifiziert ist. Im REW werden mit dieser die direkten Erbwörter angegeben.

```
rest <- (!new_line .)*
```

Dies könnte für Quellenmaterial ausreichend sein, in dem strukturierte und diskursive Abschnitte strikt getrennt sind und in einer fixen Reihenfolge vorkommen. Da dies nicht dem Normalfall entspricht, ist eine komplexere Formulierung dieser Regel nötig, die auch das Ende eine unstrukturierten Passage erkennt, so dass eventuell darauf folgende strukturierte Elemente entsprechend erfasst werden können. Es gibt dafür zwei Kriterien:

- Die unstrukturiere Passage wird durch einen Punkt beendet.
- Durch ein Trennzeichen (im Normalfall ein Semikolon) wird wiederum ein strukturierter Teilabschnitt eingeleitet.

Das erste Kriterium wirkt dabei einfach, allerdings bereiteten auch hier wieder vor allem die häufig vorkommenden, meist durch einen Punkt abgeschlossenen Abkürzungen Probleme. Es ist also entscheidend eine Liste von Elementen zu führen, die zwar (potentiell) einen Punkt enthalten, aber trotzdem nicht das Ende eines Satzes markieren.

```
rest_abbr <- text_part_dot
          / lit_ref
          / (lang_lower_dot ('-' lang_lower_dot)?)
          / (lang_upper_dot ('-' (lang_lower_dot / lang_upper_dot))?)
          / abbreviation
          / grammar_spec
          / meaning
          / ('<i>' [^<]+ '</i>')
          / ('<u>' [^<]+ '</u>')
          / ('(' [^)]+ ')')
          / ('[' [^\]]+ ']')
          / ([1-9] [0-9]? '. ' century_literal)
```

Die erste Regel ist dabei eine reine Platzhalterregel, mit derer zusätzliche abweichende Fälle über Ausnahmen angelegt werden können. Die restlichen Regeln können in die drei folgenden Kategorien eingeteilt werden:

- **Abkürzungen bzw. längere Textelemente, die Abkürzungen enthalten**: In diesem Fall handelt es sich um die Regeln 2-5, die Literaturverweise, Sprachabkürzungen (bzw. deren Verkettung mit Bindestrichen), allgemeine Abkürzungen und grammatikalische Angaben umfassen. Diese Elemente bauen wiederum stark auf die erfassten Verzeichnisse und die Ergänzung von dort fehlenden Einträgen auf.

- **Textelemenete, die zwar nur in seltenen Fällen Punkte enthalten, aber innerhalb derer kein Satzende vorkommen kann**: Diese Kategorie umfasst die Regeln 6-10 und behandelt die Fälle, in denen beispielsweise innerhalb einer Bedeutung oder einer sprachlichen Form abkürzende Schreibweisen vorkommen, die selbstverständlich nicht das Ende eines Satzes markieren.
- **Weitere Notationen mit Punkten, die häufig genug vorkommen, dass sich das Erstellen einer eigenen Regel lohnt**: Im Fall des REW handelt es sich dabei um die Angabe eines Jahrhunderts, was durch die letzte Regel abgedeckt wird.

Das zweite Kriterium für das Ende einer unstrukturierten Passage kann durch eine negative *Lookahead Assertion* abgebildet, werden, die das „Rest-Element“ abbricht, wenn auf ein Semikolon eine Belegliste folgt. Zusammengenommen kann die folgende Regel aufgestellt werden, die das „Rest-Element“ als einer Verkettung der unter *rest_abbr* festgelegten Elemente und weiterer beliebiger Zeichen definiert, die durch einen Punkt (bzw. eine Fragezeichen) oder ein Semikolon, der eine Belegliste einleitet, abgebrochen wird:

```
rest   <-   (!new_line rest_abbr? ((' ' rest_abbr) / (!('; ' record_list_sub) [^.?]))+) [.?;]?
```

Somit kann eine klare Trennung der einzelnen Sätze auch unter Vorhandensein von unstrukturierten Teilelementen sichergestellt werden.

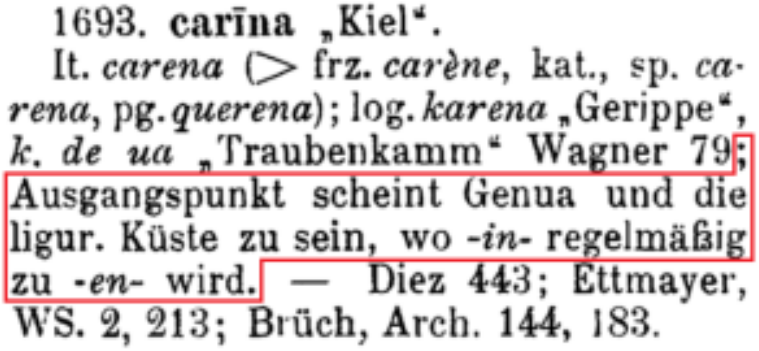
Die Beleglisten, also das Kernstück für die Erfassung der lexikalischen Daten, sind nun eine einfache Liste von Sprachbelegen (s. Kap. 5.3.4), die durch ein entsprechendes Trennzeichen voneinander abgegrenzt werden. Dabei können auf Ebene der Belegliste zwei verschiedene Typen unterschieden werden, deren Abgrenzung für die späteren Arbeitsschritten (vgl. Kap. 7.2.3) entscheidend ist. Es handelt sich hierbei um Trennzeichen zwischen einzelnen Belegen (im REW im Normalfall durch Kommata dargestellt) und Trennzeichen zwischen verschieden Sublisten (im REW durch Semikola oder durch längere Konstruktionen wie „; mit Suff.W.:“, vgl. Kap. 5.3.1 repräsentiert). Beleglisten werden (je nach Kontext in zwei Varianten, die mit einem Klein- oder Großbuchstaben beginnen) folgendermaßen definiert:

```
record_list_main  <-  (text_insertion / record_cap) record_list_rest
record_list_sub   <-  record_small record_list_rest

record_list_rest  <-  ((record_list_sep / record_sep) (text_insertion / record_small))*
```

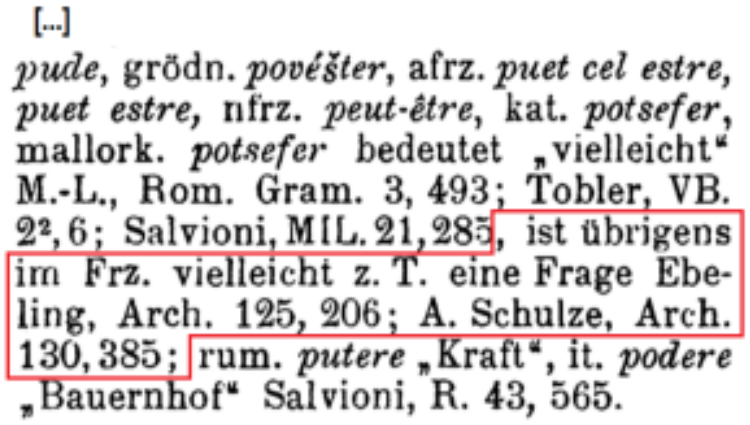
Die Regel *text_insertion* erlaubt dabei das Vorhandensein von unstrukturierten Teilabschnitten, somit können nicht nur Fälle, in denen unstrukturierte Teilsätze auf

strukturierte Folgen, sondern auch der umgekehrte Fall (s. Abb. 5.32) und zusätzlich die Möglichkeit, dass sich ein Texteinschub innerhalb einer Belegliste befindet (s. Abb. 5.33), abgedeckt werden.



1693. **carīna** „Kiel".
It. *carena* (▷ frz. *carène*, kat., sp. *carena*, pg. *querena*); log. *karena* „Gerippe", *k. de ua* „Traubenkamm" Wagner 79; Ausgangspunkt scheint Genua und die ligur. Küste zu sein, wo *-in-* regelmäßig zu *-en-* wird. — Diez 443; Ettmayer, WS. 2, 213; Brüch, Arch. 144, 183.

Abbildung 5.32: Unstrukturierter Abschnitt nach einer Belegliste im REW



[...]
pude, grödn. *povéšter*, afrz. *puet cel estre*, *puet estre*, nfrz. *peut-être*, kat. *potsefer*, mallork. *potsefer* bedeutet „vielleicht" M.-L., Rom. Gram. 3, 493; Tobler, VB. 2², 6; Salvioni, MIL. 21, 285, ist übrigens im Frz. vielleicht z. T. eine Frage Ebeling, Arch. 125, 206; A. Schulze, Arch. 130, 385; rum. *putere* „Kraft", it. *podere* „Bauernhof" Salvioni, R. 43, 565.

Abbildung 5.33: Unstrukturierter Abschnitt als Einschub in einer Belegliste in REW, S. 6682

Die beiden anfangs festgelegten Regeln für Beleglisten haben darauf aufbauend folgende Darstellung:

```
main_list  <-  record_list_main pot_dot

extra_list <-
       text_separator_literal? (list_specifier record_list_sub) pot_dot
    /  text_separator_literal? record_list_main pot_dot
    / <&'; ' record_list_sub list_comment? pot_dot
```

Die zusätzlichen Beleglisten können dabei optional mit einem einleitenden *list_specifier*

versehen sein (Fall 1), der weitere Information für die darauf folgenden Belege enthält (vgl. dazu auch Kap. 7.2.2). Der dritte Fall dient zur Behandlung des oben genannten Falles, dass eine Belegliste nach einem Semikolon auf ein „Rest-Element“ folgt.

5.3.4 Sprachbelege

Vorbemerkung zur Bezeichnung „Sprachbeleg“ in diesem Kapitel:
Grundsätzlich wird der Begriff Sprachbeleg immer entsprechend der Definition in Kap. 2.2 verwendet. Entsprechendes gilt für den englischen Begriff *record*, der für die Bezeichnung der jeweiligen Datenbanktabellen (vgl. Kap. 6.5) verwendet wird. Er bezeichnet als die Zuordnung genauer einer sprachlichen Form zu Bedeutungen, geographischer und zeitlicher Information. Auf den Wörterbuchtext bezogen entspricht dies beispielsweise folgender Angabe:

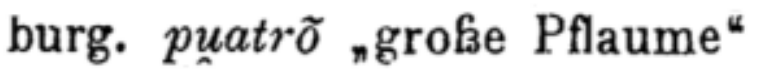


Abbildung 5.34: Sprachbeleg in (REW, S. 6688)
Es sind allerdings genauso abkürzende Schreibweisen mit mehrere Sprachen, Formen oder Bedeutungen möglich:

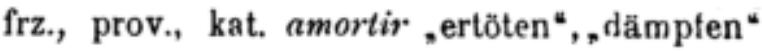


Abbildung 5.35: Abkürzende Schreibweise für mehrere Sprachbelege in (REW, S. 186)
Somit enthält dieser Abschnitt definitionsgemäß sechs verschiedene Sprachbelege, die auch als solche in die Datenbank eingefügt werden. Im Kontext der strukturellen Erkennung wird dieses Konstrukt, also ein Element einer Belegliste, hier etwas unscharf ebenfalls als Sprachbeleg bzw. *record* bezeichnet, um die Formulierungen zu vvereinfachen.

Ein Sprachbeleg kann somit mit folgender Regel beschrieben werden:

```
record   <-   ('[' record ']')
         /  lang_list (' ' / &pre_extra_info) filler_form? pre_form_info? form_it_list

form_it_list <- form_it_extended (form_sep form_it_extended)*
```

Der erste Fall behandelt dabei sogenannte Buchwörter, die mit eckigen Klammern markiert werden, während die eigentliche Definition des Sprachbelegs über eine Liste von Sprachabkürzungen und eine Liste mit Elementen von *form_it_extended* besteht. Zwischen den beiden erlauben zwei separate Regeln noch Füllwörter wie „auch“ bzw.

eine Angabe von Zusatzinformationen vor der eigentlichen Form. Letztere ist wiederum eine reine Platzhalterregel, mit der selten auftretende abweichende Notationen modelliert werden können, vgl. folgendes Beispiel:

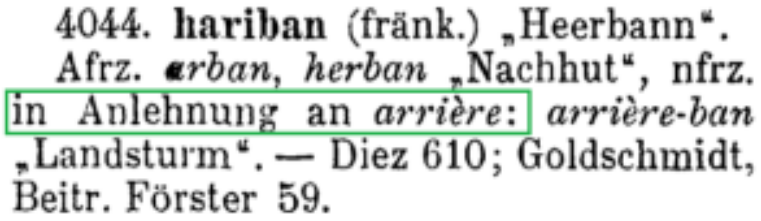

4044. **hariban** (fränk.) „Heerbann“.
Afrz. *arban*, *herban* „Nachhut“, nfrz.
in Anlehnung an *arrière*: *arrière-ban*
„Landsturm“. — Diez 610; Goldschmidt,
Beitr. Förster 59.

Abbildung 5.36: Angabe zusätzlicher Information vor der eigentlichen sprachlichen Form im REW. Dieser Fall tritt nur selten auf, da solche Angaben im Normalfall nachgelagert werden.

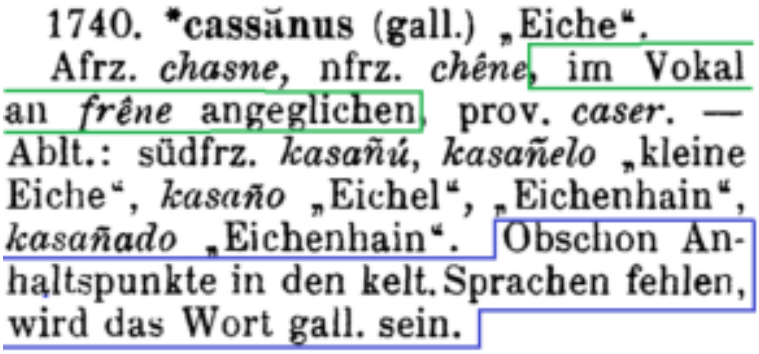
Die Regel *form_it_extended* kann dabei als einzelne sprachliche Form und einer Reihe von optionalen weiteren Elementen definiert werden:

```
form_it_extended <- form_it form_it_extra*

form_it_extra <-
   extra_info
 / grammar_info
 / borrowing
 / lit_refs
 / meanings
 / second_etymon
```

Die schließt Bedeutungen, grammatikalische Spezifikationen, Literaturverweise und ein zweites Etymon (vor allem bei Kontaminationen) ein, aber auch längere Strukturelemente wie eine Liste von Formen, die aus der vorangegangenen entlehnt sind. Die Regel *extra_info* dient zur Modellierung von weiterführender Information. Diese stellt ein Beispiel von unstrukturiertem Text da, der einer anderen Entität zugeordnet ist, in diesem Fall dem Sprachbeleg[42]. Im Gegensatz zu den bisher betrachten diskursiven Elementen, die aus eigenen Sätzen bzw. Satzbestandteilen bestanden und somit ausschließlich als Artikelbaustein behandelt wurden, werden diese in der Datenbank dem jeweiligen Beleg zugeordnet (vgl. Kap. 6.5.3). Die folgende Grafik illustriert den Unterschied:

[42]Es können beispielsweise in Einzelfällen auch Sprachen zusätzliche Informationen wie geographische Eingrenzungen zugeordnet werden, dies wird hier aber nicht im Detail behandelt.



1740. *cassănus (gall.) „Eiche“.
Afrz. *chasne*, nfrz. *chêne*, im Vokal an *frêne* angeglichen, prov. *caser*. — Ablt.: südfrz. *kasañú*, *kasañelo* „kleine Eiche“, *kasaño* „Eichel“, „Eichenhain“, *kasañado* „Eichenhain“. Obschon Anhaltspunkte in den kelt. Sprachen fehlen, wird das Wort gall. sein.

Abbildung 5.37: Zwei Formen von diskursiven Elementen im REW: Der grün markierte Text bezieht sich auf einen spezifischen Beleg, während der blaue einen unabhänger Artikelbestandteil darstellt.

Von den verschiedenen Ausprägungen von *form_it_extra* soll im weiteren ausschließlich die Angabe der Bedeutungen näher betrachtet werden, da die genauen Definitionen einerseits quellenabhängig und andererseits in den meisten Fällen selbsterklärend sind. Die Bedeutungen enthalten allerdings zum Teil eine zusätzliche Spezifikation, die in einem wichtigen Zusammenhang zur eben erwähnten Regel *extra_info* steht. Die entsprechende Regel hat folgende Form:

```
meaning   <-   '„' german_sentence '"'
             (   meaning_record_spec
               / meaning_spec
               / (' (' german_sentence ')')
             )?
```

Auf die reguläre Bedeutungsangabe folgt somit entweder eine sogenannte *meaning_record_spec,* eine *meaning_spec* oder eine eingeklammerte Zusatzangabe[43]. Um den Unterschied dieser drei Optionen zu verdeutlichen, ist eine Analyse der verschiedenen Arten von Zusatzangaben hilfreich, die im Kontexts eines Sprachbelegs vorkommen können. Grundsätzlich können drei Formen von Spezifikation bzw. zusätzlicher Information unterschieden werden:

- **Im Bezug auf die sprachliche Form**: Gibt Zusatzinformation zur Gestalt der Form selbst, es werden also beispielsweise phonetische oder morphologische Details vor allem im Bezug zum jeweiligen Etymon beschrieben. Häufig sind im

[43] Die Regel *german_sentence* beschreibt dabei keinen streng definierten Satz, sondern eine Reihe von Wörtern aus dem deutschen Alphabet, die durch Zeichen wie Leerzeichen oder Kommata verbunden werden. Es sind also sowohl einzelne Wörter, als auch längere Abschnitte bis hin zu ganzen Sätzen möglich.

REW Formulierungen wie „mit unerklärtem *t*-“ (REW, S. 2465) oder „mit *k* von *carta*“ vorhanden.

- **Im Bezug auf den Sprachbeleg also die Verwendung der Form mit einer bestimmten Bedeutung bzw. in einem bestimmten Kontext**: Zusätzliche Information ist in vielen Fällen ein Erklärungsansatz für die Verwendung der Form in einer bestimmten Bedeutung, die stark von der Bedeutung der Ausgangsform abweicht, wie bespielsweise „vielleicht nach den Sprüngen des Delphins“ (REW, S. 2544), kann aber auch andere Formen wie historische Erklärungen (vgl. z.B. „von D'Aubigné eingeführt, später *brindestoc* geschrieben“ (REW, S. 1304)) o.ä. annehmen. Eine Spezifizierung bezieht sich beispielsweise auf den Kontext der Verwendung (vgl. z.B. „namentlich von der Milch“ (REW, 1065a)) oder eine genauere räumliche bzw. zeitliche Eingrenzung (vgl. z.B. „heute namentlich im Osten und Südosten“ (REW, S. 611)).
- **Im Bezug auf die Bedeutung**: Hier wird eine Bedeutung genauer spezifiziert, d.h. die angegebene deutsche Bezeichnung (oder ein eventuelles Homonym) hat unterschiedliche Bedeutungen, die die fremdsprachliche Form nur zum Teil aufweist. Beispiele hierfür sind „'Morgen' (Feldmaß)“ (REW, 112a) oder „'Eichel' (im Kartenspiel)“ (REW, S. 122)

Bei den ersten beiden Fällen ist die Unterscheidung eher akademischer Natur, da der Übergang zwischen beiden fließend ist. Somit werden technisch die beiden ersten Typen jeweils dem Sprachbeleg selbst zugewiesen. Durchaus relevant ist aber die Unterscheidung zwischen einer Spezifikation, die sich auf die Verwendung einer bestimmten Form bezieht und einer, die sich nur auf die Bedeutung bezieht, da zweitere Teil der Bedeutungsangabe ist und auch als solche behandelt werden muss, damit diese nicht allgemeiner aufgefasst wird, als sie in der Quelle gegeben wird. Ein weiterer Unterschied ist, dass Bedeutungen auch in einem globaleren Kontext verwendet werden, d.h. sie werden beispielsweise mit anderen synonymen Bedeutungen zusammengeführt und mit externen Normdatenbanken verknüpft (vgl. Kapitel 10). Sie können somit technisch (intern, s. Kap. 12.1, oder extern, s. Kapitel 13) genutzt werden, während die beiden anderen Formen kontextabhängig und ausschließlich für eine menschliche intellektuelle Nutzung gedacht sind. Eine syntaktische Abgrenzung ist allerdings oftmals kaum möglich, wie die folgende Tabelle zeigt:

Typ	**Beispiel**	**Quelle**
Bedeutungsspezifikation ohne Klammern	Uengad. *egla* „Eichel“ im Kartenspiel	(REW, S. 122)
Bedeutungsspezifikation mit Klammern	Nfrz. *ouaille* „Schäfchen“ (im übertragenen kirchlichen Sinne)	(REW, S. 6124)
Zusatzinformation ohne Klammern	Obw. *deržer* „richten“ nach dem Deutschen	(REW, S. 2649)
Zusatzinformation mit Klammern	Frz. MA. *aleluja* „Sauerklee“ (weil er zu Ostern blüht)	(REW, S. 4000)

Tabelle 5.7: Beispiele für die Angabe von Bedeutungsspezifikationen bzw. weiterführender Information

Anhand der Struktur können somit höchstens eine Reihe von Grundtypen unterschieden werden, die häufig vorkommen, eine allgemeine Regel ist nicht möglich. Die grundsätzliche Unterscheidung orientiert sich somit an der Häufigkeit, sodass die Fälle ohne Klammerung als *extra_info* aufgefasst werden, wohingegen geklammerte Spezifikationen als Bedeutungseingrenzung interpretiert werden. Dies wird durch den letzten Fall im obigen Codebeispiel abgedeckt. Abweichende Vorkommen werden entsprechend über den Ausnahmenmechanismus geregelt, der aber das Vorhandsein entsprechender Regeln voraussetzt. Somit erklären sich die Platzhalterregeln *meaning_spec* (für Bedeutungsspezifikationen ohne Klammerung) und *meaning_record_spec* (für geklammerte Angaben nach einer Bedeutung, die sich auf den ganzen Beleg beziehen).

5.3.5 Unstrukturierte Bestandteile

Dieses Kapitel befasst sich schließlich mit der in Kap. 5.2 beschriebenen zweiten Phase der strukturellen Erfassung und deren Umsetzung mit Hilfe einer formellen Grammatik. Alle in der vorherigen Phase als unstrukturiert zurückgestellten Elemente werden hier weiterverarbeitet. Das schließt nicht nur natürlichsprachliche Sätze und längere Abschnitte ein, sondern auch weitere Elemente wie die Beleg- bzw. Bedeutungsspezifikationen, Beleg- oder Sublistentrenner und einleitende Listenelemente. Zur Illustration werden hier exemplarisch alle solchen Elemente von REW, S. 794 markiert:

794. ***aurĭdiāre** (von *aura 788*) „säuseln“. It. *oreggiare*, prov. *aurejar*, kat. *orejar* „lüften“, „erfrischen“, astur. *ouritsá* „worfeln“, sp. *orear* „lüften“. — Ablt. : it. *oreggio* und aus dem Norden entlehnt *(o)rezzo*, *arezzo* (> pg. *oressa*), béarn. *aurei*, astur. *oreo* „Lüftchen“; bologn. *urets* „gegen Norden gelegener Ort“, kors. *oreggu* „Abendkühle“, romagn. *urets* „Schatten“ Guarnerio, RIL. 48, 527. Mit *b-* von *1308* : it. *brezza* „kühler Wind“, *ribrezzo* „Schauer“, „Schauder“, „Abscheu“; lucc. *ribrezzare* „worfeln“ Salvioni, RIL. 41, 212; Pieri, AGl. 15, 357; Zs. 30, 303. (*Orezzo* AURITIUM Diez 31 ist morphologisch bedenklich und erklärt das tönende *-zz-* nicht.)

Im Gegensatz zur ersten Phase, die die Struktur des Texts als Ganzes abgebildet, und ihn darauf aufbauend in immer kleinere Bestandteile zerlegt hat, werden nun ausschließlich die relevanten Basisentitäten ohne Berücksichtigung des Kontexts gesucht und markiert. Im Fall eines Wörterbuchs sind das[44]:

[44] vgl. hierzu auch z.B. (Renders 2011, S. 118)

- Beleglisten bzw. einzelne sprachliche Formen[45]
- Literaturverweise
- Referenzen auf andere Einträge
- Einzelne Bedeutungen
- Einzelne Sprachabkürzungen
- Allgemeine Abkürzungen

Das zugrundeliegende Problem kann somit als *Named Entity Recognition (NER)* beschrieben werden, einem Teilgebiet der Computerlinguistik, das sich mit der „Erkennung und Klassifikation von Eigennamen“ (Carstensen 2010, S. 596) beschäftigt. Der Begriff *Eigennamen* wird dabei nicht ausschließlich im strikten linguistischen Sinne verwendet, sondern deutlich weiter gefasst als „in der Regel sprachliche Ausdrücke, die auf Individuen von Klassen oder Typen bestimmter Entitäten referieren, wie z. B. wie [sic] Personen-, Firmen-, Produktnamen, komplexe Datums-, Zeit-, und Maßausdrücke“ (Carstensen 2010, S. 596), eine Definition, die somit auf die oben erwähnten Elemente durchaus zutrifft. Die konkrete Durchführung variiert dabei, meist werden allerdings Techniken aus dem Bereich des maschinellen Lernens verwendet (vgl. z.B. Nadeau und Sekine 2007). Dabei ist durchaus ein starker Bezug zur Lexikographie vorhanden, wenn auch eher zum Erstellen von Wörterbüchern bzw. Datenbanken aus Textkorpora (vgl. z.B. Horák, Rambousek u. a. 2017) und weniger zur Anwendung innerhalb eines bestehenden traditionellen Wörterbuchs. Das liegt wohl hauptsächlich daran, dass solche zwar strukturell äußerst inkonsistent sind, aber doch durch den Satz und gewisse Notationen die Basiselemente sehr distinktiv markiert werden[46] und mit verhältnismäßig einfachen Methoden erkannt werden können. Im Falle der genannten Entitätsklassen ist das prinzipiell der Fall, problematisch sind hierbei wiederum bestimmte Arten von Inkonsistenzen innerhalb der Quelle. Im REW betrifft das in diesem Kontext vor allem die folgenden beiden Punkte:

- Die Verzeichnisse der verschiedenen Formen sind unvollständig (Im REW stehen aktuell 1147 (Stand 01.09.2023) aus den Abkürzungsverzeichnissen extrahierten Einträgen 1281 (Stand 01.09.2023) zusätzliche gegenüber).
- Artikelreferenzen werden (trotz entsprechender Festlegung im Abkürzungsverzeichnis, vgl. (REW, S. XXXIII)) oftmals nicht kursiviert.

Das erste Problem betrifft nicht nur die Behandlung der textuellen Bestandteile, sondern ebenso sehr die in den vorherigen Kapiteln besprochenen Strukturelemente, die

[45] Eine einzelne sprachliche Form im Text wird als Sprachbeleg mit unbekannter Bedeutung aufgefasst, vgl. Kap. 6.5.3

[46] Die technische Grundlage, um dies ausnutzen zu können, ist allerdings eine gute Erkennung der Formatierung des Quellentexts (vgl. Kap. 4.3).

diese die Abkürzungen zum großen Teil ebenfalls nutzen. Es sollte also eine möglichst niederigschwellige und intuitive Möglichkeit vorhanden sein diese Verzeichnisse in der Datenbank über die Oberfläche des Webportals entsprechend zu erweitern (vgl. Kap. 12.5). Dies spielt in den Anfangsphasen des iterativen Gesamtprozesses eine Rolle (vgl. Kap. 3.1), aber auch für eine spätere Korrektur im Einzelfall[47].

Die Lösung des zweiten Problems kann gewissermaßen aus zwei Richtungen angegangen werden. Zum einen werden auch nicht kursivierte Formen als Referenzen aufgefasst, wenn sie in bestimmten Kontexten vorkommen, also beispielsweise auf eine sprachliche Form oder bestimmte Zeichenkette wie „vgl.“ oder „s.“ folgen[48]. Zum anderen werden viele der Referenzen, die gerade in solchen Kontexten vorkommen durch den in Kap. 4.3 vorgestellten Post-Processing Vorgang automatisiert kursiviert, wenn in den entsprechenden manuell korrigierten Artikeln die fehlende Kursivierung konsequent verbessert wird[49]. Hierbei können u.a. auch die in Kap. 8.2 vorgestellten Werkzeuge verwendet werden.

Insgesamt können somit (mit wenigen Einzelausnahmen) alle zu Beginn dieses Kapitels gelisteten Entitätsklassen syntaktisch erkannt werden. Da deren Definition (mit Ausnahme der Artikelverweise) bereits in der ersten Phase gebraucht wurde, liegt es nahe die entsprechenden Regeln auch für die zweite Phase wiederzuverwenden und somit die Erkennung wiederum mit einer formellen Grammatik vorzunehmen. In Teilen müssen die alten Regeln allerdings überschrieben werden. Das ist bei den Sprachbelegen der Fall, bei denen innerhalb der Beleglisten eine Reihe von Füllwörtern und ähnlichem erlaubt sind, die im Kontext eines natürlichsprachigen Satzes nicht unbedingt in der gleichen Art und Weise gemeint sein müssen. Somit werden einzelne in der Grammatik vorhandene Unterregeln in diesem Kontext durch reduzierte Varianten oder auch die immer falsche Regel *nothing* ersetzt, so dass die hauptsächlichen Regeln weiterhin verwendet werden können. Die Beschreibung der Sprachbelege ist somit in diesem Schritt strikter und erlaubt weniger Varianz. Im Zweifelsfall können nur einfachere Bestandteile wie einzelne sprachliche Formen oder Bedeutungen erkannt werden.

Die Regeln zur Verarbeitung der unstrukturierten Bestandteile kann in folgender Weise angegeben werden:

```
text_section   <-  (special_element token_end)? until_space (spaces (special_element &token_en

special_element <- record_list_main
    / record_list_small
    / entry_ref
```

[47] Denkbar wäre auch ein technischer Ansatz zur Lösung dieses Problems. Einige Überlegungen dazu finden sich in Kapitel 14

[48] Eine allgemeine Verwendung aller Zahlen innerhalb des Texts ist weniger sinnvoll, da auch z.B. Jahreszahlen, Referenzen auf die Nummerierung innerhalb eines Artikels etc. vorkommen.

[49] Korrekturen, die den Originaltext verändern, werden speziell markiert (vgl. Kap. 8.1).

```
    / smallcaps_record
    / lit_ref
    / abbreviation
    / mult_lang_lower
    / mult_lang_upper
    / meaning
    / form_it

token_end  <-   &[.,;:?! ]
    / !.

until_space  <- [^ ]*

spaces   <-   ' '+
```

Bei der Verarbeitung wird der Text also tokenweise durchsucht und zu Beginn jedes neuen Tokens überprüft, ob es der Anfang einer der speziellen Entitäten ist. Die Regel *until_space* konsumiert im gegenteiligen Fall alle Zeichen bis zum nächsten Leerzeichen. Mit *token_end* wird sichergestellt, dass die erkannten Entitäten abgeschlossen sind, d.h. dass nicht der Beginn eines längeren Tokens als Gesamttoken erkannt wird. Bei einzelnen Sprachabkürzungen im Text werden ebenfalls Verkettungen wie „Pg.-Galiz.“ erkannt, die ein einzelnes Token bilden. Die Regel *smallcaps_record* erkennt schließlich Etyma, die im Text meist in Kapitälchen und nicht kursiviert angegeben werden. Das folgende Beispiel zeigt die entsprechenden Markierungen innerhalb des obigen Beispielartikels:

794. ***aurĭdiāre** (von *aura* *788*) „ säuseln “. It. *oreggiare*, prov. *aurejar*, kat. *orejar* „lüften“, „erfrischen“, astur. *ouritsá* „worfeln“, sp. *orear* „lüften“. — Ablt. : It. *oreggio* und aus dem Norden entlehnt *(o)rezzo*, *arezzo* (> pg. *oressa*), béarn. *aurei*, astur. *oreo* „Lüftchen“; bologn. *urets* „gegen Norden gelegener Ort“, kors. *oreggu* „Abendkühle“, romagn. *urets* „Schatten“ Guarnerio, RIL. 48, 527. Mit *b-* von *1308* : it. *brezza* „kühler Wind“, *ribrezzo* „Schauer“, „Schauder“, „Abscheu“; lucc. *ribrezzare* „worfeln“ Salvioni, RIL. 41, 212; Pieri, AGl. 15, 357; Zs. 30, 303.. (*Orezzo* AURITIUM Diez 31 ist morphologisch bedenklich und erklärt das tönende *-zz-* nicht.) number form abbreviation bib_entry entry_or_page latin_form

6 Datenbankstruktur

Bevor die weitere Verarbeitung der Strukturdaten aus dem vorherigen Kapitel betrachtet wird (s. Kapitel 7), soll an dieser Stelle behandelt wernde, welche Datenmodellierung für die endgültigen Resultatdaten verwendet wird. Dieses Kapitel beschäftigt sich somit mit der Konzeption einer relationalen Datenbank, die die fein granulierte Daten abbilden kann, die aus einem Wörterbuch erschlossen werden, und die Einhaltung der Prinzipien aus Kap. 3.1 ermöglicht. Das Modell basiert auf sechs verschiedenen Klassen von Tabellen, die zum Teil bereits aus Kap. 3.4 bekannt sind, aber hier nochmal konkreter auf den Wörterbuchkontext bezogen werden:

- **Eingangstabellen**: Diese enthalten die eigentlichen textuellen Ausgangsdaten sowie weitere Informationen zu deren Herkunft (vgl. Kap. 6.1)
- **Prozesstabellen**: In diesen Tabellen werden Daten gespeichert, die für die algorithmische Verarbeitung der Eingangsdaten zusätzlich nötig sind. Dies schließt beispielsweise Ausnahmen ein, mit denen an verschiedenen Stellen in den Prozess eingegriffen werden kann, aber auch die Regeln für die formelle Grammatik oder eine Auflistung aller spezieller Zeichen, die im Wörterbuch vorkommen. Diese Tabellen werden hier nicht im Detail besprochen, ihre Inhalte werden aber an den Stellen betrachtet, an denen sie benötigt werden.
- **Objekttabellen**: Sie stellen eine Hälfte der eigentlichen Resultatdaten (vgl. Kap. 6.2) dar und beschreiben ein bestimmtes Objekt, das aus den Eingangsdaten erzeugt wurde (z.B. Wörterbuchartikel oder Bibliographie-Einträge).
- **Geteilte Tabellen**: Diese stellen die andere Hälfte der Resultatdaten (vgl. Kap. 6.2) dar und enthalten Entitäten, die an verschiedenen Stellen vorkommen können. Beispiele hierfür sind Bedeutungen, sprachliche Formen oder Literaturverweise. Im Gegensatz zu den Tabellen der vorherigen Klasse werden hier keine Duplikate erstellt, d.h. wenn beispielsweise die gleiche Bedeutungsangabe in mehreren Artikeln vorkommt, wird nur beim Verarbeiten des ersten Artikels ein neuer Wert erstellt.
- **Anreicherungstabellen**: Diese dienen dazu Resultatdaten anzureichern oder mit externen Daten zu verknüpfen. Die Details dazu werden in den Kapiteln 10, 11.1 und 11.2 besprochen.

- **Hilfstabellen**: Diese werden automatisiert aus anderen Tabellen erstellt und dienen dem einfacheren und effizienteren Zugang auf die aktuellen Daten (vgl. Kap. 6.6)

6.1 Darstellung der Eingangsdaten

Die einzigen „Primärdaten“ im engeren Sinne enthält die Tabelle *lines*, in der die vollständigen textuellen Inhalte der Quelle in Zeilenform enthalten sind. Zusätzlich wird deren Position im Wörterbuch über die Seitenzahl, das Seitensegment und die Zeilennummer (vgl. Kap. 5.1) sowie die Pixelkoordinaten der Zeile im verarbeitenden Scan (vgl. Kap. 4.2) gespeichert. Der Inhalt der Zeile entspricht dabei immer dem Stand des Imports, jegliche Änderungen werden als explizite Datensätze in der Tabelle *corrections* angelegt (vgl. Kap. 8.1). Weiterhin ist jeder Zeile ein Importvorgang aus der Tabelle *imports* zugeordnet, die Informationen zu den Zeitpunkten der jeweiligen Importe enthält. Grundsätzlich können somit ganze Seiten, die eine sehr hohe Anzahl Fehler enthalten vollständig neu importiert werden, wobei trotzdem der jeweilige Zustand der Eingangsdaten jederzeit nachvollziehbar bleibt[1]. Aus den Werten der drei Tabellen *lines*, *imports* und *corrections* wird automatisiert die aktuelle Textdarstellung einer Zeile unter Anwendung aller Korrekturen erzeugt (vgl. Kap. 6.6).

Eine Frage, die sich in diesem Kontext stellt, ist allerdings, warum die textuellen Eingangsdaten auf Ebene von Zeilen in der Datenbank abgebildet werden. Denkbar wären ebenfalls feinere oder gröbere Granulierungen. Um dies zu begründen ist es hilfreich drei Kriterien zu beachten, die im Kontext der maschinellen Weiterverarbeitung und der Publikation sinnvoll sind:

- **Quellentreue**: Die Rohdaten sollten eine möglichst unverfälschte Abbildung des Originaltexts sein. Dieser kann somit zusätzlich über das Webportal angeboten werden, außerdem lassen sich so Fehler in den weiteren Verarbeitungsschritten leichter nachvollziehen.

- **Statische Bestandteile**: Im Fall von Korrekturen sollten sich nur die Inhalte der Textbestandteile ändern, nicht aber deren Anzahl oder Anordnung. Dies vereinfacht die Zuordnung von Textbestandteilen und den daraus abgeleiteten Daten deutlich.

- **Atomarität**: Jedes abgeleitete Objekt sollte einer Menge von Textbestandteilen zugeordnet werden, aus denen es erzeugt wurde. Dies macht den Einsatz von Verfahren zur Bündelung der Textelemente wie in Kap. 5.1 erst möglich.

[1] Nach Möglichkeit sollte dies allerdings nur in extremen Fällen verwendet werden, da evtl. bestehende Korrekturen erneut auf die neu importieren Zeilen angewendet werden müssen.

Grundsätzlich denkbare andere Arten der Darstellung wären die nächsthöhere Ebene, der Absatz, oder die nächstniedrigere Ebene, das Token[2]. Für beide Ebenen ist das zweite Kriterium allerdings problematisch, da weder Wortgrenzen noch Absatzgrenzen vom OCR-System mit Sicherheit erkannt werden können[3]. Bei den Wortgrenzen ist vor allem der Blocksatz innerhalb der relativ schmalen Spalten ein Problem. Zum Teil sind hier kaum Wortgrenzen erkennbar (vgl. Abb. 6.1), während in anderen Fällen sehr große Abstände vorkommen (vgl. Abb. 6.2).

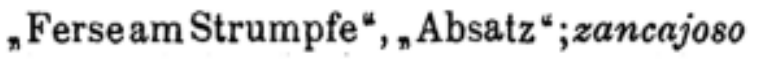


Abbildung 6.1: Beispiel für kaum vorhandene Wortgrenzen (REW, S. 9598)

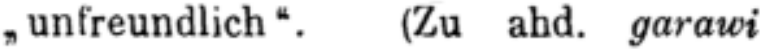


Abbildung 6.2: Beispiel für sehr große Abstände zwischen den einzelnen Textbestandteilen (REW, S. 1524)

Somit werden vom OCR-System in solche Fällen zusätzliche oder zu wenige Leerzeichen erkannt, die nur zum Teil mit den Methoden in Kap. 4.3.3 korrigiert werden können, was wiederum zu inkorrekten Tokenisierungen führt. Bei den Absatzgrenzen liegt das Problem ähnlich. Auch wenn diese in den meisten Fällen korrekt erkannt werden, würde die Korrektur von fehlenden oder überschüssigen die Anzahl und die Zuordnung der Textbestandteile verändern, was zu unnötigem Verwaltungsaufwand führt. Die Verwendung eines tokenisierten Texts kann zusätzlich mit dem ersten Kriterium kollidieren, falls gewünscht ist, dass Wörter, bei denen im Originaltext eine Silbentrennung stattgefunden hat, als ein Token dargestellt werden sollen. Die Auflösung dieser Trennung ist allerdings nicht immer trivial und kann auch zu Fehlern führen (vgl. Kap. 5.1.2). Insgesamt stellt unter diesen Voraussetzungen die Ebene der Zeilen die beste Lösung für die Darstellung der Eingangsdaten dar.

6.2 Resultatdaten

Die eigentlichen Resultatdaten werden nun aus diesen Eingangsdaten generiert. Alle Basistabellen der einzelnen Entitäten werden hierbei mit einem Zeitstempel versehen, um den Zeitpunkt der Erstellung und den damaligen Zustand der zugrundeliegenden

[2] Eine noch gröbere Granulierung würde grundsätzlich das Atomaritäskriterium verletzen, da der Absatz das „gröbste" Strukturelement ist, das noch eindeutig beispielsweise einem einzelnen Artikel zugeordnet werden kann. Eine noch feinere Unterteilung auf Einzelzeichenebene erscheint nicht besonders praktikabel.

[3] Die Zeile hingegen ist in dieser Hinsicht sehr stabil. Maximal werden hier Artefakte oder ähnliches als zusätzliche Zeilen erkannt. Diese sind allerdings nach einer Korrektur leer und können weiterhin dem entsprechenden Eintrag zugeordnet, aber bei der Verarbeitung ignoriert werden. Durch die getrennte Segmentierung der Seiten (vgl. Kap. 4.1) tritt allerdings auch dieser Fall kaum auf.

Daten jederzeit nachvollziehen zu können. Im folgenden werden diese Basistabellen zusammengefasst:

Tabelle	**Inhalt**
abbreviations	Allgemeine Abkürzungen
bibliography	Bibliographie
entries	Wörterbuchartikel
entry_corrs	Einzelne Korrekturen zu einem bestimmten Wörterbuchartikel aus dem Anhang (vgl. Kap. 8.3.2)
entry_supps	Zusätzliche Artikelabschnitte aus dem Anhang (vgl. Kap. 8.3.1)
lang_abbreviations	Sprachabkürzungen

Mit Ausnahme der Tabellen *entries* (für die eigentlichen Wörterbuchartikel) und *bibliography* (vgl. Beispiel in Kap. 3.4) enthalten die Basistabellen alle relevanten Information für diese Entität und es sind keine weiteren Bestandteile vorhanden. Alle Basistabellen werden über eine entsprechende Verknüpfungstabelle (z.B. *c_entries_lines*) mit den Zeilen verknüpft, aus denen sie generiert wurden (vgl. Kap. 5.1.2). Somit können u.a. bei Korrekturen auf Zeilenebene diejenigen Objekte gefunden werden, die neu erstellt werden müssen. Diese Verknüpfungstabelle enthält zusätzlich den Start- und den Endindex der jeweiligen Zeile im Fließtext (vgl. ebenfalls Kap. 5.1.2). Im folgenden werden die einzelnen Entitäten mit Ausnahme der Ergänzungen aus dem Anhang (siehe hierzu Kap. 8.3) im Detail besprochen[4].

6.3 Sprachen und Dialekte

Die Tabelle *lang_abbreviations* ist die digitale Repräsentation der Liste der „Sprachen und Mundarten“ in REW, S. XXVII–XXXII, die um zusätzliche Informationen erweitert wurde, die für die Verarbeitung benötigt werden. Das REW unterscheidet dabei Sprachen und Dialekte strukturell nicht und enthält eine einzelne alphabetische Liste aller entsprechenden Abkürzungen. Syntaktisch lassen sich allerdings zwei Grundtypen von Angaben unterscheiden. Zum einen gibt es geographische Angaben mit einem Präfix wie beispielsweise „MA. von“, „MA. des“, zum anderen kommen adjektivische Formulierungen wie „angelsächsisch“ vor. Diese werden in der Datenbank durch die beiden Typen *region* und *language* unterschieden. Dabei muss beachtet werden, dass die Unterscheidung rein syntaktischer Natur ist und keinen linguistischen Kriterien genügt. Die mit *language* bezeichneten Einträge beschreiben zwar in vielen Fällen Sprachen bzw. Sprachfamilien, es sind allerdings auch Dialektbezeichnungen wie „freiburgisch“ möglich. Für die technische Verarbeitung ist an dieser Stelle vor allem relevant, dass sich Einträge der Kategorie *region* wie z.B. „MA. von Genf“ zum Teil automatisiert mit geographischen Daten verknüpfen lassen (vgl. Kap. 11.2), während

[4] Die Tabelle *abbreviations*, die allgemeine Abkürzungen wie „MA.“ (= Mundart) enthält, wird hier ebenfalls ausgelassen, da sie nur eine einfache Abbildung von Abkürzungen auf (deutsche) Bedeutungsangaben darstellt.

dies bei den restlichen nicht möglich ist.

Als dritter Typ von Sprachabkürzungen wird *reference* verwendet, das solche Abkürzungen markiert, die ausschließlich eine Variante einer anderen sind[5]. Diese können auf drei Wegen zustande kommen:

- Im Abkürzungsverzeichnis wird eine explizite Referenz angegeben, z.B. „vlev. s. val-levent.“ (REW, S. XXXII).
- Im Abkürzungsverzeichnis werden mit Hilfe von Klammern mehrere Varianten angegeben, z.B. „berr(ich). = MA. von Berry.“ (REW, S. XXVIII). Die Auflösung der Klammerung wird in Kap. 7.1.1 näher betrachtet.
- Im Wörterbuchtext werden zusätzliche Varianten einer Abkürzung verwendet. Diese Einträge werden zusätzlich als *additional* markiert.

Die letzte Kategorie ist dabei mit Abstand die umfangreichste. Allein für die Angabe „serbo-kroatisch“ gibt es nach aktuellem Stand folgende Abkürzungsvarianten, von denen nur die erste als solche im Abkürzungsverzeichnis gelistet wird:

abbreviation
serbokr.
serbo-kr.
serb.-kroat.
skr.
serbo-kroat.
serbokroat.
serb.-kr.

Mit der Spalte *id_super_lang* können Hierarchien zwischen den über die Abkürzungen festgelegten Sprachen und Dialekten angegeben werden, die zur Herleitung von Formen verwendet wird, bei denen im Wörterbuchtext nur eine Sprachabkürzung gegeben wird (vgl. Kap. 7.2.1 und Kap. 12.5.3). So werden beispielsweise die Dialekte einzelner französischer Départements der französischen Sprache zugeordnet. Weiterhin ist über *id_transcription_system* die Angabe eines Transkriptionssystems bzw. Alphabets möglich, was zur Fehlererkennung bei den zugeordneten Formen genutzt werden kann (vgl. Kap. 8.2.2.

Zusätzlich zur Tabelle *lang_abbreviations* gibt es eine weitere Tabelle *languages*, die für die eigentlichen Sprachangaben in den einzelnen Wörterbuchartikeln verwendet wird.

[5] Diese Verwendung ist eine Abweichung von der ansonsten stark normalisierten Darstellung der Datenbank, da bei Referenzen ein Großteil der Datenbankfelder immer leer ist. Dies ist hauptsächlich der Fall, weil bei vielen neu auftretenden Abkürzungen nicht unbedingt klar ist, ob sie eine eigene Abkürzung sind oder nur eine Variante einer bestehenden darstellen. Durch diese Modellierung wird die nachträgliche Änderung solcher Elemente deutlich einfacher.

Das hat den Hintergrund, dass im REW Sprachabkürzungen oftmals durch eine zeitliche bzw. räumliche Spezifikation ergänzt werden. Ein Eintrag der Tabelle *languages* besteht entsprechend aus einer Sprachabkürzung, einem optionalen geographischen Präfix wie „n“ (Nord) oder „zentral“ und der ebenfalls optionalen Angabe eines Zeitraums aus der Tabelle *time_periods*. Alle mit zeitlichen Präfixen versehene Sprachangaben wie „spätmhd.“ oder „afrz.“ werden hier als Zeiträume behandelt, da sie solchen bis zum einem gewissen Maße zugeordnet werden können (vgl. auch Kap. 2.2.4).

6.4 Bibliographische Angaben

Die Modellierung der Bibliographie-Einträge entspricht in weiten Teilen der der Sprachabkürzungen. Hier werden nur zwei Typen *entry* und *reference* unterschieden, wobei Referenzen entsprechend Varianten von bibliographischen Abkürzungen listen. Für die Abbildung von Publikationsstatus und der Angabe von Autoren, Herausgebern, Gründern oder ähnlichen zugeordneten Personen werden allerdings zusätzliche Tabellen *bib_publication_data* und *bib_person_connections* verwendet (vgl. hierzu das Beispiel aus Kap. 3.4).

Für literarische Verweise im Wörterbuchtext wird die Tabelle *lit_references* verwendet, die einen Bibliographie-Eintrag mit den genauen Angaben von Band und Seitenzahl (bzw. Lemmanummer bei vielen Wörterbüchern) verknüpft. Auch die Angabe eines zusätzlichen Autors (vor allem bei Aufsätzen) ist hier möglich.

6.5 Lexikalische Daten und Wörterbuchartikel

Die Darstellung der eigentlichen Wörterbuchartikel ist dabei mit Abstand die komplexeste Modellierung. Hier enthalten sind einerseits die eigentlichen lexikalischen Daten ensprechend der Definition in Kapitel 2, zum anderen werden diese in den Kontexts des jeweiligen Artikels eingeordnet und mit weiteren Angaben wie Literaturverweisen verknüpft. Die folgende Grafik zeigt die verwendeten Tabellen und deren Relationen untereinander:

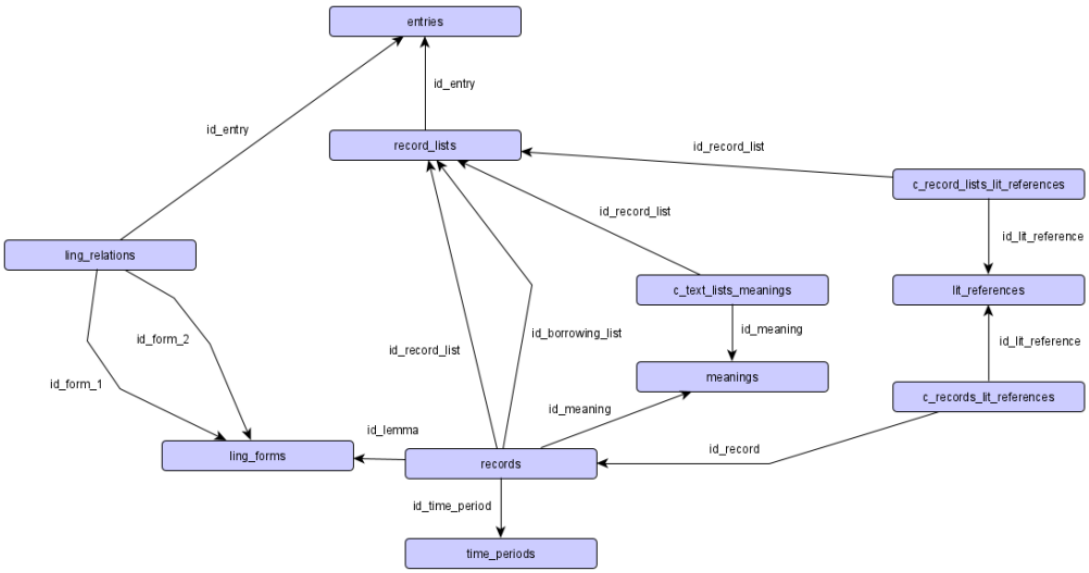


Abbildung 6.3: Datenbankstruktur für die Darstellung der Wörterbuchartikel

Die eigentliche Basistabelle *entries* enthält dabei kaum eigene Information. Sie besteht nur[6] aus den Feldern *number* und *letter* für die entsprechende Lemmanummer, den Volltextfeldern *head_text* und *body_text*, die den ursprünglichen Fließtext enthalten, und der Spalte *head_comment*, die einen Kommentar ablegt, der sich auf die gesamte Kopfzeile bezieht (und nicht auf einzelne Lemmata). Der Fließtext wird nicht für die eigentliche Darstellung der Artikel an der Oberfläche verwendet (vgl. Kap. 12.2), ermöglicht aber die Anzeige des Ursprungstexts und eine Volltextsuche in diesem (vgl. Kap. 12.1).

Die weitere Artikelstruktur wird über die zugeordneten Tabellen *record_lists* und *records* abgebildet. Ein Eintrag in *record_lists* entspricht dabei grundsätzlich einer Liste von Sprachbelegen oder einem natürlichsprachigen Satz (vgl. Kap. 5.3.3). Er wird über den Identifikator des Wörterbuchartikels, die Nummer (*part*) und (optional) den Buchstaben (*sub_part*) des Textblocks und die Position des Eintrags innerhalb des Textblocks indiziert. Dabei gelten folgende Konventionen:

- Falls keine nummerierten Textblöcke vorhanden sind wird die Nummer eins verwendet.
- Die Lemmata werden ebenfalls als *record_lists* dargestellt. Ihre Position ist immer null. Die erste Liste des Hauptteils beginnt somit mit Position eins.

Falls mehrere nummerierte Lemmata vorhanden sind, werden diese somit als eigene Listen aufgefasst. Die folgende Grafik und Tabelle zeigen die Grenzen der verschiedenen *record_list*-Elemente und deren Indizierung in der Datenbank:

[6] Abzüglich der Spalten für die Verwaltung, die für alle Basistabellen vorhanden sind (vgl. Kap. 3.4)

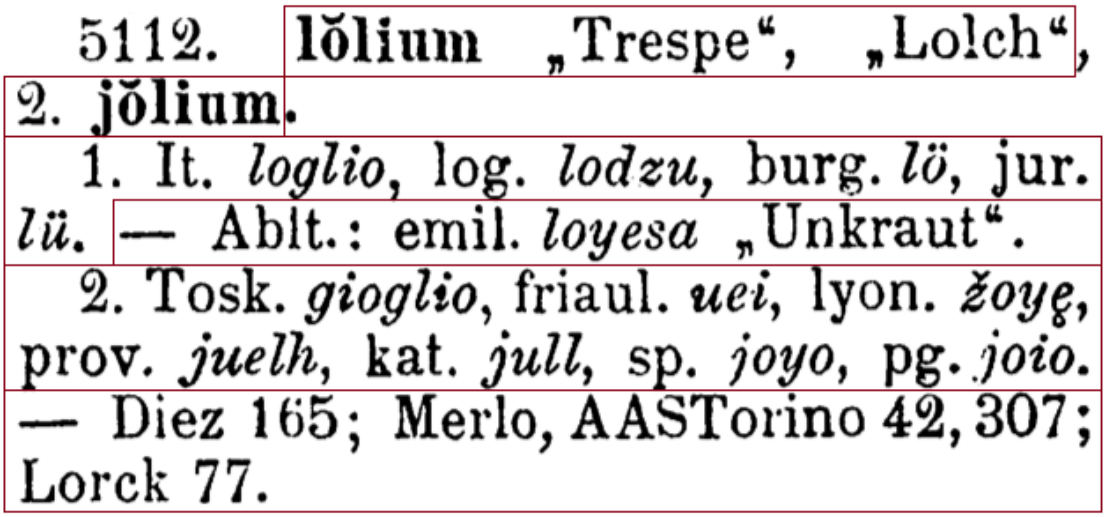

5112. **lŏlium** „Trespe“, „Lolch“, 2. **jŏlium**.
1. It. *loglio*, log. *lodzu*, burg. *lö*, jur. *lü*. — Ablt.: emil. *loyesa* „Unkraut“.
2. Tosk. *gioglio*, friaul. *uei*, lyon. *žoyę*, prov. *juelh*, kat. *jull*, sp. *joyo*, pg. *joio*. — Diez 165; Merlo, AASTorino 42, 307; Lorck 77.

Abbildung 6.4: Abschnitte des Wörterbuchtexts, die in der Datenbank als Eintrag in *record_lists* dargestellt werden am Beispiel von REW, S. 5112

part	sub_part	position	type	Quellentext
1		0	head	**lŏlium** „Trespe“, „Lolch“
2		0	head	**jŏlium**
1		1	main	It. *loglio*, log. *lodzu*, burg. *lö*, jur. *lü*.
1		2	extra	— Ablt.: emil. *loyesa* „Unkraut“.
2		1	main	Tosk. *gioglio*, friaul. *uei*, lyon. *žoye*, prov. *juelh*, kat. *jull*, sp. *joyo*, pg. *joio*.
2		2	text	— Diez 165; Merlo, AASTorino 42, 307; Lorck 77.

Tabelle 6.1: Indizierung der Elemente, die in Abb. 6.4 dargestellt werden

Die Spalte *type* gibt dabei die Art des Eintrags an und kann die folgenden Werte annehmen:

- **head**: Lemmaliste in der Kopfzeile.
- **main**: Hauptliste des Artikels. Diese ist in den meisten Artikel enthalten und gibt im Normalfall die direkten Erbwörter in den verschiedenen romanischen Sprachen, deren Bedeutung oftmals derer des Lemmas entspricht.
- **extra:** Gibt weitere Listen mit Sprachbelegen an, die beispielsweise Ableitungen, Zusammensetzungen oder ähnliches enthalten.
- **text**: Markiert einen natürlichsprachigen Abschnitt.
- **borrowing**: Wird für eine Liste von entlehnten Formen verwendet, die einer anderen Form zugeordnet sind.

Listen des letzten Typs sind dabei außerhalb der Indizierung, da sie kein eigener Grundbestandteil des Artikel sind, sondern immer einem anderen Sprachbeleg zugeordnet sind, der wiederum Teil einer *record_list* ist. Sie haben somit (wie die Einträge aus der Kopfzeile) die Position 0. Die Zuordnung zum Sprachbeleg erfolgt in der Tabelle *records* (s. u.).

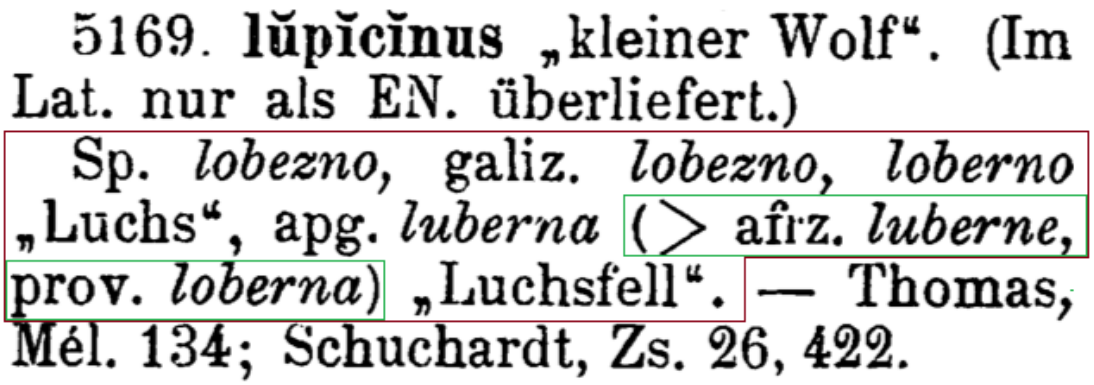

5169. **lŭpĭcĭnus** „kleiner Wolf“. (Im Lat. nur als EN. überliefert.)
Sp. *lobezno*, galiz. *lobezno*, *loberno* „Luchs“, apg. *luberna* (> afrz. *luberne*, prov. *loberna*) „Luchsfell“. — Thomas, Mél. 134; Schuchardt, Zs. 26, 422.

Abbildung 6.5: Liste mit Entlehnungen (grün). Sie ist strukturell Teil der umgebenden Liste (rot) und wird nicht indiziert.

Weitere Informationen, die vor allem der Rekonstruktion des Wörterbuchartikels dienen, sind in den Feldern *dash_separated* (Ob die Liste durch das Zeichen — abgetrennt wird), *specifier* (Potentielles einleitendes Element zu Beginn der Liste) und *bracketed* (Ob die Liste eingeklammert ist) enthalten. Zuletzt gibt es die Spalte *text*, die für „reguläre“ Listen einen eventuellen Kommentar enthält, der sich auf alle Belege bezieht, während sie bei Listen des Typs *text* den eigentlichen Inhalt enthält (für dessen genaue Repräsentation s. Kap. 6.5.3).

Diese Modellierung funktioniert in den meisten Fällen gut und bietet eine intuitive Sicht auf die Makrostruktur eines Artikels, in einem Fall ist sie allerdings in dieser Form nicht anwendbar. Dies ist so, wenn Sätze aus „regulären“ Beleglisten und natürlichsprachigen Anteilen zusammengesetzt sind. Zum Teil kann dies noch über die Darstellung des diskursiven Abschnitts als Listenkommentar über das Feld *text* modelliert werden (s. o.), in komplexeren Fällen, in denen sich mehrere natürlichsprachige und strukturierte Abschnitte abwechseln ist dies nicht der Fall. Falls dies vorkommt, wird also der Satz aufgespalten und die einzelnen Bestandteile als eigene *record_lists* modelliert.

In der Tabelle *records* wird die eine Hälfte der lexikalischen Kerndaten entsprechend Kapitel 2 abgelegt. Ein Sprachbeleg besteht hier aus Referenzen auf eine sprachliche Form, eine Bedeutung und eine Zeitperiode (in den seltenen Fällen, in denen es dazu eine entsprechende Angabe gibt). Die vierte Dimension, die geographische Zuordnung, ist nur indirekt über eine eventuelle räumliche Zuordnung der Sprachzuordnung der jeweiligen Form gegeben. Die Modellierung kann somit nicht hundertprozentig auf das

allgemeine Datenmodell abgebildet werden. Das wird so umgesetzt, um zu verhindern, dass sich das interne Modell zu weit von der Repräsentation im Quellenmaterial entfernt. Hier wird also die Sprachzuordnung einerseits als Teil der Definition einer sprachlichen Form betrachtet, anderseits als potentielle Quelle einer geographischen Lokalisierung (vgl. hierzu Kap. 11.2). In einer Datenstruktur, die exakt dem abstrakten Modell entspricht, würde beispielsweise ein italienischer Dialekt als Sprache auch die „Obersprache“ Italienisch erhalten, während die Angabe in der Quelle z.B. nach dem Schema „MA. von ...“ als rein geographische Angabe behandelt werden würde.

Zusätzlich werden die Sprachbelege wiederum in den Kontext der *record_list*, aus der sie stammen eingeordnet. Um die genaue Anordnung im Quellenmaterial abzubilden werden insgesamt vier hierarchisch funktionierende Indizes eingesetzt:

- **Index 1 (Subliste)**: Gibt die Nummer der Subliste des Belegs an. Sublisten werden meist durch Semikola getrennt, in manchen Fällen sind allerdings auch längere Formulierungen unter Verwendung natürlicher Sprache möglich (vgl. auch Kap. 5.3.3)

- **Index 2 (Position innerhalb der Subliste)**: Dieser Index wird über die Sprache definiert. Wenn für eine Sprache mehrere Formen bzw. Bedeutungen gegeben werden, werden diese auf dieser Ebene noch zusammengefasst. Kurzschreibweisen werden behandelt, als wären sie ausgeschrieben (d.h. Angaben wie „pg., sp. *astil“* werden wie „pg. *astil*, sp. *astil*“ als zwei Elemente indiziert).

- **Index 3 (Varianten innerhalb einer Sprache)**: Falls mehrere Formen bzw. Varianten einer Form angegeben werden, werden sie hier unterschieden. Dabei wird wiederum nicht zwischen abkürzenden Schreibweisen und expliziter Trennung durch Kommata im Originaltext unterschieden (also „pg. *(f)ata“* und „pg. *ata, fata“* würden beispielsweise identisch nummeriert).

- **Index 4 (Bedeutung)**: Falls eine Form mehrere Bedeutungen hat, werden diese durch den letzten Index nummeriert. Dabei spielt es keine Rolle, ob die Bedeutungen in der Quelle explizit angegeben sind oder inferiert werden müssen.

Im folgenden werden für einen einfachen Artikel die Indexe der jeweiligen Sprachbelege angegeben. Zu beachten ist dabei, dass die Lemmata als reguläre sprachliche Formen behandelt werden (vgl. auch Kap. 2.2.1), da diese jeweils durch eine solche repräsentiert werden. Die Lemmata, ihre Bedeutung und eventuelle weitere Angaben werden also ebenfalls als Sprachbelege abgebildet, die in einer *record_list* des Typs *head* enthalten sind.

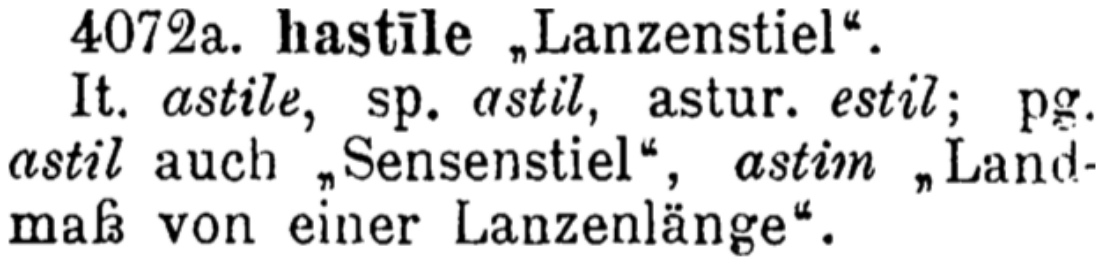

4072a. **hastīle** „Lanzenstiel“.
It. *astile*, sp. *astil*, astur. *estil*; pg. *astil* auch „Sensenstiel“, *astim* „Landmaß von einer Lanzenlänge“.

Abbildung 6.6: Eintrag 4072a aus dem REW

Sprachabkürzung	**Form**	**Bedeutung**	**Liste**	**Subliste**	**Position**	**Variante**	**Bedeutung**
lat.	hastīle	Lanzenstiel	head	0	0	0	0
it.	astile	Lanzenstiel	main	0	0	0	0
sp.	astil	Lanzenstiel	main	0	1	0	0
astur.	estil	Lanzenstiel	main	0	2	0	0
pg.	astil	Lanzenstiel	main	1	0	0	0
pg.	astil	Sensenstiehl	main	1	0	0	1
pg.	astim	Landmaß von einer Lanzenlänge	main	1	0	1	0

Tabelle 6.2: Beispiel für die Indizierung der Sprachbelege

Weiterhin speichert jeder Sprachbeleg zusätzliche Information wie das Trennzeichen zum jeweils vorherigen oder eventuelle Zusatzangaben, die diesem zugeordnet sind, so dass die originale Darstellung exakt reproduziert werden kann. Falls weitere Formen aus der im Beleg enthaltenen entlehnt werden, wird zusätzlich der Identifikator der entsprechenden Entlehnungsliste (s. o.) angegeben. Eine letzte wichtige Angabe ist die Unterscheidung, ob der Beleg ein strukturelles Element der Liste darstellt oder in einem diskursiven Element enthalten ist. Beide Klassen werden separat (nach dem oben vorgestellten vierstufigen System) nummeriert. Jeder Belegliste werden somit ihre eigentlichen Belege[7] und in natürlichsprachigen Elementen vorkommende Sprachbelege zugeordnet. Die Belege im Text können dabei grundsätzlich in allen diskursiven Elementen auf den verschiedenen Ebenen vorkommen (vgl. Kap. 6.5.3).

Die Tabellen *ling_forms* und *meanings* enthalten entsprechend ihrer Name die Daten der Hauptbestandteile der Sprachbelege, nämlich der sprachlichen Formen und Bedeutungen. Erstere besteht im Grunde aus einer Sprachzuordnung, der textuellen Darstellung der eigentlichen Form und weiteren grammatikalischen Informationen. Eine genauere Betrachtung insbesondere im Bezug auf potentielle Homonyme findet sich in Kapitel 9. Eine detaillierte Betrachtung der Behandlung von Bedeutungen findet sich in Kapitel 10.

[7] Im Fall einer reinen *text*-Liste sind keine solchen vorhanden

Literaturverweise beziehen sich entweder auf einen Beleg (wenn sie diesem folgen) oder die gesamte Belegliste (wenn sie in diskursiven Elementen innerhalb der Liste enthalten sind, vgl. auch Kap. 6.5.3). Entsprechend werden sie über die Tabellen *c_records_lit_references* und *c_record_lists_lit_references* der jeweiligen Entität zugeordnet.

6.5.1 Etymologische Relationen

Die Definition der Sprachbelege ergibt sich gewissermaßen natürlich aus deren expliziter Angabe im Wörterbuchartikel. Der zweite Datentyp des in Kapitel 2 aufgestellten Modells, nämlich Relationen zwischen den einzelnen Formen, steht zum großen Teil gewissermaßen außerhalb der Darstellung des Artikels, da ein Großteil der etymologischen Relationen in diesem nicht explizit enthalten sind. Entsprechend der Information, die das REW liefert werden dabei fünf Typen von Relationen unterschieden (von denen die ersten vier etymologischer Natur sind):

- **predecessor**: Die Vorgänger-Relation ist einerseits die häufigste Relation, die angibt, dass ein Wort von einem anderen abstammt, andererseits kann sie nur aus der Anordnung der Belege innerhalb der einzelnen Artikel erschlossen werden (vgl. Kap. 7.2.3). Die Vorgängerrelation ist dabei immer zwischen einem Lemma und einer (meist romanischen) Form im Hauptteil des Artikels definiert. Eine Unterscheidung, ob es sich um einen direkten Vorgänger oder einen komplexeren Herleitungsweg handelt, kann hierbei nicht gemacht werden. Für alle sprachlichen Formen, die nicht in einer Relation vom Typ *borrowing* oder *derivation* enthalten sind, wird eine solche unspezifischere Relation angelegt.
- **borrowing**: Entlehnungen werden durch die durch die Notation (> ...) direkt angegeben. Dieser Typ von Relation kann dabei am einfachsten und sichersten hergeleitet werden.
- **derivation**: Diese Relationen kommen nur selten vor und werden aus Strukturelementen wie „daraus“ oder ähnlich erschlossen, die gleichsprachliche Formen trennen.
- **contamination**: Diese Relation wird verwendet, wenn ein zweites Etymon angegeben wird. Diese kann entweder direkt Teil eines Sprachbelegs sein oder vorab für eine vollständige Teilliste angegeben werden (vgl. Kap. 7.2.2). Die Unterscheidung dieser Relation ist hauptsächlich für die korrekte technische Verarbeitung nötig, an der Oberfläche des Webportals wird sie wie eine Relation vom Typ *predecessor* behandelt. Es ist möglich, dass in seltenen Einzelfällen auch Relationen enthalten sind, die nicht aus einer Kontamination im eigentlichen Sinne stammen (z.B. wenn eine zweites Etymon für ein Kompositum angegeben wird, was aber kaum vorkommt).

- **flexion_forms**: Hiermit werden verschiedene Flexionsformen des gleichen Lexems miteinander verknüpft. Aktuell ist dies nur der Fall für verschiedene Kasusformen bei den lateinischen Lemmata.

Alle Relationen werden dem Wörterbuchartikel zugeordnet, aus dem sie erschlossen wurden. Dieser fungiert somit einerseits gewissermaßen als Quellenangabe, andererseits sind sie so Teil der Versionierung. Gerade bei zusammengesetzten Formen oder Kontaminationen kann es vorkommen, dass verschiedene oder auch identische Relationen zu dieser Form in verschiedenen Artikeln vorkommen. Es sind allerdings auch Fehler im Quellenmaterial möglich, bei denen eine identische Form[8] in mehren Artikeln auftaucht und deshalb fälschlicherweise mehreren Etyma zugeordnet wird. Dieser Fall muss wiederum unterschieden werden von mehreren alternativen Etyma. Falls beispielsweise mehrere Lemmata vorkommen (und die Zuordnung nicht durch eine entsprechende Nummerierung deutlich gemacht wird), kann die korrekte Relation in den meisten Fällen nicht sicher hergeleitet werden. Somit werden zwei (oder mehr) alternative Relationen erstellt, die in der Datenbank über die Verwendung einer Spalte *id_alternative* miteinander verknüpft sind.

6.5.2 Unsichere Angaben aus der Quelle

In einem stark strukturierten Datenmodell ist die Behandlung von unsicheren Angaben, die oftmals sehr informell im Quellentext angegeben werden, nicht immer trivial. Deshalb findet in diesem Kapitel eine kurze Betrachtung entsprechender Lösungsansätze statt. Im REW kommen drei verschiedene Arten von Unsicherheiten vor[9]:

- **Unsichere Sprachzuordung**: Dabei können verschiedene Fälle unterschieden werden: Einerseits wird im REW zum Teil bei Lemmata gar keine Sprache angegeben[10], andererseits werden zum Teil mehre Sprachzuordnungen, die durch ein „oder“ getrennt sind, oder eine Sprache mit Fragezeichen verwendet. Eine eindeutige Zuordnung einer Sprache ist nur im letzten Fall möglich. Hierbei wird die Zuordnung durch ein spezielles Flag als unsicher markiert. In den anderen beiden Fällen wird in der Datenbank eine Pseudo-Sprache „Unbekannt“ zugeordnet, wobei im Fall der verschiedenen Alternativen diese als Kommentar bezogen auf die Form behandelt wird.
- **Unsichere Bedeutung**: Eine Fragezeichen bei einem Beleg in der Kopfzeile nach einer Bedeutung wird als unsichere Bedeutungszuweisung interpretiert. Dies wird ebenfalls explizit in der Datenbank als unsicher markiert.

[8] Für eine Betrachtung der Identität bei sprachlichen Formen siehe Kapitel 9

[9] Damit sind explizit in der Quelle als solche angegebene Unsicherheiten gemeint und nicht solche die im Vergleich mit anderen Quellen auftreten

[10] Dies wird im Normalfall über den Zusatz „Woher?“ markiert, da eine fehlende Sprachangabe ansonsten eine lateinische Sprachzuordnung bedeutet.

- **Unsichere Etymologie**: Ein Fragezeichen nach einem Sprachbeleg im Hauptteil wird als unsichere Etymologie interpretiert. Hierbei wird somit die etymologische Relation als unsicher markiert. Selbes gilt für einleitende Elemente wie „vielleicht auch“ o.ä.

6.5.3 Behandlung von diskursiven Elementen

Alle unstrukturierten (also natürlichsprachigen) Bestandteile werden in der Datenbank in Textfeldern abgelegt. Dabei stellt sich allerdings die Frage wie mit den in Kap. 5.3.5 innerhalb dieser Elemente erkannten Entitäten umgegangen werden soll. Im relationalen Modell sind diese nur schwer umsetzbar, es wäre allerdings wünschenswert, dass beispielsweise Sprachbelege oder Literaturverweise, die in einem bestimmten Kontext vorkommen, auch über entsprechende Abfragen auf der Datenbank angesprochen werden können. Somit wurde hier ein hybrides System verwendet, bei dem die erkannten Elemente einerseits im relationalen Modell abgebildet werden und andererseits im Text durch spezielle XML-Tags markiert werden. Die folgende Tabelle gibt einen Überblick über die vorkommenden Entitäten:

Entität	**XML-Tag**	**XML-Attribute**	**Relationale Abbildung**
Allgemeine Abkürzungen	abbr	ID der Abkürzung in der Tabelle *abbreviations*	—
Sprachabkürzungen	lang	ID der Sprachabkürzung, geographischer und zeitlicher Präfix, Groß-/Kleinschreibung	—
Bedeutungen(en)	meaning	Start- und Endindex der Bedeutungen	Tabelle *c_text_lists_meanings*
Sprachbeleg(e)	list	Start- und Endindex der Belege, Groß-/Kleinschreibung	Eintrage in *records*, die als *text* markiert sind
Literaturverweis	lit-ref	Index des Literaturverweises	Tabelle *c_record_lists_lit_references*
Verweis auf Artikel	entry-ref	Index des Verweises, sprachliche Form (falls gegeben)	Tabelle *entry_references*

Tabelle 6.3: Formen von Entitäten in unstrukturierten Elementen und ihre Umsetzung

Alle XML-Tags enthalten weiterhin das Attribut *text*, welches den genauen Wortlaut im Ursprungstext wiedergibt, aus dem dieser XML-Tag erzeugt wurde. Die Erstellung einer rein textuellen Repräsentation ist also jederzeit möglich.

Im folgenden wird die Verwendung am Beispiel eines Satzes mit Zusatzinformationen verdeutlicht. Der ursprüngliche Satz lautet:

> Die Bedeutung ist z. T. „viel“ Bertoni, AR. 1, 506, heute dringt namentlich in Tirol mehr und mehr it. *assai* ein Gartner, G.Gr. 1, 602.(REW, S. 53)

Daraus wird die folgende XML-annotierte Repräsentation erzeugt:

```
Die Bedeutung ist <abbr id="298" text="z. T." /> <meaning first-index="0" last-index="0" text=
```

Die Indexe sind dabei jeweils auf die in der Datenbank dieser Belegliste zugeordneten Elemente bezogen, die Bedeutung „viel“ ist beispielsweise die erste (und einzige) Bedeutung, die in dieser Liste vorkommt und hat somit den Index 0.

Werden Sprachbelege bzw. Formen im Text genannt, ist diesen oftmals keine explizite Bedeutung zugeordnet. Zum Teil kann sie nur aus der Formulierung der jeweiligen Textpassage erkannt und nicht vom Algorithmus hergeleitet werden, zum Teil wird auch gar keine angegeben. Solche Belege werden mit einer Pseudo-Bedeutung „???“ verknüpft[11].

6.6 Hilfstabellen in der Datenbank

Eine besondere Kategorie von Tabellen in der Datenbank sind die sogenannten Hilfstabellen. Diese werden automatisch aus anderen Tabellen generiert und mit dem Präfix „a_“ markiert. Sie enthalten immer die aktuellste Version der entsprechenden Daten. Dies wird einerseits bei den Zeilen angewandt, bei denen die Tabelle *a_lines* immer den gerade aktuellen Zeileninhalt unter Berücksichtigung aller Korrekturen enthält. Andererseits werden im Kontext der Artikel Kopien der relevanten Tabellen erstellt, die nur die Einträge enthalten, die der jeweils aktuellsten Artikelversion zugeordnet sind. Bei geteilten Tabellen werden nur diejenigen Einträge gelistet, die in mindestens einer aktuellen Artikelversion referenziert werden. Inhaltlich entsprechenden die Hilfstabellen den Originaltabellen (insbesondere sind auch die Identifikatoren identisch), wobei auf bestimmte Verwaltungsinformationen verzichtet wird.

Der Grund für die Verwendung dieser Tabellen ist eine höhere Effizienz und geringere Fehleranfälligkeit der SQL-Abfragen auf der Datenbank, die meistens nur auf den gerade aktuellen Daten arbeiten. Als Beispiel soll hier eine Abfrage dienen, die die Etyma der französischen Form *beaucoup* bestimmen soll. Wird diese auf den Grundtabellen ausgeführt ist sie sehr komplex, da sichergestellt werden muss, dass die verwendeten sprachlichen Formen aktuell und die verwendeten Relationen aktuell sind (d.h. mit einer aktuellen Artikelversion verknüpft sind). Wird eine der beiden Bedingungen vergessen, werden zusätzliche veraltete Angaben mit zurückgegeben.

[11] Auch Formen in strukturierten Passagen werden mit dieser markiert, wenn die Inferenz der Bedeutung fehlschlägt, vgl. Kap. 7.2.1

```
SELECT DISTINCT f.id_form, f.id_lang, e.id_form as id_form_2, e.id_lang as id_lang_
FROM ling_forms f
     JOIN languages l ON f.id_lang = l.id_lang
     JOIN records USING (id_form)
     JOIN record_lists USING (id_record_list)
     JOIN entries en USING (id_entry)
     LEFT JOIN ling_relations lr ON id_form_1 = f.id_form
     LEFT JOIN entries en2 ON lr.id_entry = en2.id_entry
     LEFT JOIN ling_forms e ON id_form_2 = e.id_form
WHERE en2.replaced IS NOT NULL
     AND en.replaced IS NOT NULL
     AND l.id_lang_abbr = 394
     AND f.form COLLATE utf8_general_ci = 'beaucoup'
```

Wenn jeweils die entsprechenden Hilfstabellen verwendet werden, verkürzt sich die Abfrage deutlich:

```
SELECT DISTINCT f.id_form, f.id_lang, e.id_form as id_form_2, e.id_lang as id_lang_
FROM a_ling_forms f
     JOIN a_languages l ON f.id_lang = l.id_lang
     LEFT JOIN a_ling_relations lr ON id_form_1 = f.id_form
     LEFT JOIN a_ling_forms e ON id_form_2 = e.id_form
WHERE l.id_lang_abbr = 394
     AND f.form COLLATE utf8_general_ci = 'beaucoup'
```

Anhand des Beispieles kann man auch gut erkennen, warum nicht nur für die Basistabellen entsprechende Hilfstabellen erstellt werden. Je nach Art der Anfrage sind auch die restlichen Tabellen hilfreich. Insbesondere gilt dies für die geteilten Tabellen, für die die Bestimmung der Aktualität zum Teil sehr aufwendig ist, da sie von verschiedenen anderen Tabellen potentiell referenziert werden können.

Anmerkung: Neuere Datenbanksysteme unterstützen gewisse Formen der Versionierung zum Teil nativ (vgl. z.B. Huang u. a. 2017, MariaDB: System-Versioned Tables). Dabei könnte diese Funktionalität zum Teil direkt vom Datenbanksystem übernommen werden, indem eine aktuelle Partition der Tabelle anstelle der besprochenen Hilfstabellen verwendet werden. Gerade bei komplexen Fällen mit vielen miteinander verknüpften Tabelle ist allerdings eine vollständig automatisierte Umsetzung eines Versionierungskonzept schwierig.

7 Umwandlung in relationale Daten

In diesem Kapitel wird schließlich der letzte Teil der in Kap. 3.3 beschriebenen Prozesskette näher besprochen. Die Ausgangsdaten der Operationen aus Kapitel 5 bestehen aus einer abstrakten Repräsentation des Artikelaufbaus, die nun entsprechend der Struktur des vorherigen Kapitels in relationale Daten umgewandelt werden muss. Die Schwierigkeit liegt dabei weniger darin aus den hierarchisch strukturierten Daten tabellarische zu erzeugen, sondern in der Erstellung von „expliziten“ Daten, in denen Lücken, abkürzende Schreibweisen und ähnliche Konventionen aufgelöst sowie kontextabhängige und inkonsequente Formulierungen angeglichen werden. Ein einfaches Beispiel hierfür ist die Herleitung von Bedeutungszuordnungen, die im Quellenmaterial nicht direkt angegeben sind.

Auch hier soll eine möglichst formalisierte Vorgehensweise angewandt werden, um die Komplexität möglichst zu verringern und eine eventuelle Nachnutzung zu erleichtern, auch wenn dies ungleich schwieriger ist als im Fall der strukturellen Erfassung. Der Strukturbaum wird analog zum Vorgehen klassischer Implementierungen zur Verarbeitung von hierarchisch strukturierten Daten (beispielsweise XML-Parsern) mit Hilfe einer Tiefensuche[1] abgearbeitet. Jeder Knoten des Baums wird dabei zweimal verarbeitet, einmal vor der Verarbeitung der Kinderknoten und einmal nachdem diese behandelt wurden. Die folgende Abbildungen illustrieren dieses Vorgehen am Beispiel eines einzelnen Belegs:

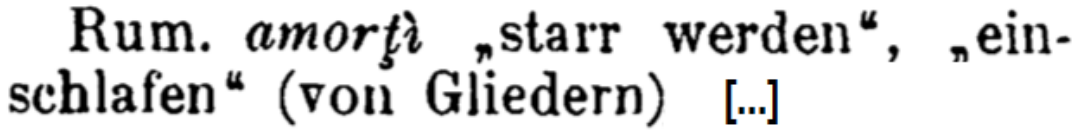


Abbildung 7.1: Ausschnitt aus REW, S. 186

[1] https://de.wikipedia.org/wiki/Tiefensuche

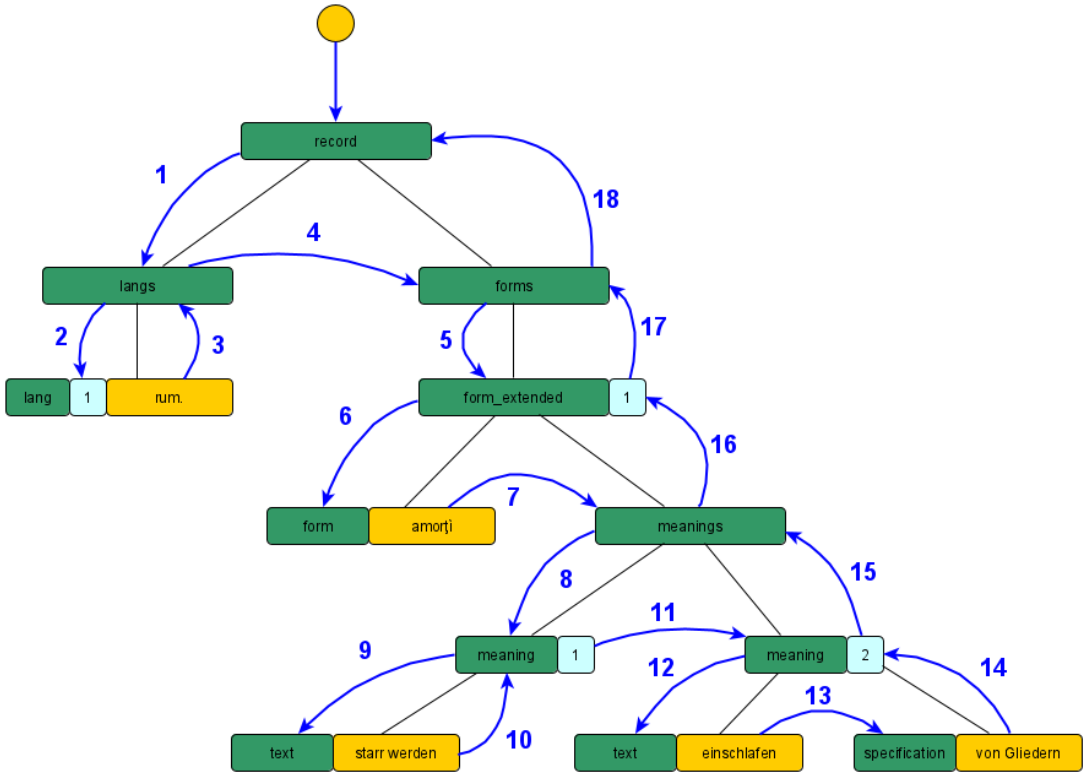


Abbildung 7.2: Verarbeitungsreihenfolge des Strukturbaums für den gegebenen Ausschnitt

Beim Verarbeiten der jeweiligen Blätter des Baumes werden entsprechende Zustandsvariablen gesetzt, die dann auf den höheren Ebenen zu komplexeren Elementen zusammengesetzt werden, sodass einzelne Datenzeilen entstehen, die das Resultat dieses Arbeitsschritts bilden. Diese können im Anschluss mit bestehenden Zeilen verglichen und in die Datenbank eingefügt werden, falls Änderungen zur aktuellen Version vorliegen (vgl. Kap. 3.4.1). So wird in der obigen Abbildung beispielsweise in Schritt 12 der Wert für den Text einer Bedeutung gesetzt, in Schritt 13 wird deren genauere Spezifizierung gesetzt, während in Schritt 14 beides zu einem Eintrag für die Tabelle *meanings* kombiniert wird. Analog wird im letzten Schritt (18) schließlich ein Eintrag für die Tabelle *records* erstellt, der die untergeordneten Daten verknüpft. Das Ergebnis dieses Teilschritts sind insgesamt die folgenden Datenzeilen:

```
[
   6: ["languages", {
         "id_lang_abbr": 890,
         "id_time_period": null,
         "geo_spec": null
         }
      ],
```

```
7: ["ling_forms", {
      "form": "amor ì",
      "id_lang": ###6###,
      "lang_unsure": false,
      "learned_word": false,
      "reconstructed": false, ...
   }
],
8: ["meanings", {
      "text": "starr werden",
      "specification": "",
      "meta": false
   }
   ],
9: ["meanings", {
      "text": "einschlafen",
      "specification": "von Gliedern",
      "meta": false
   }
],
10: ["records", {
      "id_record_list": ###81###,
      "sub_list_index": 0,
      "list_entry_index": 0,
      "variant_index": 0,
      "meaning_index": 0,
      "separator": null,
      "id_form": ###7###,
      "id_meaning": ###8###,
      "id_time_period": null, ...
   }
],
11: ["ling_relations", {
      "type": "predecessor",
      "id_entry": ###0###,
      "id_form_1": ###7###,
      "id_form_2": ###2###,
      "unsure": null,
      "id_alternative": null
   }
],
12: ["records", {
      "id_record_list": ###81###,
      "sub_list_index": 0,
```

```
            "list_entry_index": 0,
            "variant_index": 0,
            "meaning_index": 0,
            "separator": null,
            "id_form": ###7###,
            "id_meaning": ###9###,
            "id_time_period": null, ...
        }
    ],
]
```

Auch hier werden drei # zur Markierung von Referenzen auf andere Zeilen verwendet. Zum Teil werden auch Informationen aus höheren Bearbeitungsebenen verwendet, wie die Postion des Belegs in der Belegliste oder die Referenz auf das Lemma in der etymologischen Relation.

Das relativ formalisierte Vorgehen mit Hilfe der Zustandsvariablen für die Ausprägungen der einzelnen Entitäten hilft eine Basisstruktur und gerade im Zusammenspiel mit den durchaus komplexen Inferenzroutinen, die in den folgenden Teilkapiteln besprochen werden, eine gewisse Übersichtlichkeit im Programmcode zu erhalten. Zusätzlich ist es relativ robust bei Anpassungen der formellen Grammatik, die zu Änderungen im Aufbau des Strukturbaums oder zum Einfügen von bereits vorhandene Elemente an zusätzlichen Stellen führen, da die entsprechenden Knoten grundsätzlich unabhängig[2] von ihrer Position im Baum behandelt werden. Problematisch ist es allerdings beim rekursiven Auftreten bestimmter Elemente, also wenn eine bestimmte Entität innerhalb ihrer eigenen Verarbeitung erneut auftritt. Ein Beispiel ist das Vorkommen von zusätzlichen Sprachbelegen innerhalb der Beschreibung des eigentlichen Belegs:

Abbildung 7.3: Verschachtelung von zwei Sprachbelegen in REW, 1573a

Für Elemente, bei denen dieser Fall auftreten kann, wird die Verarbeitung in einer Kopie des eigentlichen Parsers vorgenommen und die daraus entstandenen Datenzeilen zum Hauptresultat hinzugefügt. Die erstellte Bibliothek stellt für diese und andere grundlegende Funktionen wie das Einfügen und Löschen einer Zeile oder das Suchen innerhalb der aktuellen Resultatdaten eine Reihe abstrakter Operationen zur

[2] Für bestimmte Operationen ist ein gewisser Kontext trotzdem nötig. Beispielsweise werden fehlende Sprachabkürzungen in der Kopfzeile anders behandelt als im Artikeltext, vgl. Kap. 7.2.1

Verfügung.

Zusätzlich zu konstanten Spaltenwerten sind Referenzen auf andere Zeilen (vgl. Kap. 3.4 und Platzhalter möglich. Letztere dienen dazu Werte zu markieren, die an der aktuellen Position in der Verarbeitung des Baumes noch nicht gesetzt werden können, weil dazu Information benötigt wird, die erst an einer späteren Stelle vorkommt. Die Platzhalter werden entweder nach Abschluss der Gesamtverarbeitung des Baumes oder beim Erreichen konkreter Elemente durch die endgültigen Werte ersetzt. Ein Beispiel für die Verwendung eines solchen Platzhalters ist die Behandlung der Bedeutung von entlehnten Formen. Wird am Ende einer Entlehnungsliste eine Bedeutung angegeben, so bezieht sie sich auf alle vorangegangenen Belege. Ist eine solche nicht vorhanden, wird die Bedeutung der Form vererbt, auf deren Basis die Entlehnung stattfand (vgl. Abb. 7.4). Zum Zeitpunkt der Behandlung eines einzelnen Belegs ist allerdings weder die eine noch die andere Information vorhanden, sodass die Bedeutung einer entlehnten Form durch einen Platzhalter ersetzt und erst nach vollständiger Verarbeitung der Entlehnungsliste und des Ausgangsbelegs nachgetragen wird.

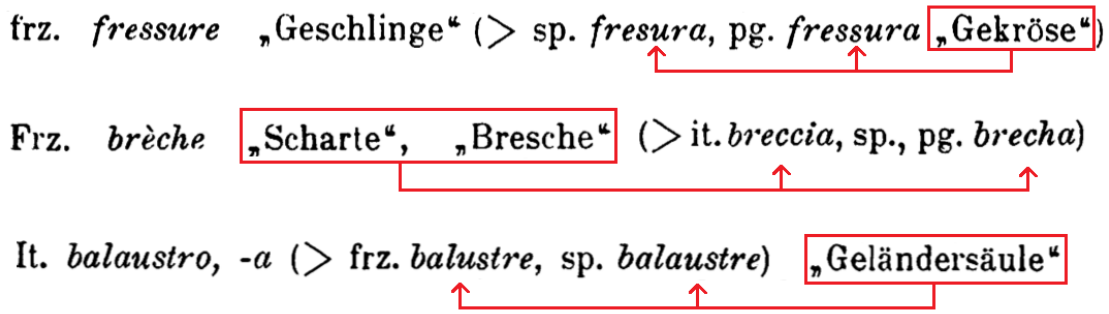


Abbildung 7.4: Bedeutungszuordnung für entlehnte Formen in verschiedenen Varianten Eventuell auftretende Spezialfälle, wie die Zuordnung von mehreren Bedeutungen zu einem Sprachbeleg mit einem Platzhalter werden vom Parser automatisch behandelt, in diesem Fall durch das Duplizieren der Zeilen. Zusätzlich wird nach Ende des eigentlichen Verabeitungsprozesses ein weiterer Nachbesserungsschritt durchgeführt, in welchem vor allem Duplikate in den Resultatdaten entfernt werden und die Referenzen der Zeilen entsprechend angepasst werden. Im Verlauf der Abarbeitung des Baumes ist die korrekte Erkennung gerade beim Vorhandensein von Platzhaltern nicht immer einfach, somit wird diese systematisch am Ende durchgeführt.

7.1 Angleichung von unterschiedlichen Notationen und Auflösung abkürzender Schreibweisen

In traditionellen Quellentexten treten häufig Notationen auf, die für menschliche Lesende einfach verständlich, für eine maschinelle Verarbeitung allerdings

problematisch sind. Dabei können vor allem die folgenden Fälle unterschieden werden:

- Globale Abkürzungen, die in verschiedenen (meist nicht im Verzeichnis aufgeführten) Varianten vorkommen
- Abkürzende Schreibweisen mit Klammern
- Abkürzende Schreibweisen, die nur im direkten Kontext verständlich sind

Erste werden grundsätzlich quellentreu in der Datenbank abgelegt, d.h. verschiedene Varianten von Abkürzungen werden als explizite Datensätze angelegt und mit den jeweiligen Vorkommen verknüpft, um nachträgliche Korrekturen einfach zu ermöglichen. Bei der Bündelung verschiedener Abkürzungsvarianten kann nie von absoluter Richtigkeit ausgegangen werden, spätere Zusammenführungen oder auch Trennungen fälschlich als gleichbedeutend behandelter Notationen sind jederzeit möglich. Im Artikelkontext werden ebenfalls die originalen Abkürzungen verwendet, während sie beispielsweise für die aggregierenden Seiten (vgl. Kap. 12.3) entsprechend zusammengefasst werden.

Die anderen beiden Kategorien werden vor dem Import entsprechend angeglichen d.h. aus geklammerten Varianten werden mehrere Einzeleinträge erstellt, während kontextsensitive Information nach Möglichkeit entsprechend um- oder ausformuliert wird.

7.1.1 Auflösung von Klammerungen

DIe Aufspaltung von geklammerten Varianten ist in den meisten Fällen wenig komplex, die Anzahl der Varianten verdoppelt sich pro Klammerung:

„*š(u)a(r)*“ → *ša, šua, šar, šuar*

Gerade im Fall von Varianten im Abkürzungsverzeichnis müssen allerdings unterschiedliche Fälle betrachtet werden

1. Reguläre Varianten, z.B. „alb(an). = albanesisch“. Dies entspricht dem Beispiel oben:

abbreviation	prefix	language	specification
alban		albanesisch	
alb		albanesisch	

2. Klammerung als Spezifikation, z.B. „alb. (piem.) = MA. von Albi (Piemont)“

abbreviation	prefix	language	specification
alb. (piem.)	MA. von	Albi	Piemont

3. Klammerung mit vollständiger Variante, z.B. „scipr. (scypr.) = MA. von S. Ciprian“

abbreviation	prefix	language	specification
scipr.	MA. von	S. Ciprian	
scypr.	MA. von	S. Ciprian	

Während die der erste Fall syntaktisch leicht unterschieden werden kann, wird zur Unterscheidung von 2 und 3 die Ähnlichkeit des geklammerten Bestandteils mit der vorangegangene Abkürzung betrachtet.

Zum Teil sind aber auch nicht alle Varianten beim Auflösen einer Klammer sinnvoll. Ein Beispiel hierfür ist die bibliographische Abkürzungen „MSL(P)(aris)“. Für diese werden nur die Varianten „MSL“, „MSLP“, „MSLParis“ erzeugt, da „MSLaris“ offensichtlich keinen Sinn ergibt. Eine automatisierte Erkennung ist hier allerdings nicht bzw. nicht ohne erheblichen Aufwand möglich, sodass eine entsprechende Ausnahme (vgl. Kap. 7.3) verwendet wird. Ähnliches gilt für die Überschneidung von Abkürzungsvarianten. So treten im REW sowohl „SNP.“ als auch „S(t)NP(h).“ auf. Für letztere wird also die Variante „SNP.“ explizit über eine Ausnahme ausgeschlossen.

7.1.2 Varianz und Abkürzungen in Bedeutungsangaben

In diesem Abschnitt wird die Verarbeitung von Bedeutungsangaben behandelt, die Inferenz von nicht explizit angegeben Bedeutungen findet sich in Kap. 7.2.1. Im Bezug auf die Zeichenketten der vorhandenen Bedeutungen im REW gibt es zwei Problematiken zu beachten.

Zum einen werden mehrere Bedeutungen auf zwei verschiedene Arten angegeben. Einerseits werden sie (konsequent) als einzelne durch Kommata getrennte Zeichenketten in Anführungszeichen dargestellt:

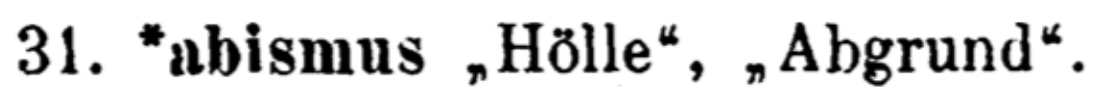


Abbildung 7.5: Kopfzeile von REW, S. 31

Andererseits werden mehrere Varianten innerhalb von Anführungszeichen aufgezählt:

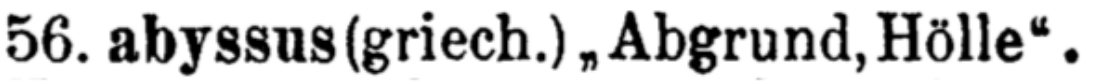


Abbildung 7.6: Kopfzeile von REW, S. 56

Somit müssen die Zeichenketten der zweiten Form aufgespalten werden. Dazu müssen diese allerdings von solchen unterschieden werden, die einen Nebensatz (z.B. „Schaf, das zu früh geworfen hat“ (REW, S. 37)) oder eine Aufzählung von Adjektiven (z.B. „großer, flacher Korb“ (REW, 1052a)) enthalten. Letzteres kann zumindest in den häufigsten Fällen durch einen regulären Ausdruck dargestellt werden, während für

ersteres eine Liste von Relativpronomen, Relativadverbien, Präpositionen und Subjunktionen geführt, die einen Nebensatz einleiten können. Somit kann ein großer Teil der Bedeutungsangaben erkannt werden, die nicht am Komma aufgespalten werden müssen. Solche, die trotzdem unerkannt bleiben, können wiederum durch eine Ausnahme (vgl. Kap. 7.3) korrigiert werden.

Weiterhin gibt es das Problem, dass manchen Bedeutungsangaben sich auf vorherige Bedeutungen oder den Kontext des Artikels beziehen und nicht ohne diese verständlich sind (vgl. auch das Beispiel in Kap. 3.2). Dies tritt in verschiedenen Varianten auf, beispielsweise:

- Abkürzungen mit Bindestrichen, z.B. „Maulesel“, „-in“ (REW, 2884a)
- Abgekürzte Aufzählungen, z.B. „das Licht löschen, dämpfen“ (REW, S. 185)
- Referenzen auf vorherige Bedeutungen, z.B. „Kette, an der Baumstämme befestigt werden“, „Schlitten mit solcher Kette“ (REW, S. 2310)

Alle solche Vorkommen werden mit zwei verschiedenen Arten von Ausnahmen behandelt, die Details finden sich in Kap. 7.3.

7.1.3 Abgekürzte sprachliche Formen

Eine verhältnismäßig häufige Konvention ist die Verwendung von Bindestrichen, um sehr ähnliche Formen anzugeben (beispielsweise männliche und weibliche Formen oder verschiedene Fälle). Auch hier wird eine einfache Routine verwendet, die häufige Varianten abdeckt:

- Besteht die Abkürzung nur aus einem Vokal und die vorherige Form endet auf einen Vokal, wird dieser ausgetauscht: *adolo, -a → adolo, adola*
- Besteht die Abkürzung nur aus einem Vokal und die vorherige Form endet auf einen Konsonanten, wird der Vokal angehängt: *andron*, *-a → andron*, *androna*
- Ansonsten wird die Stelle gesucht, an der der bzw. die Vokale(e), mit denen die Abkürzung beginnt[3] ohne Berücksichtigung von Diakritka vorkommen (falls ein anderer Vokal gefunden wird, kann die Abkürzung nicht aufgelöst werden): auditio, -ōne → *auditio*, *auditiōne* oder *biquet*, *-ette → biquet, biquette*

Abkürzungen, bei denen keine der Regeln anwendbar ist (oder bei denen sie im Ausnahmefall zu falschen Ergebnissen führen wie *pačara, -arela → pačara, pačararela*)

[3] Abkürzende Schreibweisen, die mit einem Konsonanten beginnen, kommen im REW nicht vor und werden deshalb hier nicht behandelt.

werden durch zwei verschiedene Typen von Ausnahmen behandelt. Zum einen gibt es eine globale sprachabhängige Variante *resolve_dash_lang*, die folgende Form von Regeln enthält:

```
{
   "value_in": {
      "lang_abbr":"lat.",
      "suffix":"a",
      "prev_suffix":"us"
   }
   "value_out": 2
}
```

Der Wert unter *value_out* steht für die Anzahl der Zeichen, die aus der vorangehenden Form entfernt werden, bevor die Endung angefügt wird. Für lateinische Formen wird also die Kombination *...us, -a* durch ein einfaches *-a* (anstatt sonst *-usa*) ersetzt, z.B. *affĕctus, -a → affĕctus, affĕcta.*

In sonstigen Fällen werden einzelfallbasierte Regeln des Typs *resolve_dash* erstellt:

```
{
   "value_in": ["apex", "-ĭce"]
   "value_out": "apĭce"
}
```

Beide Varianten werden möglichst kontextunabhängig definiert, um eine Wiederverwendbarkeit[4] zu gewährleisten. Im letzteren Fall hat das allerdings den (theoretischen) Nachteil, dass es grundsätzlich möglich ist, dass eine solche Regel für eine Sprache gültig ist und für eine andere (mit orthographisch identischen Formen) nicht. In einem solchen Fall müsste eine lokale auf den Artikelkontext bezogenen Ausnahme des Typs *form_data* (s. Kap. 7.3) erstellt werden, um ein richtiges Ergebnis zu erhalten. In der Praxis trat dieses Problem allerdings nicht auf.

7.2 Auflösung von impliziter Information

7.2.1 Herleitung von ausgelassenen Elementen

Die wohl häufigste Form von „impliziter Information“ (mit Ausnahme der Relationen) ist die Auslassung bestimmter Angaben. Dieses Teilkapitel beschäftigt sich mit

[4] Auch evtl. als Traingsdaten für Ansätze des maschinellen Lernens, vgl. Kapitel 14.

Auslassungen von Bedeutungsangaben und Sprachabkürzungen, während das folgende Kapitel die Auslassung von Formen als einen Spezialfall enthält.

Für die Auslassung von Sprachen in der Kopfzeile, gilt im REW die Konvention, dass eine fehlende Angabe für ein lateinisches Lemma steht. Eine Ausnahme sind hier die Formen mit unsicherere Sprachzuordnung, die im Normalfall durch ein zusätzliches „Woher?“ markiert sind (vgl. auch Kap. 6.5.2). In den strukturierten Passagen des Hauptteils tritt eine Auslassung der Sprachzuordnung im Normalfall nur in sehr einfachen Fällen auf:

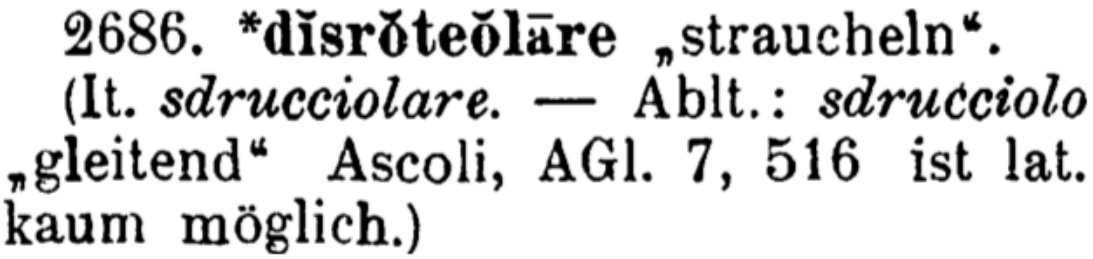

2686. *dĭsrŏteŏlāre „straucheln“. (It. *sdrucciolare*. — Ablt.: *sdrucciolo* „gleitend“ Ascoli, AGl. 7, 516 ist lat. kaum möglich.)

Abbildung 7.7: Artikel im REW, in dem eine Sprachangabe ausgelassen wird

Somit kann als Sprachzuordnung, die der letzten Form (mit Ausnahme von Formen in diskursiven Elementen) verwendet werden. In diskursiven Elementen findet wiederum sehr häufig eine Auslassung der Sprachabkürzung statt. Ohne eine Form der inhaltlichen Analyse der natürlichsprachigen Passage, kann hier nur in einfachen Fällen eine Sprachzuordnung stattfinden. Dies ist der Fall, wenn es genau eine orthographisch gleiche Form in den strukturierten Passagen gibt, von der die Sprachzuordnung übernommen werden kann. Ist dies nicht so, wird die Form nur als Textbestandteil aufgefasst und nicht strukturell behandelt. Ein letzter Spezialfall sind Formen in Kapitälchen im Text, mit denen Etyma bzw. historische Formen (zum Teil) markiert werden. Wenn diese keine explizite Sprachzuordnung enthalten, wird die Sprache des Lemmas verwendet. Überall sind in seltenen Fällen falsche Zuordnungen möglich, die Ausnahmen erfordern.

Der weitaus komplexere (und häufigere Fall) ist die Herleitung von Bedeutungen. Hier werden im REW prinzipiell Konventionen festgelegt:

> […] die romanische Bedeutung wird nur dann gegeben, wenn sie von der des Stichwortes abweicht. Besondere Bedeutungen in den Mundarten folgen dann, durch ; von den Grundformen getrennt. Bei den Ableitungen und Zusammensetzungen gilt eine Bedeutung für sämtliche ihr vorangehenden Formen.(REW, S. XI)

Es ist allerdings nicht immer klar, wie diese im Einzelfall auszulegen sind. Zum einen sind durch ein Semikolon nicht immer nur mundartliche Formen abgetrennt (bzw. es kommen mehrere Semikola vor), zum anderen wird keine Aussage über die

Bedeutungszuordnung zu solchen Formen ohne explizite Angabe getroffen. Sie scheinen zum Teil die Bedeutung das Lemmas zu übernehmen (vgl. Abb. 7.8), zum Teil aber auch die Bedeutung der nachfolgenden Form(en) (vgl. Abb. 7.9). Entsprechend werden Formen innerhalb der Hauptliste, die durch ein Semikolon getrennt sind grundsätzlich behandelt, als wären sie Teil der Zusammensetzungen bzw. Ableitungen, nur wenn sie keinerlei explizite Bedeutungsangaben enthalten, erben sie die des jeweiligen Lemmas. Diese Methodik ist nicht in allen Fällen korrekt, scheint aber einen Großteil der betrachteten Artikel korrekt abzubilden[5]. Auch zu zu weiteren Listen, die weder als Ableitungen noch als Zusammensetzungen markiert sind, wird strenggenommen nichts festgelegt. Sie werden somit wie diese behandelt.

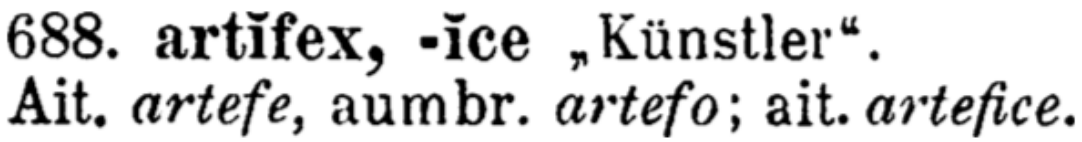

688. **artĭfex, -ĭce** „Künstler“.
Ait. *artefe*, aumbr. *artefo*; ait. *artefice*.

Abbildung 7.8: Semikolongetrennte Form in REW, S. 688

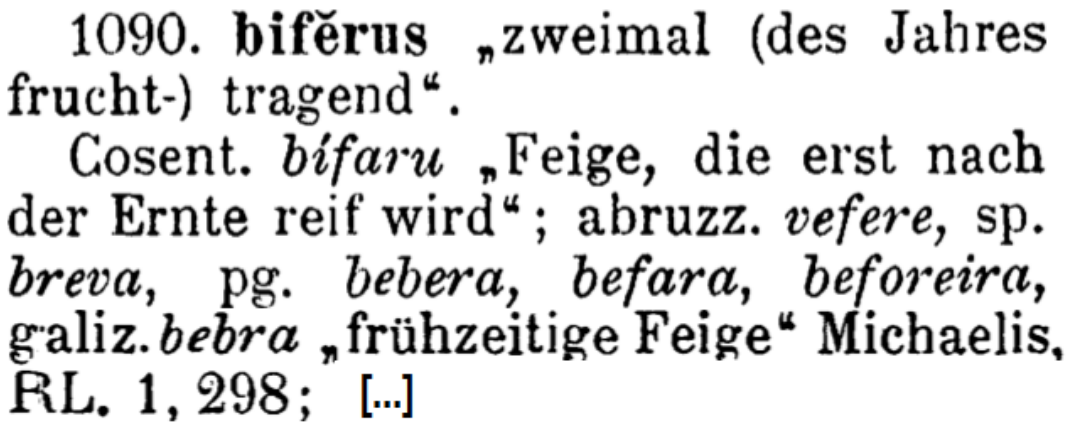

1090. **bifĕrus** „zweimal (des Jahres frucht-) tragend“.
Cosent. *bífaru* „Feige, die erst nach der Ernte reif wird“; abruzz. *vefere*, sp. *breva*, pg. *bebera*, *befara*, *beforeira*, galiz. *bebra* „frühzeitige Feige“ Michaelis, RL. 1, 298; [...]

Abbildung 7.9: Semikolongetrennte Form in einem Ausschnitt aus REW, S. 1090

Insgesamt führt dies zu folgendem Behandlungsschema für Bedeutungen (falls keine entsprechende Ausnahme existiert), dessen Regeln in absteigender Reihenfolge überprüft werden.

- Ist eine explizite Bedeutungsangabe vorhanden, wird diese verwendet.
- Falls diese nicht vorhanden ist und nach einer darauffolgenden (meist phonetischen) Variante ein Bedeutungsvermerk ist, wird dieser verwendet.

[5] Grundsätzlich werden alle inferierten Bedeutungen an der Oberfläche speziell markiert, da sich eine gewisse Unsicherheit nicht vermeiden lässt, vgl. Kap. 12.2

- Ist die Form Teil einer Entlehnungsliste wird die nächste Bedeutung der folgenden Formen innerhalb der Liste verwendet. Wenn keine Bedeutungsangabe vorhanden ist, wird die derjenigen Form verwendet, die Basis für die Entlehnung war.

- Ist die Form Teil der Kopfzeile (also ein Lemma) wird diejenige der ersten Form mit expliziter Bedeutungsangabe verwendet (hier gibt es hauptsächlich die beiden Fälle, dass dem ersten oder dem letzten Lemma eine Bedeutung zugeordnet ist).

- Ist die Form innerhalb der Hauptliste und vor dem ersten Semikolon, wird die Bedeutung des Lemmas übernommen.

- Ist die Form innerhalb der Hauptliste und nach dem ersten Semikolon, wird die erste Bedeutung der nachfolgenden Formen verwendet, falls keine vorhanden ist, die des Lemmas.

- Ansonsten wird die erste vorhandene Bedeutung nachfolgender Formen verwendet.

Trifft keine der Bedingungen zu, wird ein Sprachbeleg mit der „Pseudo-Bedeutung“ unbekannt erstellt. Dies ist häufig bei Ableitungslisten oder ähnlichem der Fall, die keine explizite Bedeutungsangabe enthalten. Zum Teil ist hier die Bedeutung des Lemmas anwendbar, zum Teil sind allerdings auch offensichtlich andere (nicht angegebene Bedeutungen) korrekt, weil beispielsweise die Formen nicht von der selben Wortart sind wie das Lemma. Das Aufstellen einer allgemeingültigen Regel ist somit hier nicht realistisch.

Konzeptuell werden unbekannte Bedeutungen nur dann über Ausnahmen ergänzt, wenn aus dem Artikelkontext ohne weitere Information klar erkennbar ist, welche Bedeutung eine gewisse Form hat, falls also nur die automatisierte Herleitung fehlgeschlagen ist. Ist dies nicht der Fall wird absichtlich keine Bedeutung vergeben, da der Datensatz den Anspruch hat den Informationsgehalt der Quelle wiederzugeben und keine zusätzlichen Informationen zu intergrieren[6]. Auch in einfachen Fällen, in denen die Bedeutungsangabe wohl weggelassen wurde, weil diese für Romanisten trivial ist, wird also keine Ausnahme eingeführt. Bei der sonstigen Behandlung der Bedeutungen wird keine Interpretation vorgenommen, nur in sehr offensichtlichen Fällen werden falsche Bedeutungszuordnungen (die aber formell den Regeln aus der Vorwort entsprechen) korrigiert. Ein Beispiel hierfür ist der Artikel REW, 2644b, in welchem den romanischen Formen die Bedeutung „tanzen“ über eine Ausnahme zugewiesen wird, obwohl sie laut den Regeln die Bedeutung des Lemmas übernehmen müssten:

[6]Dies kann nur sinnvollerweise im Abgleich mit anderen Quellen geschehen.

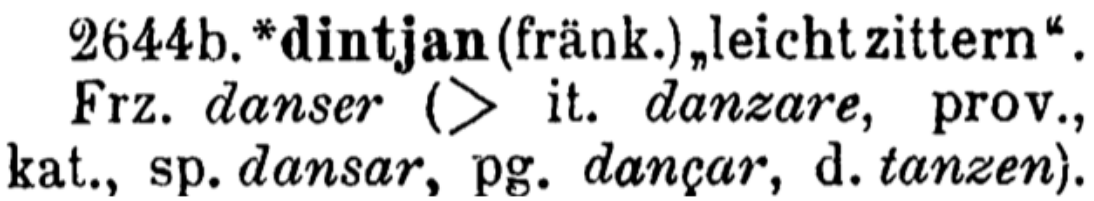
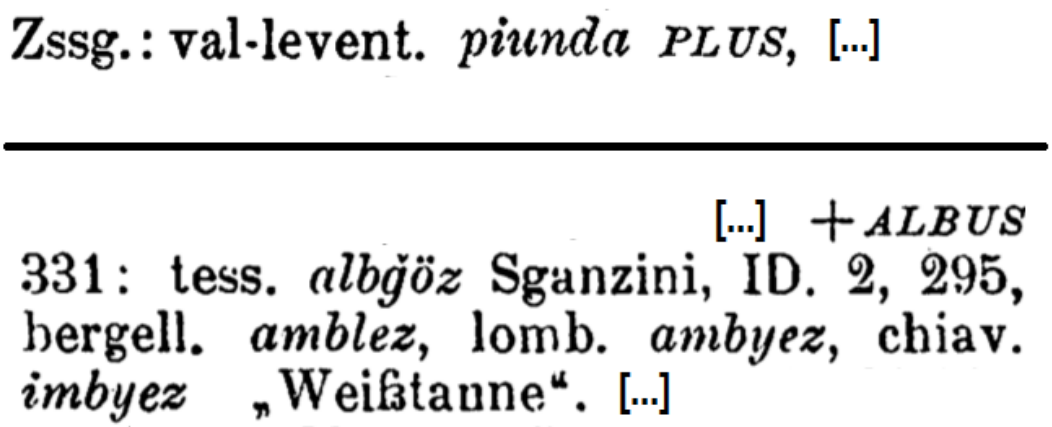

2644b. *dintjan (fränk.) „leicht zittern“. Frz. *danser* (> it. *danzare*, prov., kat., sp. *dansar*, pg. *dançar*, d. *tanzen*).

Abbildung 7.10: Beispiel, bei dem die Regeln aus dem Vorwort offensichtlich nicht angewendet werden sollen.

7.2.2 Information aus einleitenden und nachgelagerten Elementen

Der Quellentext enthält oftmals Informationen, die sich auf mehrere Formen beziehen oder einzelne Belege nachträglich als abweichend markieren. Zum Beispiel wird eine Kontamination, die sich auf mehrere Formen bezieht zu Beginn einer zusätzlichen Liste von Formen angeführt:


Zssg.: val-levent. *piunda* PLUS, [...]

[...] + ALBUS
331: tess. *albǧöz* Sganzini, ID. 2, 295, bergell. *amblez*, lomb. *ambyez*, chiav. *imbyez* „Weißtanne“. [...]

Abbildung 7.11: Angabe eines zweiten Etymons für eine einzelne Form (oben, REW, S. 53) und für alle Formen einer Liste (unten, REW, S. 25)

Insgesamt werden im REW solche Konstruktionen für folgende Art der Zusatzangaben regelmäßig verwendet:

- Kontaminationen
- Ortsnamen
- Grammatikalische Angaben
- Bedeutungen
- Markierung der Belege als dialektal

Dabei können strukturell ebenfalls verschiedene Grundtypen unterschieden werden:

- Ein einleitendes Element enthält eine Information für alle Elemente der Liste (z.B. für
- Ortsnamen „ON.:“)
- Eine Angabe nach einem Semikolon enthält eine Information für alle Elemente der Subliste (z.B. für eine Genuszuordnung „; Mask.“)
- Eine nachträgliche Angabe gilt für alle vorangehenden Formen (z.B. „; überall ...“)
- Eine nachträgliche Angabe enthält abweichende oder zusätzlichen Information für einzelne Formen (z.B. „asp. Mask.“)

Die ersten drei Varianten stellen dabei keine größere konzeptuelle Herausforderung dar, wobei die dritte aufgrund ihrer nachgelagerten Natur schwieriger in den Gesamtablauf bei der Verarbeitung des Strukturbaums einzubetten ist. Die komplexeste Aufgabe ist allerdings die Verarbeitung der letztgenannten Form der Zuordnung, da in diesem Fall keine explizite Form genannt wird und diese aus den vorherigen Formen hergeleitet werden muss. Im einfachsten Fall sind die in der Zusatzinformation angegeben Sprachabkürzungen in der vorangehenden Liste vorhanden. Somit können bestehende Bedeutungszuordnungen überschrieben werden bzw. zusätzliche Sprachbelege angelegt werden (falls „auch“ oder eine ähnliche Angabe vorhanden ist):

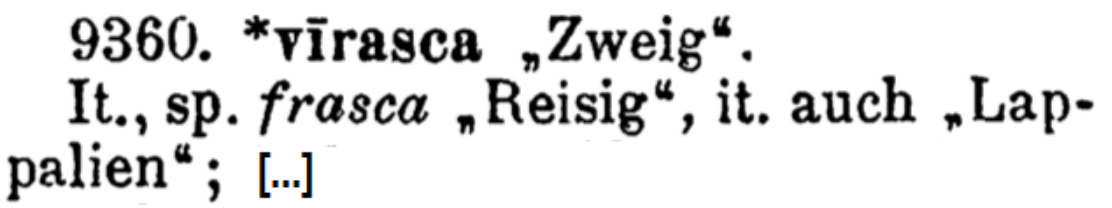

9360. *vīrasca „Zweig“.
It., sp. *frasca* „Reisig“, it. auch „Lappalien“; [...]

Abbildung 7.12: Eine einfache Form der nachgelagerten Bedeutungsangabe im REW

Auch leichte Variationen der Sprachangabe unter Verwendung der selben Sprachabkürzungen (z.B. „frz.“ vs. „südfrz.“ oder „kat.“ vs. „akat.“) können ohne Probleme erschlossen werden. Schwieriger wird es, wenn einzelne regionale Dialekte angeben werden, die so nicht in der Hauptliste vorkommen. Hierfür kann eine Hierarchie der verschiedenen Sprachabkürzungen verwendet werden. Mit deren Hilfe können die folgenden Regeln zur Herleitung der eigentlichen Form erstellt werden (für alle Regeln gilt dabei, dass bei mehreren gültigen Alternativen die letztgenannte verwendet wird, also diejenige die am nähesten an der Zusatzangabe ist):

- Besteht eine Form mit identischer Sprachabkürzung wird diese verwendet.

- Besteht eine Form einer in der Hierarchie übergeordneten oder untergeordneten Sprache, wird diese verwendet.
- Besteht eine Form, der die gleiche Sprache übergeordnet ist, wird diese verwendet.
- In anderen Fällen wird immer der direkte Vorgänger verwendet.

Eine gewisse Fehlerrate ist hier zu erwarten, da eine sehr hohe Zahl an Spezialfällen vorkommt, die nicht vollständig mit Regeln abgedeckt werden können. Auch ist in manchen Fällen die korrekte Zuordnung nicht mit Sicherheit ersichtlich. Im folgenden Beispiel wären grundsätzlich beide Formtexte möglich, wobei die zweite (die auch vom Algorithmus ausgewählt wird) in diesem Fall wohl die wahrscheinlichere ist:

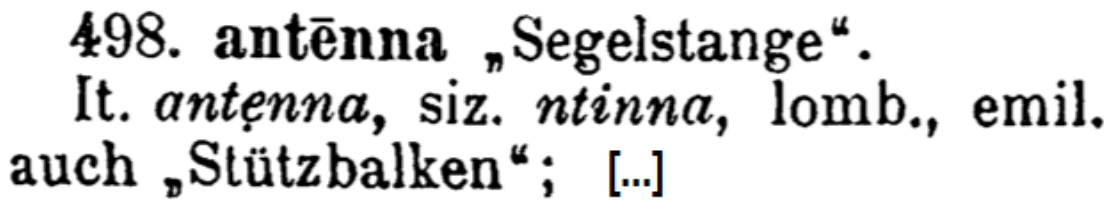

498. antēnna „Segelstange“.
It. *antẹnna*, siz. *ntinna*, lomb., emil. auch „Stützbalken“; [...]

Abbildung 7.13: Beispiel aus dem REW, in dem die Zuordnung einer Form unklar ist

7.2.3 Herleitung von Relationen aus der Position der Elemente im Artikel

Wird eine Form nicht direkt durch ihre Umgebung als Ergebnis einer Entlehnung oder Derivation ausgezeichnet (vgl. auch Kap. 6.5.1), wird sie über eine Nachfolger-Relation mit dem zugehörigen Etymon verbunden. Solange die Nummerierung von Kopfzeile und Hauptteil sich entspricht, werden jeweils die Formen in den Beleglisten des jeweiligen Abschnitt mit dem Lemma mit der gleichen Nummer verknüpft. Ist nur der Hauptteil nummeriert werden alle Formen mit dem jeweiligen Lemma verknüpft. Ist umgekehrt nur die Kopfzeile nummeriert oder im Hauptteil sind Nummern vorhanden, die in der Kopfzeile nicht existieren, werden jeweils alternative Relationen mit allen Lemmata erzeugt. Für jede Form gilt also, dass sie entweder vom ersten Lemma oder vom zweiten Lemma etc. stammt, eine genauere Spezifikation ist in diesem Fall nicht möglich.

Treten mehrere Varianten von Lemmata auf, die keine eigene Nummerierung aufweisen, werden auf Basis des REW zwei Fälle unterschieden:

- Unterschiedliche Flexionsformen
- Phonetische (oder sonstige) Varianten

Der erste Fall ist dabei sehr häufig durch die Angabe einer zusätzlichen (verkürzten) Akkusativform bei lateinischen Formen repräsentiert (z.B. „fragor, fragōre“ (REW,

S. 3474)). In diesem Fall werden die romanischen Formen nur mit dieser über eine Vorgänger-Relation verknüpft[7]. In anderen Fällen (z.B. „abiēteus, abēteus“ (REW, S. 25)) werden wiederum alternative Relationen erstelllt[8].

7.3 Ausnahmen im Importprozess

Bei den Ausnahmen im Verarbeitungsprozess werden zwei Hauptkategorien unterschieden, lokale und globale Ausnahmen. Lokale Ausnahmen sind (zumindest) an einen bestimmten Artikel gekoppelt und können beliebig genau weiterspezifiziert werden, bis hin zu einzelnen Positionen in den jeweiligen Beleglisten[9]. Grundsätzlich sollten soweit möglich globale Ausnahmen den lokalen vorgezogen werden, da diese unter Umständen (intern und extern) wiederverwendet werden können. So sollte beispielsweise die Aufspaltung einer Bedeutungsangabe mit einer globalen Ausnahme des Typs *meaning_splitting* durchgeführt werden, wobei sie theoretisch auch über eine des Typs *get_meaning* behandelt werden könnte. Lokale Ausnahmen sind somit nur in Fällen sinnvoll, in denen die abweichende Behandlung nur im Artikelkontext verwendet werden kann.

Die folgende Tabelle listet alle Typen von Prozessausnahmen und erklärt kurz deren Anwendungsbereich. Aufgrund der hohen Anzahl unterschiedlicher Ausnahmen, werden diese nicht im Detail besprochen, zum Teil werden diese aber in den folgenden Kapiteln wieder aufgegriffen. Einige wenige Typen von Ausnahmen beziehen sich dabei nicht auf die Verarbeitung der Wörterbuchartikel. Diese sind gesondert markiert.

[7] Die Nominativ und Akkusativform werden zusätzlich über die Relation *flexion_forms* miteinander verbunden.

[8] Auch Entlehnungen, die nicht klar einer Formvariante zugeordnet werden können, werden analog behandelt.

[9] Eine Ausnahme hier ist die sehr allgemeine Ausnahme *replace_text*, die sich nicht auf den Artikel an sich bezieht, sondern auf die Datenbankzeilen, die aus diesem erzeugt werden. Die Definition findet hier über den Tabellennamen und Werte für eine beliebige Anzahl von Spalten statt.

Bedeutungen	meaning_splitting	global	Steuert die (Nicht-)Aufspaltung einer Bedeutungsangabe (vgl. Kap. 7.1.2).
	get_meaning	lokal	Weist einer Form eine oder mehrere Bedeutungen zu.
	ignore_head_meaning	lokal	Eine (von mehreren) Bedeutungen des Lemmas wird nicht auf die romanischen Formen übertragen. Das ist sinnvoll bei direkten Übersetzungen oder übertragenen Bedeutungen, die zur Illustration der Herkunft des Lemmas dienen, aber für die anderen Formen nicht relevant sind.
	meaning_data	lokal/global	Ändert einzelne Felder einer Bedeutung. Eine globale Anwendung ist beispielsweise sinnvoll, um eine Bedeutung als „Bedeutungsklasse“ auszuweisen (vgl. Kap. 10.1), während eine lokale Verwendung verwendet werden kann, um beispielsweise eine zusätzliche Spezifikation zu einer Bedeutung hinzuzufügen (vgl. Kap. 10.2).
	meaning_abbreviation	lokal	Löst eine (kontextabhängige) Bedeutungsabkürzung auf.
Sprachen	spec_meaning_lang	lokal	Gibt die Sprache der Form an, deren Formtext für nachgelagerte Informationen verwendet werden soll (vgl. Kap. 7.2.2).
	lang_abbreviation	lokal	Löst eine (kontextabhängige) zusätzliche Abkürzung einer Sprache auf (z.B. „nord.- und südit.“).
	get_lang	lokal	Weist einer Form eine Sprache zu.
Formen	resolve_dash	global	Löst eine konkrete Formabkürzung mit Bindestrich auf (vgl. Kap. 7.1.3).
	resolve_dash_lang	global	Löst ein bestimmtes Muster von Formabkürzungen in einer bestimmten Sprache auf (vgl. Kap. 7.1.3).
	form_data	lokal	Ändert einzelne Felder der Daten einer sprachlichen Form.
	form_abbreviation	lokal	Löst eine (kontextabhängige) Formabkürzung auf.
Belege	record_data	lokal	Ändert einzelne Felder der Daten eines *Sprachbelegs.*
Etymon	get_etymon	lokal	Weist einer Form ein oder mehrere Lemmata als Etymon zu.
Literatur	bib_splitting	Bibliographie	Steuert die Varianten von bibliographischen Abkürzungen mit Klammern (vgl. Kap. 7.1.1).
	bib_abbreviation	lokal	Löst eine zusätzliche (kontextabhängige) Abkürzung einer bibliographischen Abkürzung auf
Allgemein	replace_text	lokal	Ersetzt allgemeine Felder von Zeilen, die in die Datenbank eingefügt werden. Dies ist die allgemeinste und mächtigste Form der Ausnahme, sie sollte allerdings nur benutzt werden, wenn keine der

Tabelle 7.1: Ausnahmen für verschiedene Verarbeitungsprozesse

8 Umgang mit Korrekturen

Dieses Kapitel beschäftigt sich mit verschiedenen Arten, wie Korrekturen auf den Eingangsdaten vorgenommen werden können. Dabei wird nicht nur die technische Sicht, sondern auch die Umsetzung von Korrekturmöglichkeiten über die Oberfläche eines Webportals betrachtet. Der erste Teil (Kap. 8.1) betrachtet wie einzelne Änderungen auf Basis der Textzeilen vorgenommen werden können und wie Nutzenden dies so unaufwendig und intuitiv wie möglich gestattet wird. Der zweite Abschnitt (Kap. 8.2) beschäftigt sich mit großflächigeren Korrekturen von systematischen Fehlern, die in den Eingangsdaten auftreten. Dies können einerseits strukturelle Probleme sein, die durch die Methoden in Kap. 4.3 nicht behoben wurden, aber auch inhaltliche Fehler, die auf Basis der reinen Textzeilen schwer zu beheben sind, aber unter Verwendung der bereits verarbeiteten Artikel zum Teil deutlich einfacher aufgefunden werden können. Zuletzt wird das Einpflegen von Korrekturen, die im Quellenmaterial enthalten sind, wie Anhänge und Verbesserungen von Fehlern in einzelnen Artikeln besprochen (Kap. 8.3).

8.1 Modellierung von Änderungen an der Textbasis

Alle Änderungen an den originalen Textzeilen werden als explizite Datensätze angelegt (vgl. Kap. 3.2). Um ein möglichst einheitliches Format zu verwenden, das gut relational darstellbar ist, werden folgende Felder verwendet:

id_line	ID der Zeile
number	Alle Änderungen auf der gleichen Zeile werden hier aufsteigend nummeriert, sodass sie in der Erstellungsreihenfolge auf diese angewendet werden können.
index	Position innerhalb der Zeile, an der die Änderung beginnt
value	Einzufügender Wert an der Stelle *index*
length	Anzahl Zeichen, die entfernt werden sollen
error_type	Gibt die Art des Fehlers an

Tabelle 8.1: Datenbankfelder für Änderungen an einzelnen Zeilen

Dieses Format erlaubt das Einfügen aller drei Arten von Änderungen in strukturell identische Spalten. Bei reinen Einfügungen ist der Wert *length* null, während bei reinen Löschungen das Feld *value* leer ist. Änderungen an der Formatierung, bei denen

beispielsweise zu Beginn und Ende eines Tokens ein öffnender und schließender Tag eingeführt werden, werden über zwei separate Änderungen dargestellt.

Das Feld *error_type* unterscheidet drei Werte:

- **ocr**: Fehler, die bei der automatischen Texterkennung entstanden sind.
- **source**: Eindeutige Fehler im Quellenmaterial. Hier werden Fehler behoben, die so bereits im ursprünglichen Werk enthalten sind.
- **auto**: Korrekturen, die durch das Post-Processing in Kap. 4.3 vorgenommen wurden

Beim Fehlertyp *source* werden nur sehr offensichtliche Probleme behoben, wie beispielsweise falsch geschriebene Personennamen, fehlende schließende oder öffnende Klammern oder Kommata statt Punkten. Trotzdem ist teilweise eine gewisse Interpretation nicht zu vermeiden. In folgender Textstelle ist beispielsweise das Entfernen der ersten schließenden Klammer und das der zweiten möglich, was beides einen grundsätzlich gültigen Artikeltext erzeugt, der aber eine unterschiedliche Semantik aufweist:

> frz. *malart* (> cosenz. *millardu*), benev. *mallarda* Bertoni, AR. 1, 415)(REW, S. 5392)

Um Nutzenden das einfache Anlegen der Korrekturen zu erlauben, wurde ein Wordpress-Plugin erstellt, welches dies auf einfache Weise ermöglicht. Der Grundgedanke dabei ist neben dem zu korrigierenden Scan eine Reihe von Textfeldern anzuordnen, die dessen Zeilen entsprechen und den Inhalt der Zeile aus der Datenbank enthalten. Alle Änderungen, die in den Textfeldern vorgenommen werden, werden direkt in die Datenbank übertragen und dort als „Korrektur-Datensätze“ angelegt. Dabei ist sowohl eine allgemeine Übersicht möglich, die die Korrektur aller Seiten des Quelle erlaubt (vgl. Abb. 8.1), aber auch die Verwendung einzelner Zeilen oder Ausschnitte aus den jeweiligen Scans (s. Kap. 8.2.2 und Kap. 12.5).

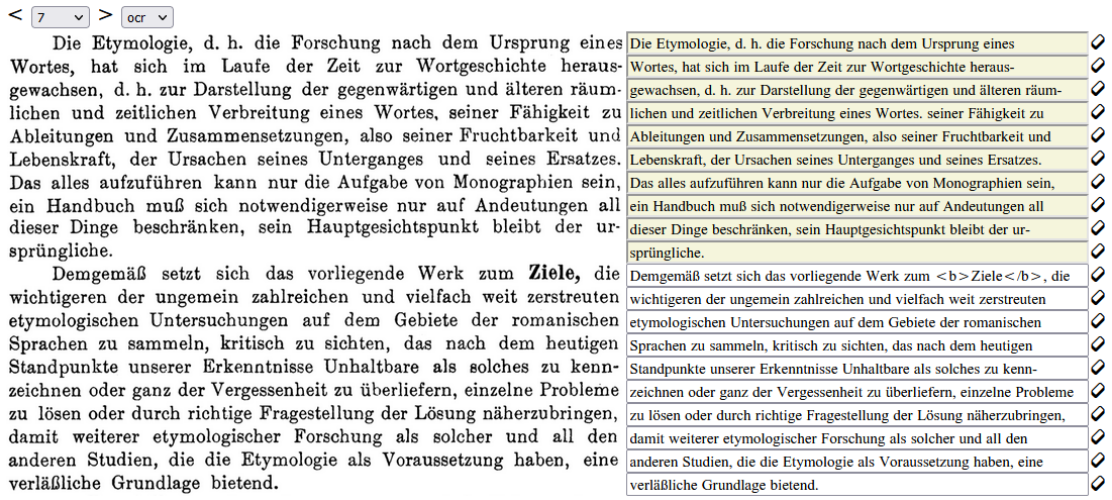


Abbildung 8.1: Ausschnitt der Korrekturoberfläche für Seite 7 aus dem REW

Ein neuer Absatz wird dabei über den Wechsel der Farbgebung markiert. Oben kann der jeweilige Fehlertyp ausgewählt werden (s. o.), das Radierersymbol rechts der Zeilen erlaubt die Entfernung aller Formatierungstags innerhalb der Zeile. An allen Stellen ist weiterhin eine Bildschirmtastatur mit allen Nicht-ASCII-Zeichen vorhanden (vgl. Abb. 8.2). Diese wird hier (und in den folgenden Abschnitten aus Platzgründen nicht mit angezeigt).

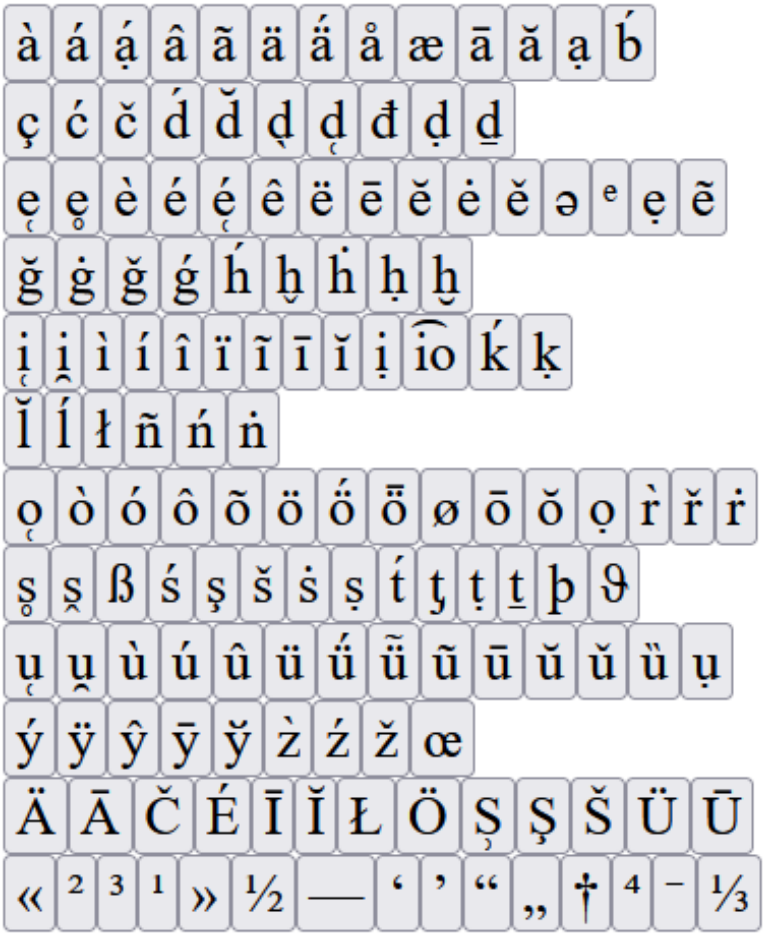


Abbildung 8.2: Bildschirmtastatur für die Eingabe von speziellen Zeichen

8.2 Massenkorrekturen

Korrekturen, die über einzelne Artikel oder Seiten hinausgehen, können über verschiedene Ansätze durchgeführt werden[1]. Eine erste einfache Möglichkeit ist die Überprüfung von bestimmten Konventionen im Quellenmaterial wie beispielsweise der alphabetischen Sortierung bestimmter Elemente. Dies erlaubt zwar nur das Auffinden von Fehlern einer sehr eingeschränkten Form, andererseits sind gerade diese oft besonders schwerwiegend, da beispielsweise eine möglichst fehlerfreie Erkennung der Abkürzungslisten wichtig für die weitere Verarbeitung ist. Auch auf Basis der Lemmata ist das Vorhandensein einer möglichst korrekten Nummerierung[2], auf der die

[1] Dieser Abschnitt befasst sich hier nur mit Werkzeugen für die Auffindung von Fehlern und eine möglichst einfache manuelle Korrektur. Eine potentiell automatisierte Verbesserung wie in Kap. 4.3 wird hier nicht behandelt.

[2] Für Korrekturen der Transkription der Lemmata vgl. auch Kap. 8.2.2

Darstellung des gesamten Wörterbuchs beruht, sicherlich als besonders wichtig einzustufen.

Die reine alphabetische Reihenfolge ist allerdings in den Abkürzungslisten nicht immer konsequent eingehalten (vgl. Abb. 8.3), sodass auch falsche Positive möglich sind.

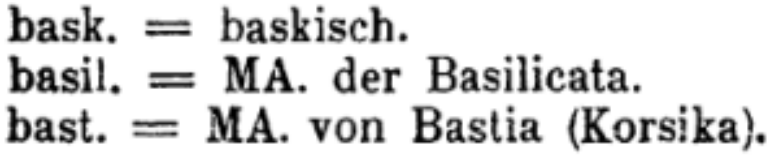


Abbildung 8.3: Beispiel für die Nicht-Einhaltung der alphabetischen Reihenfolge (REW, S. XXVIII)

Auf Basis der Lemmata wird (außer der Überprüfung der aufsteigenden Nummerierung) überprüft, ob das jeweils erste Lemma im Alphabet nach dem vorherigen kommt und zusätzlich, ob eventuelle weitere Lemmata mit dem gleichen Anfangsbuchstaben beginnen. Auch hier findet man nicht nur tatsächliche Fehler, sondern auch Ungenauigkeiten (oder auch absichtliche Abweichungen von der rein alphabetischen Darstellung) im Quellenmaterial. So folgt beispielsweise im REW das Lemma **acchordium** auf **achordare** oder das Lemma **dand** auf **danea**.

8.2.1 Allgemeine strukturelle Fehler

Bei manuellen Korrekturen bestimmter Wörterbuchartikel fallen oftmals systematische Fehler auf, die an verschiedenen Stellen auftreten. Dies können zwar der Ausgangspunkt für einer Verbesserung der Post-Processing-Schritte (vgl. Kap. 4.3) sein, falls das aber nicht möglich ist (oder auch für die Korrektur der bereits importierten Zeilen) ist es allerdings sinnvoll ein Werkzeug zu erstellen, das Fehler einer bestimmten Struktur artikelübergreifend findet und eine Anpassung ermöglicht. Insbesondere gilt dies für Fehler, die sich nur sehr schwer automatisiert beheben lassen. Dies gilt beispielsweise für das Vorkommen von zwei Punkten, die immer einen Fehler darstellen, aber durch verschiedene richtige Darstellungen ersetzt werden können (z.B. durch —, einen einfachen Punkt oder auch .,). Ein anderes Beispiel ist das Token *Wort*, das vom OCR-System regelmäßig als *Wert* erkannt wird. Obwohl das tatsächliche Vorkommen des Tokens *Wert* (zumindest im REW oder den meisten anderen linguistischen Ressourcen) deutlich seltener ist als das von *Wort*, ist es trotzdem wenig sinnvoll einfach alle Vorkommen von *Wert* durch *Wort* zu ersetzen und so andere Fehler einzuführen. Oftmals ist auch das Vorkommen ungewöhnlicher Notationen ein Hinweis auf einen anderen Fehler, so weisen beispielsweise kursive Seitenzahlen auf eine nicht erkannte literarische Abkürzungen hin, da Kursivität bei Ziffern im REW nur für Artikelreferenzen verwendet wird oder fett markierte Formen im Artikeltext (die grundsätzlich nicht vorkommen) auf gröbere Fehler in der Gesamtstruktur.

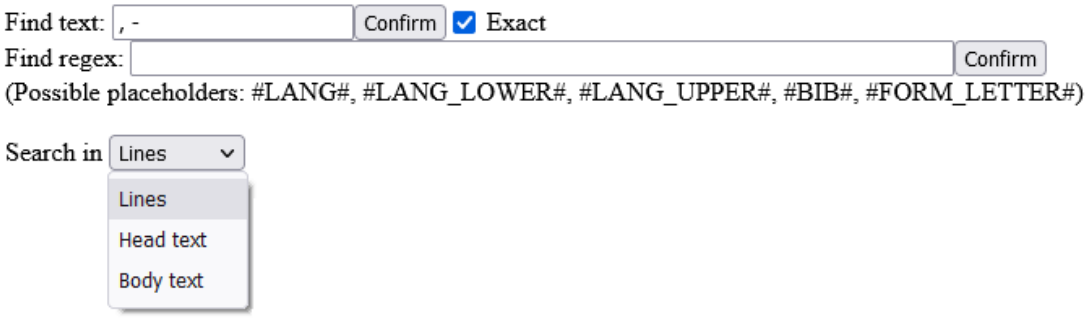


Abbildung 8.4: Einfache Eingabemaske zur Durchführung von Massenkorrekturen

Abb. 8.4 zeigt eine Eingabemaske, die dazu dienen kann solche Fehler in den Gesamtdaten zu finden. Für die Suche gibt es dabei drei Möglichkeiten:

- Suche mit Zeichenkette und Berücksichtigung von Diakritika und Groß- und Kleinschreibung
- Suche nach der exakten Zeichenkette
- Suche mit Hilfe eines regulären Ausdrucks

Im Fall eines regulären Ausdrucks sind weiterhin verschiedene Platzhalter möglich, mit denen man beispielsweise das Vorhandsein einer beliebigen Sprachabkürzung an einer bestimmten Stelle vorsehen kann. Die Suche findet dabei entweder auf Ebene der Zeilen oder anhand des zusammengeführten Fließtexts statt. Somit können einerseits Fehler gefunden werden, die sehr eng mit der Zeilenstruktur verknüpft sind (beispielsweise im Kontext der Worttrennung), aber auch solche die nur im Fließtext erkannt werden können (beispielsweise falsche Zeichen innerhalb von Wörtern, die im Originaltext getrennt auftreten). Die weitere Unterteilung nach Kopfzeile und Hauptteil dient dazu Fehler zu erkennen, die hauptsächlich oder ausschließlich im entsprechenden Abschnitt auftreten. Das einfachste Beispiel hierfür ist der Fettdruck, der ausschließlich in der Kopfzeile auftritt. Wenn <b>-Tags im Hauptteil vorkommen, wurde somit entweder ein Artikelanfang nicht als solcher erkannt, oder es wurde ein Wort fälschlicherweise fett markiert.

Die Ergebnisse der Suche sind dabei immer die jeweiligen Zeilen, in denen der gesuchte Bestandteil auftaucht. Im Fall der Suche im Fließtext ist somit eine Zuordnung der Zeilen zu den entsprechenden Positionen im Text wichtig (vgl. Kap. 5.1.2). Die folgende Abbildung zeigt die Resultate für das Vorkommen von Zeichenketten der Form Komma - Leerzeichen - Bindestrich, einer Kombination die ausschließlich durch Fehler bei der Texterkennung zustande kommt.

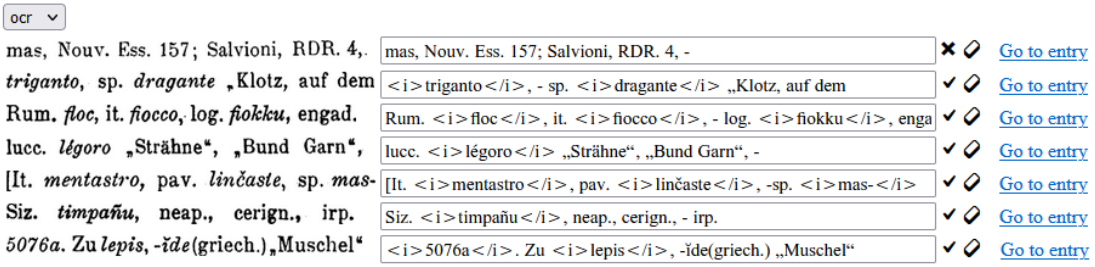


Abbildung 8.5: Ergebnisse der Suche nach dem Vorkommen des Suchstrings „, -“

Mit regulären Ausdrücken können aufwendigere Suchen durchgeführt werden. Die folgende Abbildung zeigt die Ergebnisse der Suche nach Tokens, die mit einem kleingeschriebenen „lüge“ beginnen, auf das nicht das Zeichen „n“ folgt:

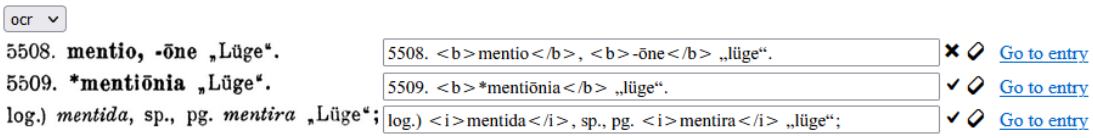


Abbildung 8.6: Ergebnisse der Suche nach dem Vorkommen des regulären Ausdrucks [„]lüge[^n]

Durch die Zeilenänderungen werden die betroffenen Artikel mit *needs_update* markiert, sodass sie im Anschluss gesammelt re-importiert werden können. Es sind aber zusätzlich die einzelnen Artikel verlinkt, sodass diese auch einzeln bearbeitet bzw. neu importiert werden können (vgl. Kap. 12.5).

8.2.2 Fehler bei sprachlichen Formen

Mit allen bisher vorgestellten Methoden ist eine Korrektur der Transkription von sprachlichen Formen nicht oder kaum möglich. Eine einfache Möglichkeit, um offensichtliche Fehler aufzufinden ist die Definition von Transkriptionssystemen bzw. Alphabeten, in denen Formen einer gewissen Sprachzugehörigkeit transkribiert werden. Besonders einfach ist dies für die romanischen Hauptsprachen sowie die lateinischen Lemmata, die zusätzlich zu den klassischen lateinischen Buchstaben nur Länge- und Kürzezeichen enthalten können. Diese Korrektur findet auf Basis der bereits erschlossenen Formen statt, denen bereits eine Sprache zugeordnet wurde (entweder weil diese explizit angegeben ist oder weil diese inferiert wurde, vgl. Kap. 7.2). Das Transkriptionssystem wird dabei über Abweichungen vom (modernen) lateinischen Alphabet definiert. Im Fall der italienischen Sprache kann dies beispielsweise über folgende Regeln durchgeführt werden:

id_rule	id_system	type	character
50	5	forbid	j
53	5	forbid	k
56	5	forbid	w
59	5	forbid	x
62	5	forbid	y
65	5	allow	à
68	5	allow	é
71	5	allow	rave;
74	5	allow	ì
77	5	allow	ò
80	5	allow	ù
83	5	allow	ó
86	5	allow	’

Tabelle 8.2: Regeln für die Beschreibung des italienischen Alphabets

Für Sprachen oder Dialekte, denen ein Transkriptionssystem zugewiesen wurde, können dann mit Hilfe einer einfach Eingabemaske abweichende Fälle gefunden werden. Um deren Korrektur möglich zu machen, wird die Zeile bzw. Zeilen angezeigt, in denen die Form vorkommt. Dies ist möglich, weil alle Sprachbelege[3] mit den Indexen im jeweiligen Originaltext versehen sind. Eine Zuweisung dieser ist möglich, weil bei der strukturellen Erfassung allen Entitäten jeweils ihre Position im Text zugeordnet ist (vgl. Kap. 5.2.5). Aus der Position im Fließtext können wiederum die Zeilen erkannt werden, die sich mit diesem Intervall überschneiden. Abb. 8.7 zeigt dies am Beispiel der französischen Sprache für eine Form bei der ein fehlerhaftes Diakritikum erkannt wurde:

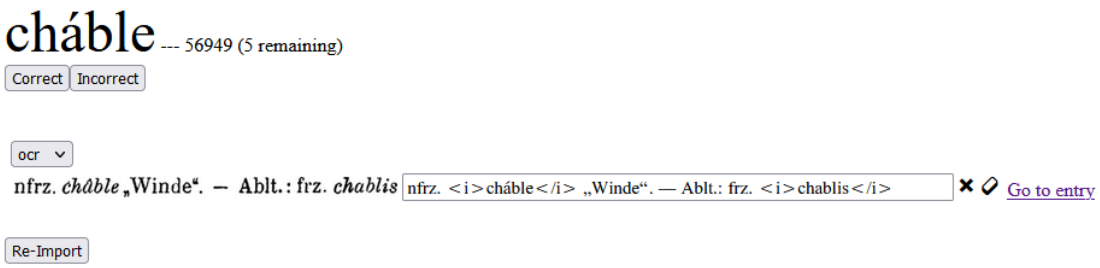


Abbildung 8.7: Beispiel für die Bedienelemente beim Auffinden von fehlerhaften Formen

In diesem Fall stellt diese Form eine von fünf zum aktuellen Zeitpunkt als ungültig transkribiert erkannten dar. Wird die Zeile (oder im Allgemeinen alle Zeilen, in denen diese Form vorkommt) geändert und ein Re-Import angestoßen, wird die nächste Form

[3] Eine Zuordnung der Form selbst wäre nicht sinnvoll, da die gleiche Form in mehreren Artikeln vorkommen kann.

angezeigt[4]. Es sind allerdings zum Teil auch abweichende Transkriptionen vorhanden, die aber absichtlich sind, z.B. bei zusätzlichen phonetischen Informationen für dialektale Varianten (vgl. Abb. 8.8). Solche Formen können explizit als gültig markiert werden und tauchen somit im weiteren nicht mehr auf.

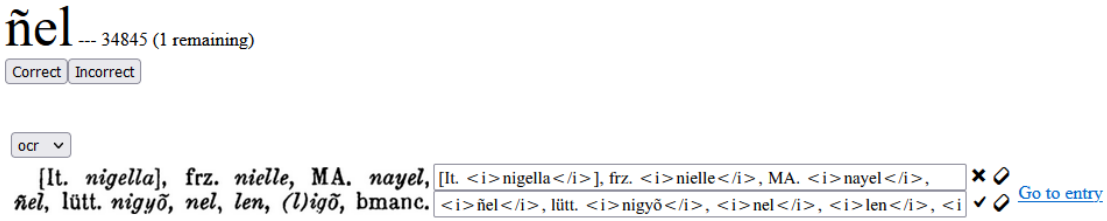


Abbildung 8.8: Beispiel für eine französische (Dialekt-)form, die nicht dem Transkriptionssystem entspricht und trotzdem gültig ist.

Zusätzlich zu falsch transkribierten Formen werden zum Teil auch falsch inferierte Sprachzuordnungen durch diese Methode aufgefunden. Ein Beispiel sind Formen in Kapitälchen, die ohne explizite Sprachzuordnung als lateinisch interpretiert werden. In gewissen Kontexten können sie allerdings auch beispielsweise griechisch sein, was zum Teil an dieser Stelle bei der Überprüfung der lateinischen Formen als Fehler erkannt wird.

8.3 Explizite Korrekturen in der Quelle

Die Inhalte dieses letzten Abschnitts weichen etwas von denen der vorherigen ab, beschäftigen sich aber ebenfalls mit Korrekturen an den Eingangsdaten. In diesem Fall geht es allerdings um solche, die explizit im Quellenmaterial vorhanden sind. Im Fall des REW können drei verschiedene Arten von solchen Angaben im Anhang unterschieden werden:

- Zusätzliche vollständige Artikel
- Längere Passagen, die zu einem bestehenden Artikel gehören
- Einzelkorrekturen bestimmter Zeilen des Artikeltexts

Der erste Fall lässt sich dabei am leichtesten behandeln, die entsprechenden Artikel werden wie gewohnt importiert. Die beiden anderen Fälle werden im folgenden besprochen.

[4] Die ungültige Form selbst ist (für die Versionierung) weiterhin in der Datenbank vorhanden, ist aber mit keiner aktuellen Artikelversion mehr verknüpft.

8.3.1 Zusatzinformation zu bestehenden Artikel

Die Nachträge zu den Artikeln im REW bestehen aus einer Lemma-Nummer und einem Textabschnitt, der zum entsprechenden Artikel hinzugefügt werden soll. Sie werden dementsprechend erfasst und in einer eigenen Datenbank-Tabelle *entry_supps* abgelegt. Beim (Re-)Import eines Artikels, zu dem es einen Nachtrag gibt, wird dieser an den eigentlichen Artikeltext angehängt und zusammen mit diesem mit Hilfe der formellen Grammatik verarbeitet. Im Importmodus eines Artikels (vgl. Kap. 12.5) werden zusätzlich zum Scan des eigentlichen Artikels der Ausschnitt des Scans angezeigt, der den Anhang enthält. Eine Bearbeitung ist an dieser Stelle allerdings nicht möglich, da ein Re-Import des Anhangs notwendig ist, bevor dessen korrigierte Version in den Artikel integriert werden kann. Somit ist nur eine spezielle Seite verlinkt, über die der Anhang alleine korrigiert und re-importiert werden kann.

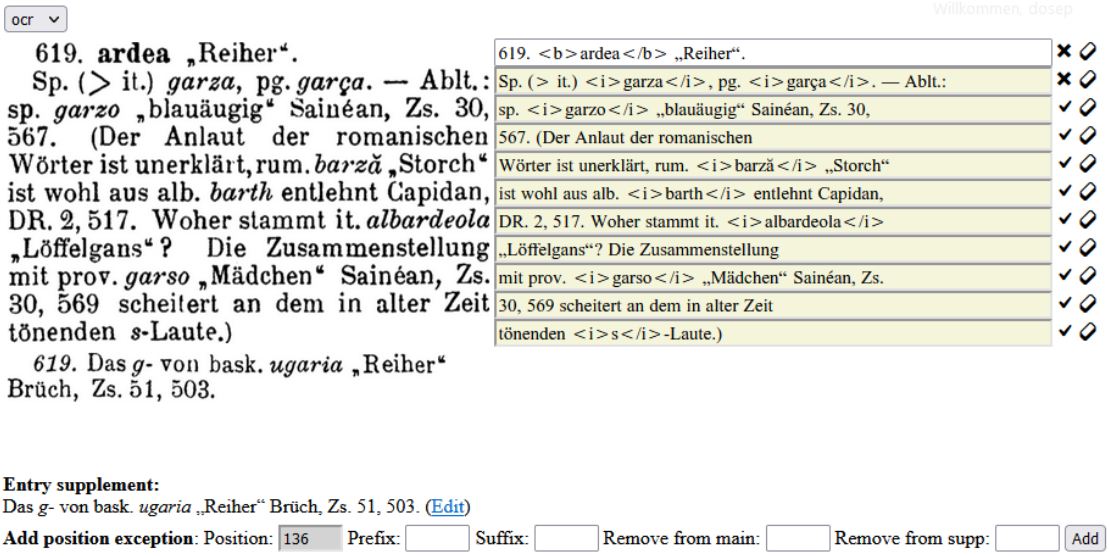


Abbildung 8.9: Artikel mit Anhang. Mit den Textfeldern unten kann eine Ausnahme für die genaue Position erstellt werden.

Standardmäßig wird der Anhang entweder mit dem Trennzeichen „—“ an den Artikeltext angehängt (falls der Anhang mit einem Großbuchstaben oder einer Klammer beginnt) oder mit einem Semikolon unter Entfernung des abschließenden Satzzeichens an den letzten Satz des Originaltexts angehängt. Wenn dieses Verhalten im Einzelfall nicht sinnvoll ist, kann die genaue Einfügeposition über eine spezielle Ausnahme festgestellt werden. Die Position kann dabei über eine Markierung des Beginns im Artikeltext festgelegt werden. Zusätzlich können vor oder nach dem Anhang Zeichen eingefügt werden (*prefix* und *suffix*) oder einzelne Zeichen am Ende des Originaltexts oder zu Beginn des Anhangs entfernt werden.

8.3.2 Einzelne Fehlerkorrekturen

Einzelne Verbesserungen werden im REW durch eine Lemma-Nummer, eine Zeilennummer[5], optional die Nummer eines Blocks und den eigentlichen Inhalt definiert. Analog zur Behandlung der Anhänge werden die Verbesserungen in einem ersten Schritt in eine eigene Tabelle *entry_corrs* importiert und können anschließend in den entsprechenden Artikel integriert werden. Während manche der Änderungen sehr systematisch sind („str. ...“ für eine Entfernung und „... statt ...“ für eine Ersetzung) und dementsprechend einfach anzuwenden sind, enthalten die meisten nur den korrigierten Text und die genaue Verwendung muss erst bestimmt werden. Automatisiert werden dabei zwei Fälle behandelt:

- Reine Einfügungen
- Ersetzungen, deren Abstand vom Originaltext nicht zu groß ist

Beide arbeiten mit einer Tokenisierung der Zeile(n) und der Korrekturzeichenkette, die grundsätzlich der aus Kap. 4.3 entspricht, aber auf die dort verwendete teilweise Abstrahierung der Tokens verzichtet. Die Nutzung einer tokenisierten Fassung ist sinnvoll, da die Korrekturangaben ausschließlich aus ganzen Wörtern bestehen. Zuerst wird überprüft, ob eine Einfügung vorliegt. Dazu wird jede Untermenge von Tokens der Korrekturangabe, bei der aus der Mitte eines oder mehrere Tokens entfernt werden, betrachtet und getestet, ob diese in der Originalzeile enthalten ist. Wird eine solche Stelle gefunden, kann diese durch die gesamte Korrektur ersetzt werden. Dies soll hier am Beispiel einer Korrektur von REW, S. 65 verdeutlicht werden. Die Korrekturangabe ist dabei „Baist, RF. 8, 512“. Dies wird tokensiert zu

```
["Baist",","," ","RF."," ","8",","," ","512"]
```

Im weiteren werden alle Zeichenketten gesucht, die entstehen, wenn man eine Reihe von Tokens (mit Ausnahme des ersten und letzten) entfernt und die restlichen neu zusammensetzt:

```
["Baist 8, 512", "Baist8, 512", "Baist, 512", "Baist 512", "Baist512", ...]
```

Die entsprechende Zeile enthält in diesem Fall die Zeichenkette „Baist 512“, somit wird diese durch die Korrektur ersetzt.

Kann eine solche Einfügung nicht erkannt werden, werden im weiteren Verlauf allgemeinere Ersetzungen überprüft. Dabei werden aus der Textzeile alle

[5] Die Zeilennummer gibt an, in welcher Zeile die Korrektur beginnt. Sie kann sich allerdings über mehrere Zeilen erstrecken.

Teilzeichenketten überprüft, die aus der gleichen Anzahl von Tokens bestehen, wie die der Korrektur. Für jeden Bestandteil wird der Abstand zur Korrekurangabe (unter Verwendung der Levenshtein-Distanz, vgl. Kap. 4.3.3) berechnet. Für diesen Vergleich werden vorab die Leerzeichen verdoppelt, um die Kosten für eine Veränderung der Tokengrenzen zu erhöhen. Falls es genau eine Zeichenkette gibt die einen minimalen Abstand zur Korrekturzeichenkette hat und dieser nicht zu groß ist[6], kann die Ersetzung an dieser Stelle vorgenommen werden. Falls dies nicht der Fall ist, wird das gleiche Prozedere mit einer jeweils eins kleineren oder eins größeren Tokenzahl wiederholt. Ein einfaches Beispiel ist die Korrektur `lim. dezousiná`, die auf Zeile 7 in REW, S. 51 angewandt wird. Die tokenisierte Fassung besteht aus drei Tokens:

```
["lim.", " ", "<i>dezousiná</i>"]
```

Es werden somit alle Teilabschnitte der entsprechenden Zeile `„unbebautes Land", lim. dezuosiná`, die ebenfalls aus drei Tokens bestehen untersucht und deren Abstand berechnet:

Zeichenkette	**Abstand zur Korrektur**
„unbebautes␣	21
unbebautes␣Land	21
␣Land“	20
Land“,	22
“,	21
,␣lim.	20
␣lim␣	19
lim.␣dezuosiná	2
␣dezuosiná␣	8

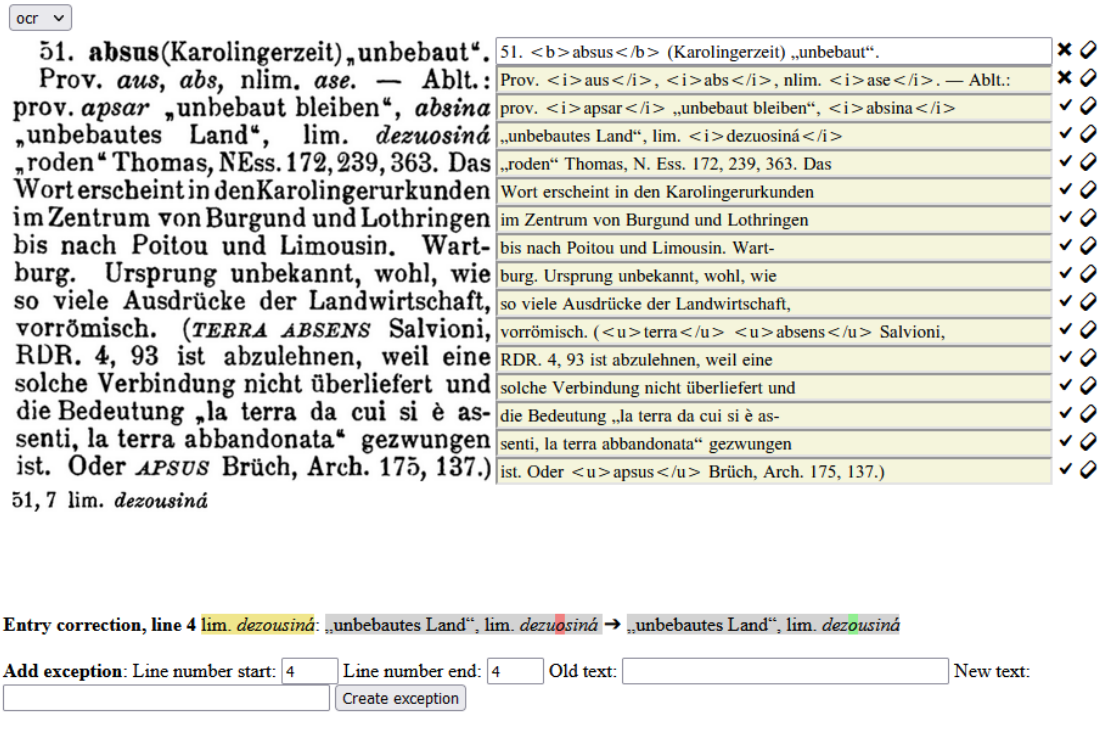
Der Bestandteil mit dem geringsten Abstand[7] wird somit durch die Korrektur ersetzt.

Diese Vorgehen kann den Großteil der Verbesserungen behandeln, in sehr komplexen Fällen wird allerdings kein (oder sehr selten) ein falsches Ergebnis erzeugt. Bei einigen Verbesserungen ist außerdem die angegebene Zeilennummer nicht korrekt, so dass die Verbesserung ebenfalls nicht eingepflegt werden kann. Somit kann wiederum eine Ausnahme definiert werden, die entweder nur die korrekte(n) Zeilennummer(n) enthält oder falls nötig die explizite Angabe der Korrektur (also welche Zeichenkette durch welche neue ersetzt werden soll).

[6] Als maximale Distanz wurde hier der Wert acht bzw. die Hälfte der Länge der Korrekturzeichenkette verwendet.

[7] Der Abstand ist in diesem Fall zwei, weil an dieser Stelle die Standard-Implementierung der Levenshtein-Distanz in der Programmiersprache PHP verwendet wurde, die den Vergleich auf Ebene von Bytes durchführt, d.h. Zeichen mit Diakritika werden als zwei Zeichen behandelt. Für die Ergebnisse des Algorithmus spielt das im Normalfall keine größere Rolle, wenn die maximale Distanz entsprechend etwas höher definiert wird.

[caption id=„attachment_87158“ align=„alignnone“



width=„915“]

9 Zusammenlegung sprachlicher Formen

EIn häufiges Problem bei digitalen Projekten in der Linguistik ist der Umgang mit Homonymie, speziell die Erkennung und entsprechende Markierung von Homographen, d.h. von unterschiedlichen sprachlichen Formen, deren textuelle Repräsentation identisch ist. Aus technischer Sicht ist dabei ein eventueller Unterschied in der jeweiligen Aussprache, solange dieser nicht explizit durch beispielsweise Länge- und Kürzezeichen kodiert wird, nicht relevant bzw. kann nicht automatisiert erkannt und damit nicht genutzt werden. Bei der Verarbeitung eines Wörterbuchtexts entstehen dabei zwei gegensätzliche Probleme, einerseits die Trennung von unterschiedlichen homonymen Formen, andererseits die Bündelung von gleichen Formen, die sich nicht oder nur in der Art der Umschrift unterscheiden. Beide Kriterien sind dabei entscheidend; fände keinerlei Bündelung statt, wäre jegliche kontextübergreifende Abfrage oder Informationsaggregation unmöglich, eine Zusammenfassung aller textuell identischen Formen würde hingegen grundlegende sprachwissenschaftliche Prinzipien verletzen und dem Quellenmaterial nicht gerecht werden.

Im Kontext der Lexikographie ist eine Unterscheidung auf Lemmataebene verhältnismäßig einfach, da diese bereits vom Autor anhand der Lemmatisierung vorgenommen wurde. Schwieriger ist die Zuordnung allerdings, wenn bestimmte Etyma zusätzlich an anderer Stelle im Text genannt werden. Ein Beispiel aus dem REW hierfür ist die Zusatzinformation „Hat PILA völlig verdrängt.“ (REW, S. 5693), wobei die lateinische Form sich grundsätzlich auf die Lemmata **pīla** (6496), **pīla** (6497) oder **pĭla** (6498) beziehen könnte und nur aus dem Kontext erschlossen werden kann, welche davon tatsächlich referenziert wird. Auf Basis der Einzelformen ist dieses Problem in den meisten Fällen vorhanden. Gesucht ist also ein Mechanismus, der entscheidet, ob zwei orthographisch identische Formen Homonyme sind oder nicht. Da das Problem vor allem aus technischer Perspektive unter dem Gesichtspunkt der Automatisierung betrachtet wird, wird hier nicht näher auf die Unterscheidung zwischen Homonymie und Polysemie eingegangen, die durchaus umstritten ist (vgl. Nikula 2013, S. 228–230), sondern eine einfache Form der etymologischen Definition verwendet, die ein Homonym folgendermaßen festlegt:

> Wort, das mit einem andern gleich lautet, den gleichen Wortkörper hat (aber in der Bedeutung (und Herkunft) verschieden ist.(DWDS: Homonym)

Für die technische Bestimmung der Identität sprachlichen Formen mit identischer Sprachzuordnung, Orthographie und grammatikalischer Spezifikation werden hieraus folgende Kriterien erstellt:

- Formen mit (herleitbarer) Etymologie sind identisch, wenn sie mindestens ein gemeinsames Etymon haben, sonst nicht.
- Formen mit (herleitbarer) Bedeutung aber ohne Etymologie sind identisch, wenn sie sich in mindestens einer Bedeutung überschneiden, sonst nicht.
- Formen, die im gleichen Artikel vorkommen, sind identisch, wenn dieser nicht bereits Homonyme nach den obigen Kriterien enthält.
- Falls ein Etymon in einer Artikelreferenz enthalten ist und der referenzierte Artikel ein orthographisch gleiches Lemma hat, sind beide identisch.
- Andere Formen sind identisch, wenn keine orthographisch gleiche Form ein Homonym hat.

Das letzte Kriterium ist quellenübergreifend und fasst vor allem Formen, die ohne Bedeutungsangabe im Text verschiedener Artikel vorkommen, untereinander und mit einer eventuell existenten näher bestimmten Form zusammen. Die Intuition ist dabei, dass nicht näher bestimmte Formen identisch mit einer genauer spezifizierten Form sind, wenn es davon im gesamten Wörterbuch nur genau eine gibt. So ist offensichtlich das französische *billard*, das in REW, S. 1101 ohne Bedeutung erwähnt wird identisch mit dem in REW, S. 1104 gelisteten. Genauso sind die deutschen Formen *lappen*, die in REW, 4803a und REW, S. 4905 ohne explizite Bedeutung erwähnt werden, erkennbar identisch. Das Kriterium ist allerdings insofern problematisch, als dass es nicht zum Zeitpunkt eines einzelnen Imports entschieden werden kann, sondern erst wenn das vollständige Material vorliegt. Somit können zum Importzeitpunkt nur die ersten Kriterien überprüft werden. Für alle weiteren Fällen werden unterschiedliche (nummerierte) Formen angelegt. Die Anwendung der ersten vier Kriterien zum Zeitpunkt des Imports ist möglich, da sie zwei entscheidende Eigenschaften aufweisen:

- **Symmetrie***:* Sie funktionieren somit „in beide Richtungen“, d.h. die Importreihenfolge der Artikel spielt keine Rolle für die Identität der Formen.
- **Konstanz:** Sie ändern sich nicht, wenn weitere Artikel eingefügt werden (oder bestehende Objekte angereichert werden[1]). Somit kann ausgeschlossen werden, dass eine neue Artikelversion einzig aufgrund einer „externen“ Änderung erstellt werden muss, obwohl sich die eigentlichen Artikeldaten nicht geändert haben.

Die Nummerierung der Formen ist somit fest (solange sich die entsprechenden Artikel nicht ändern). Eine weitere Zusammenlegung der Formen anhand des letzten Kriteriums findet erst zu einem späteren Zeitpunkt statt[2]. Dafür gibt es grundsätzlich

[1] An dieser Stelle werden deshalb nur identische Bedeutungsangaben für das zweite Kriterium verwendet, da die Verknüpfung von synonymen Bedeutungen ansonsten nachträglich die Nummerierung ändern könnte.

[2] Grundsätzlich könnte man auch beim Import alle auftretende Formen unterscheiden und erst später zusammenführen. Dies wäre allerdings wenig praktikabel, da selbst mehrere Vorkommen einer Form im gleichen Artikel als einzelne Einträge angelegt würden.

die beiden Möglichkeiten bestehende Formen zu einer zu kombinieren oder sie zu behalten und durch eine zusätzliche Gruppierung zu bündeln. In diesem Fall wird die zweite Variante verwendet, da jede sprachliche Form eine persistente ID erhält, die sich nachträglich nicht mehr ändern soll. Alle URLs, die auf dieser basieren bleiben so dauerhaft gültig (vgl. hierzu Kap. 12.3).

Eine Zusammenlegung kann stattfinden, wenn alle Artikel initial importiert wurden (und muss bei Korrekturen für betroffene Formen neu berechnet werden). An dieser Stelle werden auch Formen zusammengeführt, deren grammatikalische Angaben unterschiedlich sind, sich aber nicht widersprechen, falls es nur einen Kandidaten gibt. Falls es also beispielsweise nur eine Form mit der Genuszuweisung *maskulin* und eine ohne Genuszuweisung gibt, werden sie zusammengefasst. Falls es eine weitere mit femininer Zuweisung gibt, ist die Zusammenführung nicht möglich. Weiterhin werden hier zusätzlich Formen als identisch markiert, die eine schwächere Variante des zweiten Kriteriums erfüllen, das keine identischen Bedeutungen verlangt, sondern auch synonyme erlaubt (zum Auffinden von solchen siehe Kap. 10.2).

Eine letzte Zusammenführung findet auf Basis von orthographisch identischen sprachlichen Formen unterschiedlicher Dialekte der gleichen übergeordneten Sprache statt, wenn sie vom selben Etymon abstammen. Dies findet auch bei unterschiedlichen Bedeutungen statt, d.h. ein solches Vorkommen wird als Polysemie betrachtet. Auch dieser Schritt muss in der zweiten Phase stattfinden, da die Sprachhierarchie nicht konstant ist und somit eine nachträgliche Änderung der Nummerierung möglich wäre. Ein Beispiel ist der folgende Ausschnitt, in dem alle Formen in den verschiedenen italienischen Dialekten zusammengefasst werden.

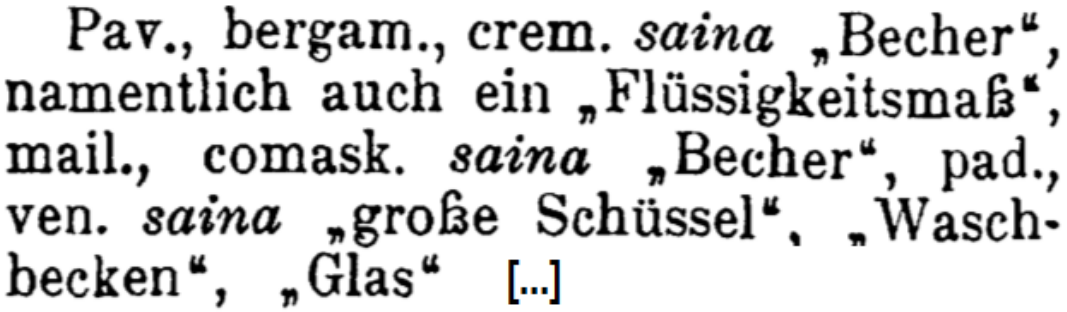

Pav., bergam., crem. *saina* „Becher“, namentlich auch ein „Flüssigkeitsmaß“, mail., comask. *saina* „Becher“, pad., ven. *saina* „große Schüssel“, „Waschbecken“, „Glas“ [...]

Abbildung 9.1: Ausschnitt aus REW, S. 2433

Auf den in Kap. 12.3 näher beschriebenen Detailseiten einer Form wird sowohl die Zusammlegung, als auch die Trennung im Bezug auf andere Formen dargestellt. Für die Form *saina* aus dem obigen Beispiel sieht dies so aus:

saina **(pav., bergam., crem., mail., comask., pad., ven.)**

Occurances

- Entry 2433 (derivation): pav., bergam., crem., mail., comask. „Becher“
- Entry 2433 (derivation): pav., bergam., crem. <ein Flüssigkeitsmaß>
- Entry 2433 (derivation): pad., ven. „große Schüssel“
- Entry 2433 (derivation): pad., ven. „Waschbecken“
- Entry 2433 (derivation): pad., ven. „Glas“ (Gefäß)

Homonyms

- bergam. *saina* „Schleppnetz“

Abbildung 9.2: Auschnitt einer Detailseite (Link)

Im folgenden wird die Durchführung des zweistufigen Verfahrens anhand von zwei Beispielen illustriert. Eines behandelt drei Vorkommen der lateinischen Form *gabata*. Im ersten Schritt können das erste und dritte Vorkommen als identisch erkannt werden, da eine Referenz auf die entsprechende Lemmanummer vorhanden ist. Das zweite Vorkommen kann allerdings nicht sicher zugeordnet werden, da es theoretisch möglich wäre, dass in einem späteren Artikel eine weitere homonyme Form vorkommt. Da dies nicht der Fall ist, werden die beiden Einträge im zweiten Schritt der gleichen Gruppe zugeordnet.

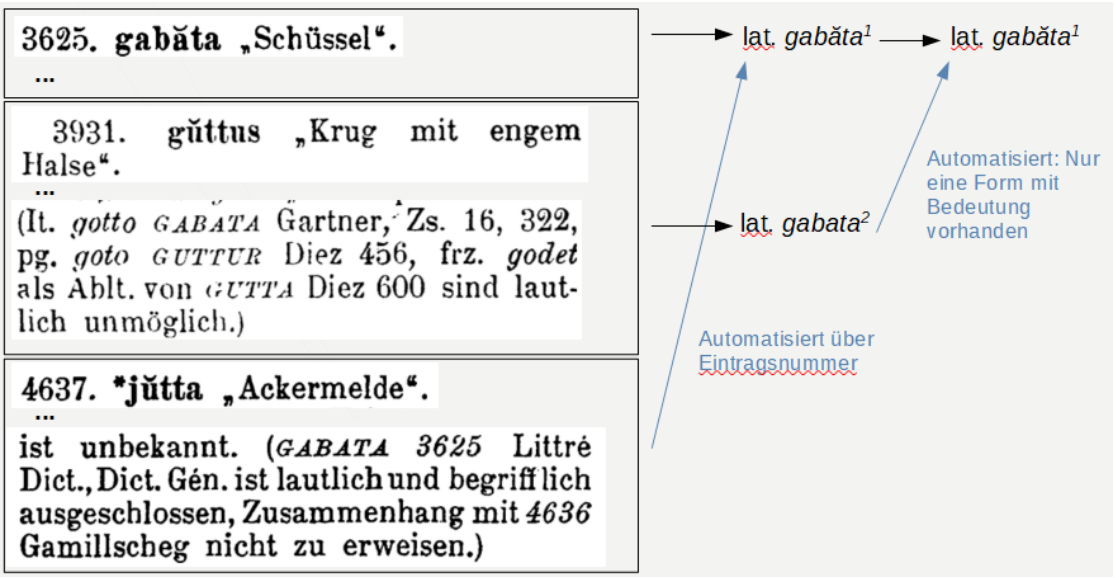
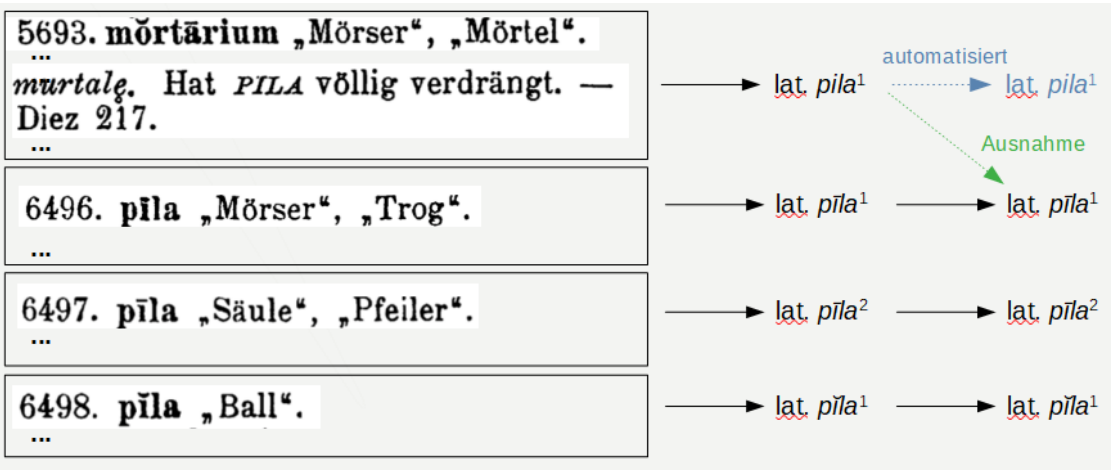


Abbildung 9.3: Beispiel für die Zusammenführung von Formen in zwei Phasen

Im zweiten Beispiel ist diese Zuordnung nicht (automatisiert) möglich. In diesem Fall handelt sich wieder um die bereits zu Beginn erwähnten drei Lemmata *pīla* bzw. *pĭla*,

die für die Transkription ohne Diakritikum in Frage kommen[3]. Automatisiert bleibt es hier bei vier verschiedenen Formen. Eine Zuordnung zum korrekten Lemma kann somit höchstens über eine Ausnahme durchgeführt werden.

[caption id=„attachment_87805“ align=„alignnone“



width=„866“]

[3] Auch wenn die Form ein Längenzeichen enthielte, könnte sich nicht zugeordnet werden, da hierfür zwei Homonyme existieren.

10 Bedeutungen und Konzepte

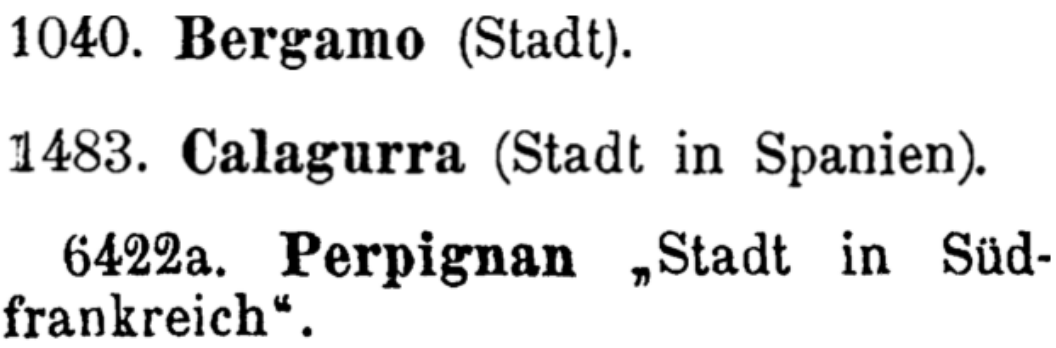
In diesem Kapitel wird das Themengebiet der Bedeutungsangaben näher betrachtet. Zuerst wird die Art analysiert, in der traditionelle Werke diese formulieren und welche unterschiedlichen Formen von Bedeutungsangaben vorkommen (Kap. 10.1). Im weiteren steht eher die technische Nutzbarkeit und Normierung der einzelsprachlichen Bedeutungsangaben im Vordergrund, wobei die Verknüpfung mit der externen Wissensdatenbank *Wikidata* genauer betrachtet wird (Kap. 10.2).

10.1 Bedeutungen und Bedeutungsklassen

Die Bedeutungsangaben in traditionellen Wörterbüchern sind in vielen Fällen sehr unsystematisch. Dies bezieht sich nicht nur auf die Art der Notation (vgl. hierzu Kap. 7.1.2), sondern vor allem auch auf verschiedene Varianten der Bedeutungsangaben, die nicht (oder nicht konsequent) unterschieden werden. Auf abstrakter Ebenen kann zwischen einer Übersetzung und der Beschreibung des zugrundeliegenden Konzepts unterschieden werden. Im ersten Fall wird eine direkte Wiedergabe der Bezeichnung in einer anderen Sprache angegeben, im zweiten eine abstrakte Umschreibung für eine bestimmte Sache, also eine technische (nicht unbedingt idiomatische) Angabe der Bedeutung. Besonders augenscheinlich wird diese Unterscheidung oftmals anhand der Bedeutungsangabe bei Redewendungen. So werden beispielsweise in REW, S. 172 einerseits die Beschreibung „Ausruf der Matrosen, wenn sie einige Tage kein Land gesehen haben“, andererseits aber Übersetzungen wie „helf Gott“ oder „so wahr mir Gott helfe“ verwendet. Diese Unterscheidung ist offensichtlich abhängig davon, ob eine Konstruktion in der Erklärungssprache existiert und geläufig genug ist, um keiner weitere Erklärung zu bedürfen. Weniger verständlich ist die unterschiedliche Verwendung zum Teil bei Eigennamen. So wechseln sich hier „Übersetzungen“ wie „Bagdad“ für das Lemma *Bagdad* (REW, S. 881) und Beschreibungen wie „Stadt in Frankreich“ für das Lemma *Limoges* (REW, S. 5053) ab.

Im ganz großen Teil der Fälle ist der Übergang zwischen beidem allerdings fließend, da besonders für allgemein bekannte Konzepte eine Bedeutungsangabe in einem Wort sowohl als Übersetzung, als auch als Konzeptbeschreibung aufgefasst werden kann. Bei sehr spezialisierten Konzepten geht die Tendenz zur Konzeptbeschreibung, allein schon, weil es oftmals keine passende Übersetzung in der jeweiligen Erklärungssprache gibt. Somit ist die Unterscheidung in der Praxis oftmals akademisch und hat auch für technische Anwendungen keine größere Bedeutung (vgl. auch Kap. 10.2.2). Gerade

anhand der oben genannten Toponyme tritt allerdings ein weiterer, deutlich relevanterer Unterschied zu Tage. Eine Bedeutungsangabe wie „Stadt in Frankreich“ wird so interpretiert, dass die sprachliche Form eine spezifische Stadt in Frankreich bezeichnet und nicht das Konzept STADT IN FRANKREICH, für das Bezeichnungen grundsätzlich durchaus denkbar wären. Im REW ist eine Unterscheidung beider Varianten gerade bei den Toponymen durchaus vorhanden, indem anstatt der Anführungszeichen runde Klammern benutzt werden, die Verwendung ist aber höchst inkonsequent:


1040. **Bergamo** (Stadt).

1483. **Calagurra** (Stadt in Spanien).

6422a. **Perpignan** „Stadt in Südfrankreich“.

Abbildung 10.1: Verschiedene Notationen für die Markierung von Städtenamen im REW

Die Verständlichkeit bleibt dabei im Normalfall für menschliche Lesende durchaus erhalten, da beispielsweise das Wort „Stadt“ alleine je nach Fall konsequent in Klammern oder Anführungszeichen gesetzt wird (und in diesem Fall auch ansonsten eine Erkennung intuitiv möglich wäre), für die technische Verarbeitung ist dies allerdings zum Teil ein Problem. Vielfach findet die Unterscheidung auf sprachlicher und nicht auf struktureller Ebene statt, so wird beispielsweise „Fisch“ für das Konzept FISCH und „ein Fisch“ für die Beschreibung einer Art bzw. Gattung von Fisch verwendet, die nicht näher spezifiziert wird. Abstrakt kann hierbei der Fall einer direkten Konzeptbeschreibung und der Beschreibung eines übergeordneten Konzepts unterschieden werden. Zweitere wird hier als *Bedeutungsklasse* bezeichnet, da sie nicht ein spezifisches Konzept beschreibt, sondern eine Klasse von Konzepten, von der eine nicht genauer spezifizierte Instanz der sprachlichen Form zugeordnet ist[1].

Bei der Verarbeitung der Bedeutungen werden solche Inkonsistenzen möglichst angeglichen. Dabei wird eine Liste von Literalen und regulären Ausdrücken verwendet, mit denen die häufigsten offensichtlichen Fälle von Bedeutungsklassen erkannt werden. Aktuell werden folgende Werte verwendet:

[1] Zum Teil wird die genaue Instanz bei einer solchen Angabe in Klammern doch weiter spezifiziert, z.B. „Art Fisch“ (leuciscus phoxinus) (REW, S. 3501). Diese Fälle werden als reguläre Bedeutungen behandelt.

value
Art
ein [A-ZÄÖÜ][a-zäöüß]+$
eine [A-ZÄÖÜ][a-zäöüß]+$
Stadt
Staat
Ort
Name
Bezeichnung
eine Art

Die regulären Ausdrücke ermöglichen dabei beispielsweise „ein Fisch“ von „ein Schiff anbinden“ zu unterscheiden. Einzelne Fälle von Bedeutungsklassen können auch durch eine entsprechende Ausnahme (vgl. Kap. 7.3) erkannt werden, die das Flag *meta* setzt, welches konventionelle Bedeutungen von Bedeutungsklassen in der Datenbank unterscheidet. An der Oberfläche wird eine Notation in spitzen Klammern zur Markierung dieser Form von Bedeutungsangaben verwendet, um diese von anderen Nutzungen von runden Klammern abzugrenzen.

10.2 Verknüpfung mit externen Konzepten

Zur Nachnutzbarkeit von lexikalischen Daten und zur Verwendung in technischen Anwendungen, ist eine Verknüpfung mit externen Normdaten unverzichtbar (vgl. auch Kap. 2.2). Aber auch ein Webportal kann für sich von einer solchen profitieren, indem einzelne Elemente durch die Vernetzung mit externen Portalen zusätzlich illustriert und mit weiterführender Information verknüpft werden. Im Fall von semantischer Information ist dies allerdings ungleich komplizierter, als bei solcher, die von Natur aus einer gewissen Normierung administrativer oder wissenschaftlicher Art unterliegt (z.B. geographische Einheiten oder Sprachen). Da die Bedeutung grundsätzlich beliebige Konzepte (oder Sachen) beschreiben kann, kommen fachspezifische Wissensdatenbanken in diesem Fall kaum in Frage, vielmehr wäre eine solche gefragt, die (zumindest grundlegend) die Eigenschaften einer Ontologie im philosophischen Sinn erfüllt. Bemühungen solche Normierungssysteme zu erstellen sind kein neues Phänomen (vgl. z.B. Hallig und Wartburg 1963 [1952]), aufgrund von moderner Webtechnologie sind allerdings grundlegend andere Möglichkeiten vorhanden kollaborativ ein solches System zu erstellen und ständig zu erweitern. Hierbei sind vor allem die beiden Web-Portale *DBpedia* und *Wikidata* relevant. Beide sind gewissermaßen aus *Wikipedia* entstanden, während *DBpedia* allerdings algorithmisch aus *Wikipedia* erzeugt wird und das Ziel hat die dort enthaltenen Daten im Kontext des *Semantic Webs* zugreifbar zu machen (vgl. Lehmann u. a. 2015), wurde *Wikidata* als zentrale Wissensdaten für *Wikipedia* (und andere *Wikimedia*-Projekte) geschaffen,

um die verschiedenen einzelsprachigen Versionen zu vereinheitlichen (vgl. z.B. Voß u. a. 2014, S. 3.1) und sieht sich selbst als „central storage for the **structured data** of its Wikimedia sister projects“ (Wikidata Startseite). Die Identifikatoren aus *Wikidata* werden inzwischen so oft in den verschiedensten Kontexten zur Normierung und Verknüpfung von projektspezifischen Daten verwendet, sodass sie bereits als zentrale Identifikatoren vorgeschlagen wurden, die die der andere Normdatenbanken ablösen könnten (Veen 2019). Im Bereich der Lexikographie wird *Wikidata* zum Teil ebenfalls eingesetzt (vgl. z.B. Lücke 2021a), beliebter scheint hier allerdings *DBpedia* zu sein (vgl. z.B. Abgaz 2020, Tittel und Chiarcos 2018), was wohl damit zusammenhängt, dass viele Ansätze in diesem Bereich sich sehr stark auf *Linked Open Data* beschränken und *DBpedia* vollständig darauf aufbaut[2]. Auch in der Spezifikation des *OntolexLemon* (vgl. Kap. 2.3) wird in den Beispielen *DBpedia* verwendet, auch wenn das Modell prinzipiell für sämtliche Ontologien (im informatischen Sinne) offen ist (Cimiano, John P. McCrae und Buitelaar 2016).

Ein großer Nachteil von *DBpedia* ist allerdings, dass dessen Einträge an die *Wikipedia*-Artikel gebunden sind und somit keine Darstellung von Konzepten möglich sind, die nicht dessen „Relevanzkriterien“ erfüllen. Im Gegensatz dazu können in *Wikidata* jederzeit neue Einträge angelegt werden, die unabhängig von *Wikipedia*-Artikel sind[3]. Somit ist *Wikidata* deutlich vielseitiger und bietet bessere Möglichkeit zur Erweiterung. Ein weiterer Nachteil von *DBpedia* ist außerdem, dass die Identifikatoren sprachabhängig sind, da die jeweiligen URLs auf den Titeln der jeweiligen *Wikipedia*-Sprachversion aufbauen, während *Wikidata* numerische Identifikatoren (mit dem Präfix *Q*) verwendet, die sprachunabhängig sind. An dieser Stelle werden für die Normierung der Bedeutungen des REW deshalb Identifikatoren aus *Wikidata* verwendet[4].

Die Verknüpfung mit Konzepten einer Wissensdatenbank hat außerdem noch einen weiteren (internen) Vorteil: Zum Teil können synonyme Bedeutungen aufgefunden und zusammengefasst werden (vgl. auch Kap. 12.3). Somit können beispielsweise sprachliche Formen mit der Bedeutung „Heuschrecke“ zusammen mit jenen der Bedeutung „Heupferdchen“ angezeigt werden, da beide Bedeutungsangaben mit der selben QID verknüpft sind. Voraussetzung für eine sinnvolle Vernetzung ist allerdings, dass die Bedeutungsangaben innerhalb der Quelle konsistent und unterscheidbar sind. Die Verknüpfung der Bedeutung „Elsbeere“ mit dem *Wikidata*-Konzept Frucht der Elsbeere und „Elsbeerbaum“ mit Elsbeere ist dabei kein grundsätzliches Problem, solange innerhalb des Quellenmaterials mit „Elsbeere“ konsequent die Frucht beschrieben wird. Es zeigt sich aber, dass die Verknüpfung von Bedeutungsangaben mit Konzepten bis zu einem gewissen Maß kontextabhängig ist und nicht ohne jegliche

[2] *Wikidata* bietet allerdings auch entsprechende Schnittstellen an.

[3] Auch hier sollten gewisse Kriterien beachtet werden (vgl. auch Kap. 10.2.2), diese sind allerdings deutlich weniger restriktiv.

[4] Ein Verknüpfung mit *DBpedia* könnte für Einträge, die einem Wikipedia-Artikel zugeordnet sind allerdings sogar automatisiert erstellt werden.

Einschränkungen für andere Quellen wiederverwendet werden kann. Problematisch sind allerdings Fälle, in denen die genaue Unterscheidung einer Bedeutung nur aus dem Artikelkontext klar wird. Ein Beispiel im REW sind die Artikel 2433 und 7687, die beide die Bedeutung „Glas“ enthalten. Aus dem Lemma und den restlichen Bedeutungen ist allerdings ersichtlich, dass in einem Fall ein Gefäß gemeint ist, während im zweiten Artikel das Material beschrieben wird. In solchen Fällen sollten beide Bedeutungen über eine Ausnahme eine zusätzliche Spezifikation erhalten, um sie (innerhalb der Quelle und bei der Verknüpfung nach außen) unterscheiden zu können.

10.2.1 Automatisierte Verknüpfung mit Wikidata

In *Wikidata* besteht wie bereits erwähnt der zentrale Identifikator eines *Items* aus dem Präfix *Q* und einer Nummer. Ein minimaler Eintrag muss weiterhin mindestens eine (englischsprachige) Bezeichnung, eine Beschreibung oder Aliase enthalten. Die weitere Definition erfolgt grundsätzlich über *Statements*, die den Eintrag mit anderen internen oder externen Elementen verknüpfen. Diese haben die Form von Tripeln, die zwei Entitäten mit Hilfe einer *Property* verknüpfen. Zusätzlich sind allerdings Übersetzungen der drei genannten Angaben in verschiedene Sprachen möglich und sinnvoll:

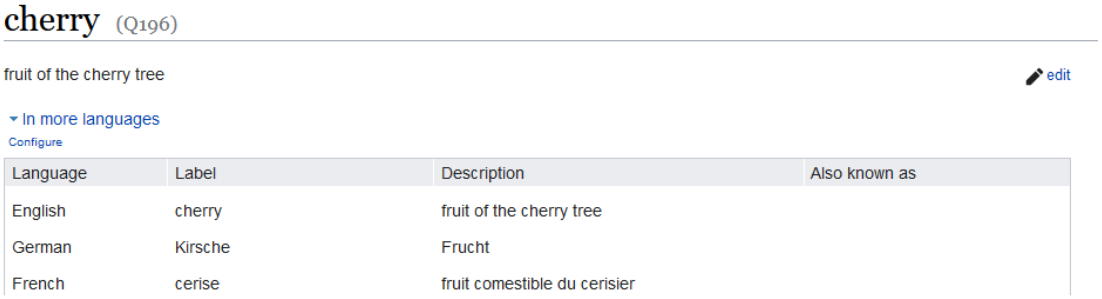


Abbildung 10.2: Ausschnitt aus dem *Wikidata*-Eintrag für das Konzept KIRSCHE

Die Beschreibungen sind dabei ausdrücklich nicht als vollständige Definition des entsprechenden Konzepts gedacht, sondern nur zur Unterscheidung von anderen Einträgen:

> The **description** on a Wikidata entry is a short phrase designed to disambiguate items with the same or similar labels. A description does not need to be unique; multiple items can have the same description, however no two items can have both the same label and the same description.
>
> [...]
>
> Descriptions are not full sentences, but small bits of information. In most cases, the proper length is between two and twelve words.(Wikidata Help:Description)

Beispiele für solche Beschreibungen sind „mythisches Wesen“ für die Bezeichnung „Basilisk“[5] oder „Spielkartenfarbe“ für die Bezeichnung „Eichel“[6]. Diese Kombination aus Bezeichnung und einer Beschreibung als Spezifikation entspricht somit in vielerlei Hinsicht auch der eher intuitiven und weniger definitorischen Darstellung der Bedeutungen in traditionellen Wörterbüchern und kann zum Teil für eine automatisierte Verknüpfung genutzt werden. Dazu kann folgende einfache *SPARQL*-Abfrage[7] an den *Wikidata Query Service*[8] geschickt werden, um beispielsweise das Konzept zu finden, das der Bedeutung „Kirsche“ entspricht:

```
SELECT distinct ?entry ?entryLabel ?entryDescription WHERE{
  ?entry ?label "Kirsche"@de.
  ?article schema:about ?entry.
  SERVICE wikibase:label { bd:serviceParam wikibase:language "en". }
}
```

Dabei werden alle *Items* zurückgegeben, die über den deutsche Namen „Kirsche“ verfügen[9]. Oftmals werden hier allerdings mehre Ergebnisse gefunden, so auch in diesem Fall:

entry	**entryLabel**	**entryDescription**
wd:Q8344151	Cerise	Wikimedia disambiguation page
wd:Q2740434	Kirsche	Wikimedia disambiguation page
wd:Q196	cherry	fruit of the cherry tree
wd:Q2741078	cherry	heraldic figure

Um die Resultate einzuengen werden bestimmte Überklassen (wie beispielsweise Familien- oder Markennamen) in der Abfrage ausgeschlossen:

```
SELECT distinct ?entry ?entryLabel ?entryDescription WHERE{
  ?entry ?label "Kirsche"@de.
  ?article schema:about ?entry.
  FILTER NOT EXISTS {?entry wdt:P31 wd:Q5}.                 #Personen
  FILTER NOT EXISTS {?entry wdt:P31 wd:Q167270}.            #Marken
  FILTER NOT EXISTS {?entry wdt:P31 wd:Q4167410}.           #Wikimedia-Begriffsklärun
  FILTER NOT EXISTS {?entry wdt:P31 wd:Q101352}.            #Familiennamen
  FILTER NOT EXISTS {?entry wdt:P31/wdt:P279* wd:Q15642541}.  #geographische Verwal
  FILTER NOT EXISTS {?entry wdt:P31/wdt:P279* wd:Q17537576}.  #kreative Werke
  FILTER NOT EXISTS {?entry wdt:P31/wdt:P279* wd:Q618123}.    #geographische Gebiet
  FILTER NOT EXISTS {?entry wdt:P31/wdt:P279* wd:Q43229}.     #Organisationen
```

[5] https://www.wikidata.org/wiki/Q152519
[6] https://www.wikidata.org/wiki/Q1301333
[7] https://www.w3.org/TR/rdf-sparql-query/
[8] https://query.wikidata.org/
[9] Es werden dabei alle drei Felder *Bezeichnung*, *Beschreibung* und *Aliase* durchsucht.

```
  SERVICE wikibase:label { bd:serviceParam wikibase:language "en". }
}
```

In diesem Beispiel kann das Ergebnis so auf einen einzigen (korrekten) Treffer eingeschränkt werden. Beim systematischen Anwenden dieser Anfrage auf alle 24157 zum damaligen Zeitpunkt nicht mit einer QID verbundenen Bedeutungen[10] konnten 13,3 Prozent eindeutig einem *Wikidata*-Konzept zugeordnet werden, 8,4 Prozent lieferten mehrere Kandidaten zurück[11]. Die gefundenen Konzepte beschränken sich dabei sehr deutlich auf Bedeutungen, die nur aus einem Token bestehe. Für alle weiteren ist die Anzahl der gefundenen QIDs vernachlässigbar:

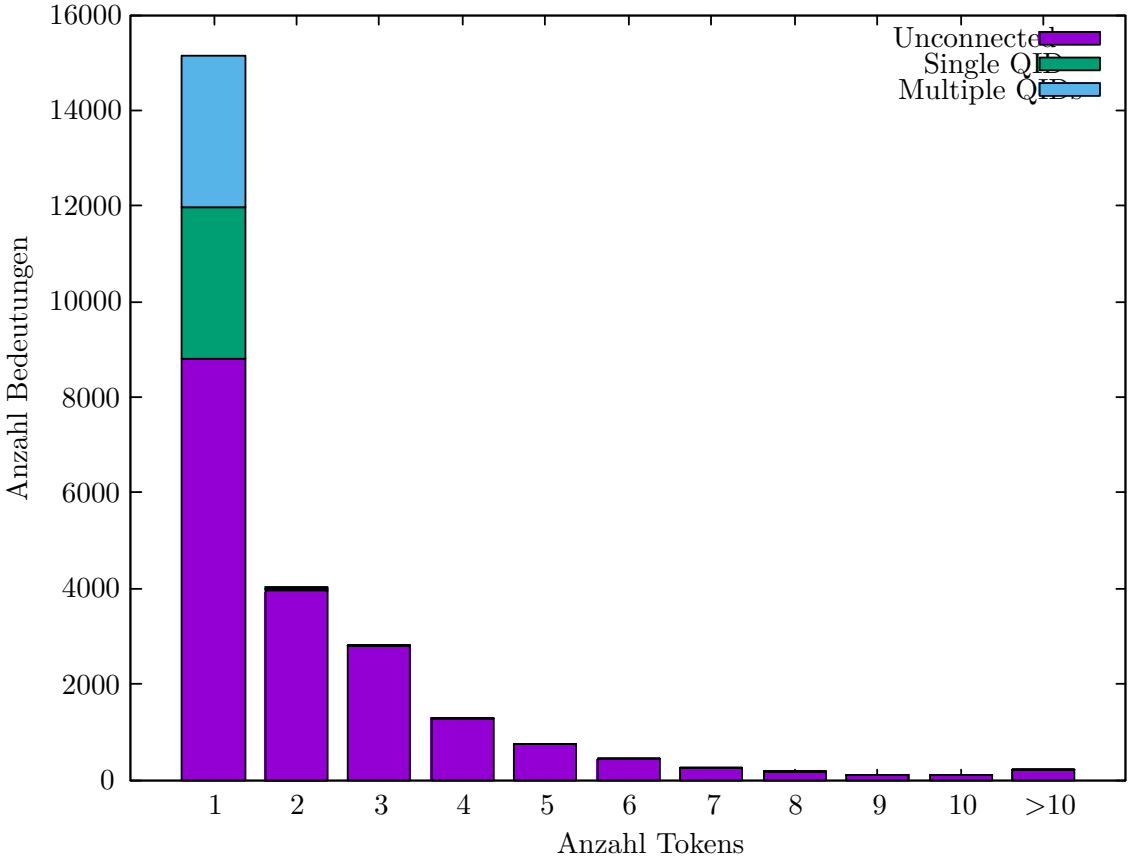


Abbildung 10.3: Anzahl der Tokens pro Bedeutungsbeschreibung und Erfolgquote bei der Zuordnung von QIDs

Für die relativ niedrige Anzahl von gefundenen Konzepten, können zwei Faktoren

[10] Dies ist der Großteil aller aus dem REW extrahierten Bedeutungen. Im Gegensatz dazu waren 497 Einträge im Rahmen verschiedener Tests bereits mit QIDs verknüpft.

[11] In der Liste der Bedeutungen ist ein gewisser Anteil an OCR-Fehlern zu erwarten. Zum Zeitpunkt des *Wikidata*-Abgleichs waren bei einer Stichprobe von 500 zufällig gewählten Werten insgesamt 15 fehlerhafte Bedeutungsbeschreibungen vorhanden (= 3%), was sich nicht signifikant auf die Zahlen auswirken sollte.

verantwortlich sein. Zum einen sind die sprachabhängigen Daten in vielen Fällen unvollständig oder entsprechen nicht den eigentlichen Konventionen. Das hängt zum Teil auch damit zusammen, dass große Teile über automatische Importe erzeugt wurden und erst nach und nach verbessert werden. Wenn keine deutschen Bezeichnungen oder Aliase vorhanden sind oder diejenigen die im REW verwendet werden nicht vorkommen, kann selbstverständlich auch keine automatisierte Verknüpfung stattfinden. Auch fehlende *Statements* können ein Grund sein, dass kein eindeutiges Konzept gefunden wird. Viele spezifischere Konzepte sind aber auch unabhängig von der Benennung nicht in *Wikidata* vorhanden. Kap. 10.2.2 beschäftigt sich näher mit dieser Thematik. Die Abfragen, die mehrere Ergebnisse liefern, können zwar an dieser Stelle nicht zur automatisierten Verknüpfung mit Konzepten verwendet werden, im Kontext des Webportals können sie aber angewandt werden, um Nutzenden eine Liste von Kandidaten zur Auswahl zur Verfügung zu stellen.

Bei einer Untersuchung der Bedeutungen, denen eine eindeutige QID zugeordnet werden konnte, wurden von 500 zufälligen Bedeutungen 442 dem korrekten *Wikidata*-Konzept zugeordnet (88,4%). Drei weitere wurden einem leicht unterspezifiziertem Konzept zugewiesen (beispielsweise „Straßenjunge“ zu Straßenkind). Bei den restlichen wurde ein fehlerhafter Eintrag gefunden, in den meisten Fällen ein zu spezifisches Konzept oder eines, das über einen homonymen Ausdruck bezeichnet wird. Im Bezug auf Wortarten gab es zwischen Substantiven[12] und Adjektiven[13] keine Unterschiede in der Erkennungsrate, bei den 16 Verben gab es keine fehlerhaft zugewiesenen Konzepte. Aufgrund der geringen Menge von Verben und Adjektiven sind diese Werte allerdings wenig aussagekräftig. Unabhängig von der Erkennungsrate scheint sich der Eindruck zu bestätigen, dass in *Wikidata* hauptsächlich nominalisiert bezeichnete Konzepte existieren, da die Anteil der Nicht-Substantive in den verknüpften Bedeutungen, deutlich geringer ist, als der im gesamten Datenbestand[14]. Das heißt aber nicht zwingend, dass die entsprechenden Konzepte nicht vorhanden sind, sondern nur, dass sie meist nicht automatisiert verknüpft werden können. Grundsätzlich sollte die Verknüpfung von Konzept und Bedeutung unabhängig von grammatikalischen Eigenschaften der Bedeutungsangabe sein. So werden beispielsweise auch die Bedeutungen „Frucht“ und „Obst“ mit der gleichen QID verknüpft, da der Numerus nur auf Basis der sprachlichen Form relevant ist. In vielen Fällen kann eine verbale Bedeutung auch mit einem nominal formulierten Konzept verknüpft werden (z.B. „behandeln“ (einer Krankheit) mit treatment), sodass eine Beurteilung solcher Fälle schwierig ist und einer eigenen Untersuchung bedürfte.

[12] 420 / 472 korrekt erkannt = 89,0%

[13] 8/ 9 korrekt erkannt = 88,9%

[14] Die Lemmata des REW wurden mit einer sehr einfachen Methodik aus Basis von lateinischen Wortendungen und Groß- und Kleinschreibung und Endungen der deutschen Bedeutungen Wortarten zugeordnet. Die Einschätzung der Verteilung der Wortarten beruht auf dieser Daten, die allerdings nicht unbedingt repräsentativ für alle Formen im REW sind. Die Verteilung für Substantive, Verben und Adjektive ist dabei 65,2%, 23,1% und 11,7%, während sie in der Stichprobe der automatisiert verknüpften Daten 95%, 3,2% und 1,8% sind.

10.2.2 Behandlung von Bedeutungen ohne direkte Entsprechung

Wie bereits im vorherigen Kapitel erwähnt, können gerade längere Bedeutungsangaben nicht unbedingt mit *Wikidata* verknüpft werden, da sehr spezialisierte Konzepte dort nicht enthalten ist. Die Plattform selbst definiert Kriterien dafür, welche Konzepte angelegt werden sollen, die vereinfacht formuliert folgende neuen Einträge erlauben:

- Einträge, die auf eine Seite eines bestehenden Wikimedia-Projekts verweisen.
- Das Konzept wird in einer „ernsthaften“ öffentlich verfügbaren Online-Ressource beschrieben.
- Es wird gebraucht, um *Statements* für andere Einträge zu definieren.

Die genaue Formulierung findet sich in Wikidata:Notability. Nun erlaubt gerade das zweite Kriterium einen gewissen Interpretationsspielraum, unabhängig davon stellt sich allerdings die Frage, wie sinnvoll es wäre für jede Bedeutungsangabe ein *Wikidata*-Konzept anzulegen, besonders wenn die Bedeutung nach heutigen Maßstäben nicht mehr relevant ist oder es um einzelne Dialektbegriffe geht, deren Bedeutung ansonsten nicht vorkommt. Beispiele für solche Bedeutungsangaben sind „kleiner, runder Käse“ (REW, S. 8509) oder „die drei letzten Tage des März und die drei ersten des Aprils“ (REW, S. 9109). Eine einfache 1:1 Verknüpfung kommt in solchen Fällen eher nicht in Frage, es wäre aber gerade beim ersten Beispiel eine „Beschreibung“ mit Hilfe von mehreren *Statements* aus dem Vokabular möglich, das auch in *Wikidata* selbst verwendet wird. Gerade im Kontext des *Semantic Webs* liegt es nahe sich von der eigenartig starren Vorstellung einer direkten Identität bei allen semantischen Beziehungen zu verabschieden, wie sie auch das *ontoLex Lemon* Modell vorsieht (vgl. Kap. 2.3.2). Die gewohnten direkten Beziehungen können über die *Property* exact match abgebildet werden. Komplexere Fällen wie das Beispiel „kleiner, runder Käse“ würden in einer solchen Modellierung durch mehrere *Statements* beschrieben werden:

subclass of (P279)	Käse (Q10943)
has quality (P1552)	klein (Q24245823)
has quality (P1552)	rund (Q59564206)

Anstatt Tripeln werden hier nur zwei Werte angegeben, da das Subjekt jeweils die Bedeutung bzw. das durch sie beschriebene Konzept selbst ist.

Anmerkung zur *Property* „has quality“: Diese *Property* und ihre Verwendung ist in *Wikidata* zum Teil inkonsequent. Sie wird definiert als „the entity has an inherent or distinguishing non-material characteristic“[15]. Anhand der Beispiele ist sie gedacht, um ein Konzept mit einer abstrakten Eigenschaft zu verknüpfen, die verschiedene Ausprägungen hat. Dies ist beispielsweise so für die Tripel (human, has_quality, gender) oder (quark, has_quality, baryon number), die dort exemplarisch genannt

[15] https://www.wikidata.org/wiki/Property:P1552

werden. Für die Zuweisung von konkreten Eigenschaften zu einem konkreten Objekt werden dann spezialisiertere *Properties* wie sex or gender verwendet, mit der einer bestimmten Person ein bestimmtes Geschlecht zugewiesen werden kann. Ein allgemeinere Formulierung einer *Property,* die einem Konzept eine konkrete Ausprägung einer Eigenschaft zuweist, ist zum aktuellen Zeitpunkt nicht vorhanden. Somit sind auch innerhalb von *Wikidata Statements* wie im obigen Beispiel vorhanden, um beispielsweise eine Größenangabe zuzuordnen (vgl. z.B. (dwarf, has_quality, small) in Q214045).

Gerade Unterklassen von bekannten Konzepten, die genauer spezifiziert werden (oder auch die unterspezifizierten Bedeutungsklassen aus Kap. 10.1) können mit einer solche Formulierung oftmals abgebildet werden. Schwieriger sind Bedeutungsbeschreibungen, die auf Ähnlichkeit zu einem anderen Konzept beruhen (z.B. „ein sackähnliches Netz“), da eine *Property* der Form „Ähnlich zu etwas“ aktuell nicht vorhanden ist. Grundsätzlich könnten natürlich auch zusätzliche *Properties* unabhängig von deren Verwendung in *Wikidata* definiert werden, die Erstellung einer eigenen (Teil-)Ontologie soll hier allerdings vermieden werden. Auch ist eine exakte Beschreibung eines bestimmten Konzepts zwar optimal, aber nicht immer realistisch. Bei sehr komplexen Bedeutungsbeschreibungen stößt der Ansatz der exakten Abbildungen schnell an seine Grenzen. Bedeutungen wie „Halsband aus drei Stücken, das den Schweinen umgebunden wird, damit sie nicht durch die Hecken brechen“ (REW, S. 1600) können mit einem formellen ontologischen Vokabular kaum realistisch beschrieben werden. Dies ist allerdings auch nicht unbedingt nötig. Technische Anwendungen auf den Daten könnten allein von grundlegender semantischer Information deutlich profitieren. In diesem Fall wäre eine einfache Verknüpfung über *subclass of* mit dem Konzept Halsband für die meisten realistischen Anwendungsfälle bereits ausreichend. Im Unterschied zu klassischen gröberen semantischen Klassifizierungen wie in Hallig und Wartburg 1963 [1952] ist in diesem Fall die genaue Art der Relation (also vor allem *subclass of* oder *exact match*) nachvollziehbar. Auch eine (teil-)automatisierte Verknüpfung von längeren Bedeutungsbeschreibungen mit entsprechenden Überklassen erscheint realistischer als bei der Verwendung einer exakten Abbildung des jeweiligen Konzepts. Diese kann trotzdem im Einzelfall angelegt werden, eine Verfügbarmachung solcher Relationen für eine eventuelle Nachnutzung (vgl. Kap. 13.2) ist dabei allerdings umso wichtiger.

11 Vernetzung und Anreicherung

Dieses Kapitel geht kurz auf die Zuordnung von Literatur und Sprachen zu entsprechenden externen Ressourcen ein, welche die wissenschaftliche Arbeit erleichtern können, indem Verlinkungen und Visualisierungen zur Verfügung gestellt werden.

11.1 Vernetzung der Literaturangaben

Ein gewisser Teil der im REW referenzierten Werke ist frei zugänglich. Es sind zum aktuellen Zeitpunkt zwar keine höheren „Digitalisierungsgrade“ (Lücke 2016) vorhanden[1], für viele gibt es allerdings Scans, die seitengenau referenziert werden können. Eine wichtige Quelle ist hier die Open Library des sogenannten *Internet-Archivs*. Die folgenden Tabelle zeigt die für einer Verlinkung notwendigen Daten:

abbreviation	volume	page_start	page_end	link	prefix	suffix	offset
Diez				https://archive.org/details/etymologischesw05diezgoog	/page/n	/mode/1up	33
Zs.	11	1	152	http://dfg-viewer.de/show?set[image]=6&set[mets]=http%3A%2F%2Fwww.digizeitschriften.d			
Zs.	11	153	288	http://dfg-viewer.de/show?set[image]=10&set[mets]=http%3A%2F%2Fwww.digizeitschriften.d			
Zs.	11	289	432	http://dfg-viewer.de/show?set[image]=14&set[mets]=http%3A%2F%2Fwww.digizeitschriften.d			
Zs.	11	433	588	http://dfg-viewer.de/show?set[image]=18&set[mets]=http%3A%2F%2Fwww.digizeitschriften.d			
Zs.	13			https://archive.org/details/zeitschriftfrro15pfisgoog	/page/n	/mode/1up	9

Tabelle 11.1: Vernetzungsdaten für literarische Quellen

[1] Eine Ausnahme ist das FEW, das im REW als „Wartburg“ referenziert wird Für dieses werden allerdings keine konkreten Seitenzahlen verwendet.

Das Werk wird über eine Abkürzung und (optional) die Nummer eines Bands beschrieben. Die URL kann dann aus einer Basis-URL und der Seitennummer zusammengefügt werden. Die Basis-URL sollte hierbei auf die Quelle selbst (also eine entsprechende Startseite oder ähnliches) verweisen, sodass diese auch für Verweise ohne konkrete Seitenzahl verwendet werden kann. Somit sind weiterhin zwei Felder *prefix* und *suffix* vorgesehen, die auf die Basis-URL folgen und jeweils vor und nach der Seitenzahl angefügt werden. Das Feld *offset* beschreibt, ab welcher Seite die Nummerierung beginnt, da oftmals die erste Seite des Scans nicht mit der Seite eins übereinstimmt. Zum Teil ist allerdings auch die Nummerierung innerhalb des Scans nicht streng aufsteigend, was beispielsweise bei Fehlern im gescannten Material oder nicht paginierten zusätzlichen Seiten der Fall ist. Hierfür kann mit *page_start* und *page_end* ein Bereich der Quelle angegeben werden, für die das Feld *offset* gültig ist. In diesem Fall müssen somit für einen Band mehrere Zeilen angelegt werden, die einzelne Passagen beschreiben (vgl. „Zs. 11“ in der obigen Tabelle).

Wird eine Quelle nicht über Seitenzahlen, sondern über eine Lemmanummer referenziert, ist die Verknüpfung aufwendiger. In einem solchen Fall ist zusätzlich eine Abbildung von Seitenzahlen auf Lemmanummern notwendig:

abbreviation	page	start_entry	end_entry
Lokotsch	1	1	10
Lokotsch	2	11	20
Lokotsch	3	21	32
Lokotsch	4	33	51
Lokotsch	5	52	64

Tabelle 11.2: Zuordnung von Seiten zu Lemmanummern

Diese kann oftmals automatisiert aus den Kopfzeilen in der Volltextdarstellung der jeweiligen Quelle erzeugt werden. Je nach Qualität der zugrundeliegenden Texterkennung erfordert dies allerdings trotzdem vielfach eine erhebliche Nachkorrektur. Umso wichtiger ist somit, dass solche Ergebnisse zugänglich und damit nachnutzbar sind (vgl. Kap. 13.2).

11.2 Räumliche Zuordnung von Sprachen und Dialekten

Die räumliche Zuordnung von Sprachen findet auf Basis der Sprachabkürzung und einer eventuellen zusätzlichen geographischen Einschränkung (z.B. „frz.“ und „südfrz.“)

statt. In diesem Fall wurden hierzu folgenden Grundannahmen verwendet:

- Für die romanischen Nationalsprachen wird aus pragmatischen Gründen jeweils nur der zugehörige Nationalstaat verwendet. Diese Entscheidung ist zum einen auf die bloße Existenz von Angaben wie „südfrz." oder „nordit." gegründet, deren Formulierung zumindest andeutet, dass auch die jeweiligen Länder gemeint sind. Zum anderen werden speziell abweichende Formen oftmals mit Formulierungen wie „schweiz." oder ähnlich bezeichnet. Grundsätzlich bleibt diese Konvention trotzdem streitbar, sie erlaubt allerdings weiträumigere geographische Visualisierungen, die sonst nicht so einfach möglich wären.
- Bei mehrdeutigen Angaben (wenn also beispielsweise eine Gemeinde und eine Provinz den gleichen Namen haben) wird jeweils die hierarchisch höchste Ebene verwendet.

Somit kann ein Teil der Mundartsbezeichnungen der Form „MA. von ..." (oder ähnlich) automatisiert mit einem Eintrag der *Geonames*-Datenbank verbunden werden. Grundlage hierfür sind API-Abfragen der Form *http://api.geonames.org/search?username=xxx&name=Palermo*, deren Ergebnisse anhand der verschiedenen Kategorien von Ortsbezeichnungen und der zweiten oben genannten Konvention gefiltert werden. Zusätzlich wird von *Geonames* im Falle von administrativen Einheiten der offizielle Name übernommen, falls im REW eine andere Namensvariante vorkommt. Dies hat den Grund, dass Polygondaten über *Geonames* nicht ohne weiteres abgreifbar sind. Somit werden diese anhand der offiziellen Namen aus dem Datenbestand des GADM-Projekts gewonnen, das Polygondaten für die Grenzen von Verwaltungseinheiten weltweit zur Verfügung stellt. Ein Teil der vorhandenen Sprachen kann automatisiert verknüpft werden, andere können manuell nachgetragen werden (vgl. Kap. 12.5.3).

11.3 Vereinfachung zu Hexagonen

Die Visualisierung im Webportal (vgl. Kap. 12.3) dient eher der Illustration der grundsätzlichen Möglichkeiten und ist soll somit auf möglichst unkomplizierte Weise funktionieren. Da die originalen Polygone im Bereich von Überlappungen schwer darzustellen sind, wird hier eine vereinfachte Darstellung mit Hilfe eines Hexagongitters verwendet. Somit reduziert sich der Fall von Überschneidungen auf gemeinsame Hexagone. Um diese Darstellung zu erzeugen, wird ein Ausschnitt des Gitters von Hexagonen mit einem festen Durchmesser über den Bereich des jeweiligen Polygons gelegt und nur die diejenigen Hexagone behalten, die sich mindestens zu 50% überschneiden. Abb. 11.1 illustriert diesen Vorgang. Eine Ausnahme wird bei sehr kleinen Polygonen gemacht, für die für kein Hexagon der Grenzwert überschritten wird. Hier wird das Hexagon mit der größten Überschneidung verwendet.

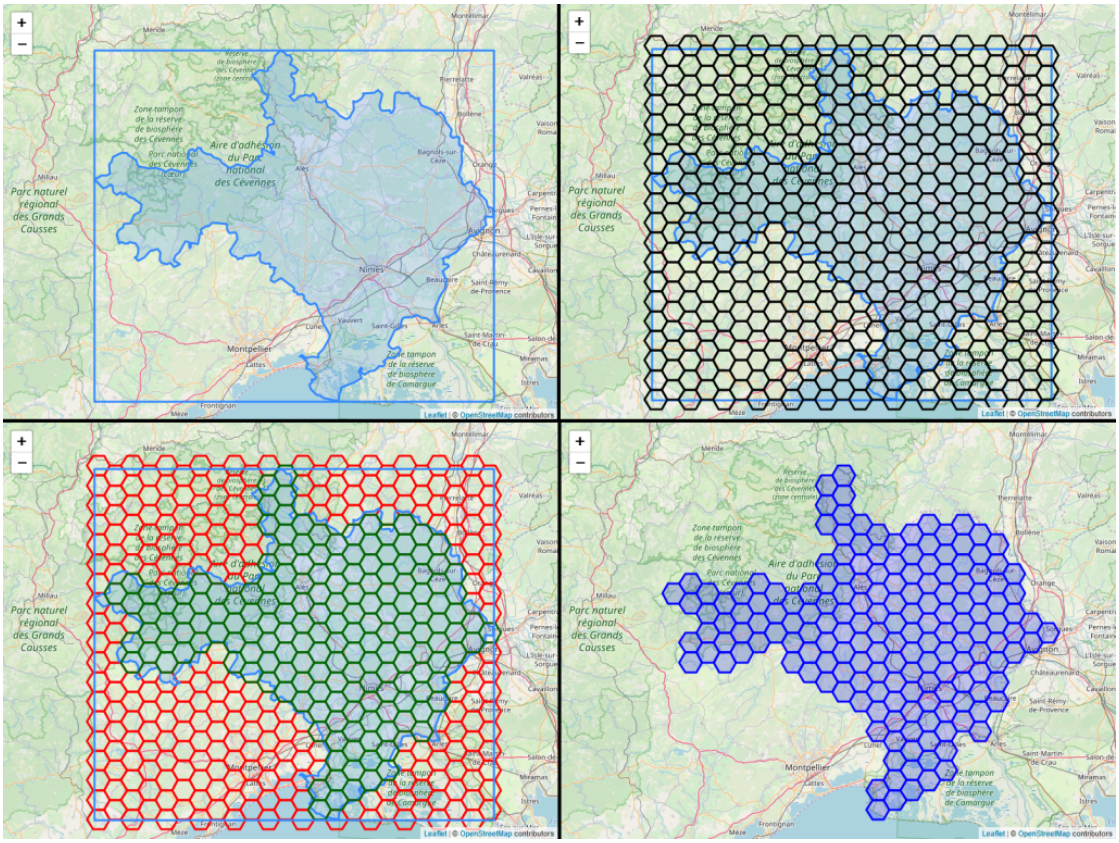


Abbildung 11.1: Teilschritte der Erstellung von entsprechenden Hexagondarstellungen der Polygone

12 Publikation

Auf Basis aller in den bisherigen Kapiteln erstellten Daten soll hier nun die grundsätzliche Konzeption eines Webportals beschrieben werden, welches Nutzenden einen möglichst effizienten und intuitiven Zugang zum erschlossenen Datenmaterial bietet[1]. Eine exemplarische Umsetzung steht unter rew-online.gwi.uni-muenchen.de zur Verfügung. Ein grundlegendes Merkmal sind dabei unterschiedliche Zugriffswege, die im Kern verschiedenen linguistischen Fragestellungen entsprechen (Kap. 12.1). Weiterhin sind alle Objekte (nicht nur die eigentlichen Wörterbuchartikel, sondern auch sprachliche Formen, Bedeutungen, Sprachangaben und literarische Quelle) einer eindeutigen ID und über diese einer URL zugeordnet, die Details zu diesen und ihre Verknüpfung mit anderen Objekten im Rahmen des Quellenmaterials enthält.
Kap. 12.2 beschäftigt sich mit den eigentlichen Wörterbuchartikeln, während Kap. 12.3 die weiteren Entitäten behandelt. Kap. 12.4 zeigt Beispiele für eine statistische Auswertung des Gesamtmaterials. Entsprechend der Konzeption aus Kap. 3.3 sind weiterhin verschiedene Interaktionsmöglichkeiten vorhanden, mit denen die generierten Resultatdaten korrigiert und Verknüpfungen mit externen Ressourcen angelegt werden können (Kap. 12.5).

12.1 Zugriffswege und Suchfunktionalitäten

Der erste und prominenteste Zugangsweg bleibt die Anordnung bzw. Nummerierung der Lemmata in der ursprünglichen Quelle:

[1] Möglichkeiten für die technische (Nach)nutzung werden im folgenden Kapitel besprochen

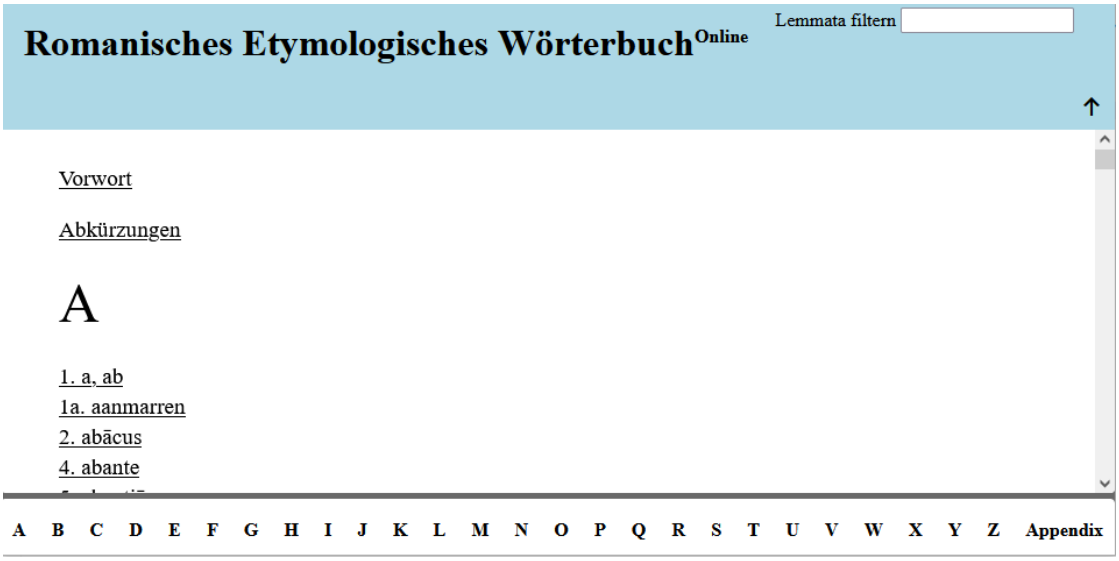


Abbildung 12.1: Artikelliste im Webportal (Link)

Als Zusatzfunktionalität ist hier nur eine Filterung der Lemmata nach einem beliebigen Suchstring (ohne Berücksichtigung von Diakritika oder Groß-/Kleinschreibung) und eine Verlinkung der jeweiligen Anfangsbuchstaben vorhanden. Somit ist sowohl ein Überblick über die Lemmatisierung des Quelle möglich, als auch ein schneller Zugriff auf spezifische Einträge.

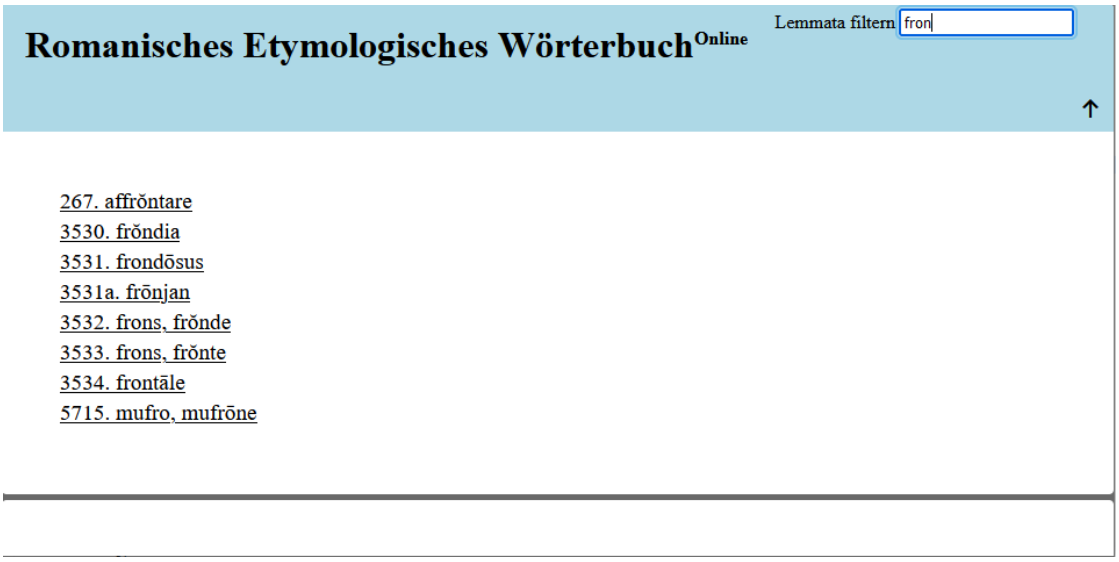


Abbildung 12.2: Gefilterte Artikelliste

Die Kopfzeile enthält Icons für die weiteren Funktionalitäten, vor allem eine Volltextsuche und eine Suche innerhalb spezifischer Entitäten oder nach etymologischen Relationen. Die Ergebnisse der Volltextsuche basieren dabei auf dem

originalen Artikeltext und markieren alle Vorkommen in diesem:

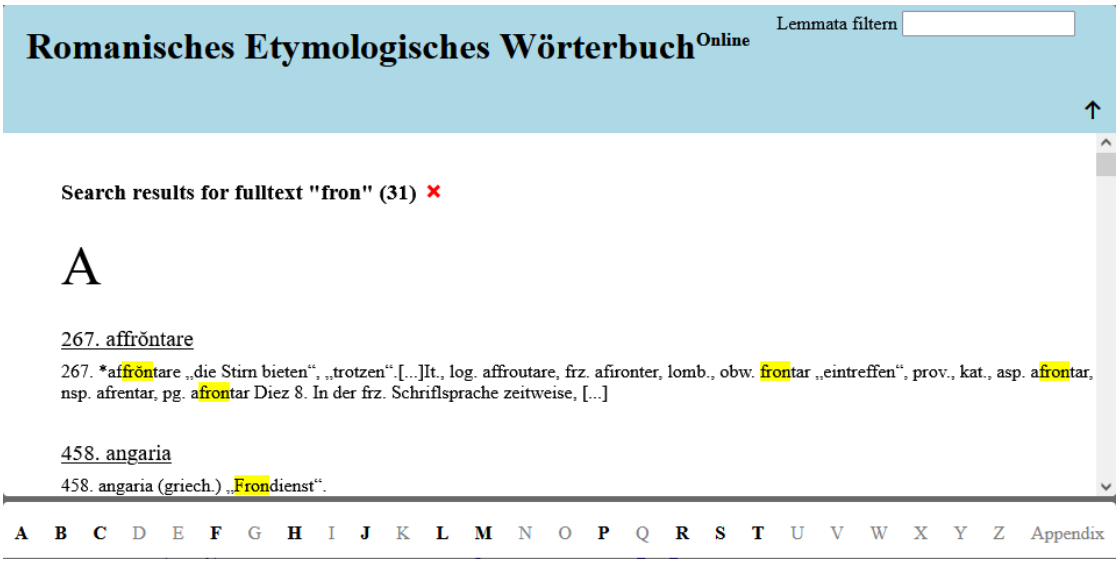


Abbildung 12.3: Volltextsuche im Quellenmaterial (Link)

Die Ergebnisse jeder Art von Suche werden grundsätzlich innerhalb der Startseite in die Artikelliste eingebettet. Die Filterung der Lemmata ist dabei unabhängig von der aktuellen Darstellung, so dass die Ergebnisse einer Suchabfrage weiter eingeschränkt werden können. Mit Ausnahme der Volltextsuche finden sich alle weiteren Zugriffsmöglichkeiten über das Icon *Recherchewerkzeuge*:

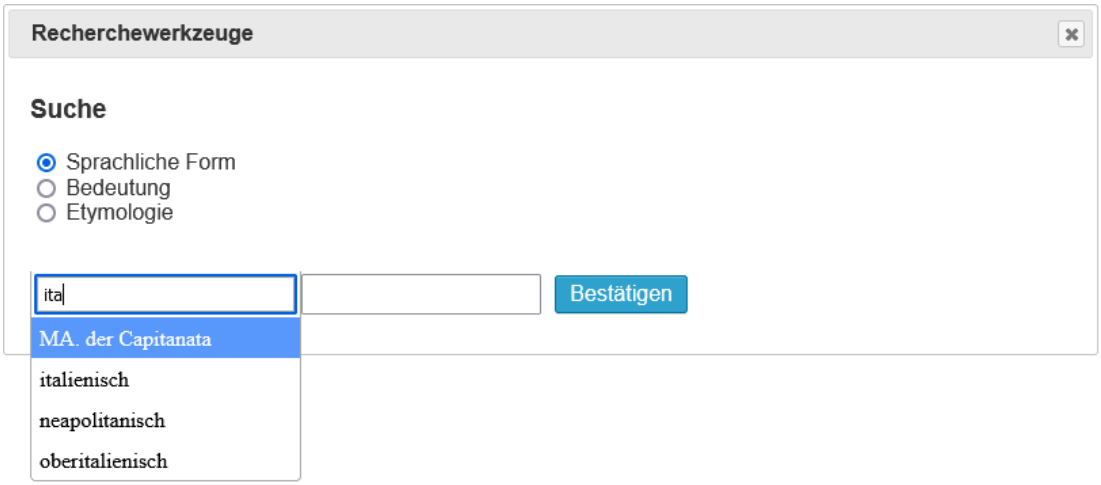


Abbildung 12.4: Spezifische Suchmöglichkeiten

Diese Werkzeuge erlauben die Suche nach sprachlichen Formen, Bedeutungen oder der Herkunft einer spezifischen Form. Die Auswahl einer Sprache ist bei der Suche nach einer Bedeutung nicht vorhanden, bei sprachlichen Formen ist sie optional. Ohne

Sprachzuordnung werden alle Formen die orthographisch den Suchstring enthalten aufgefunden:

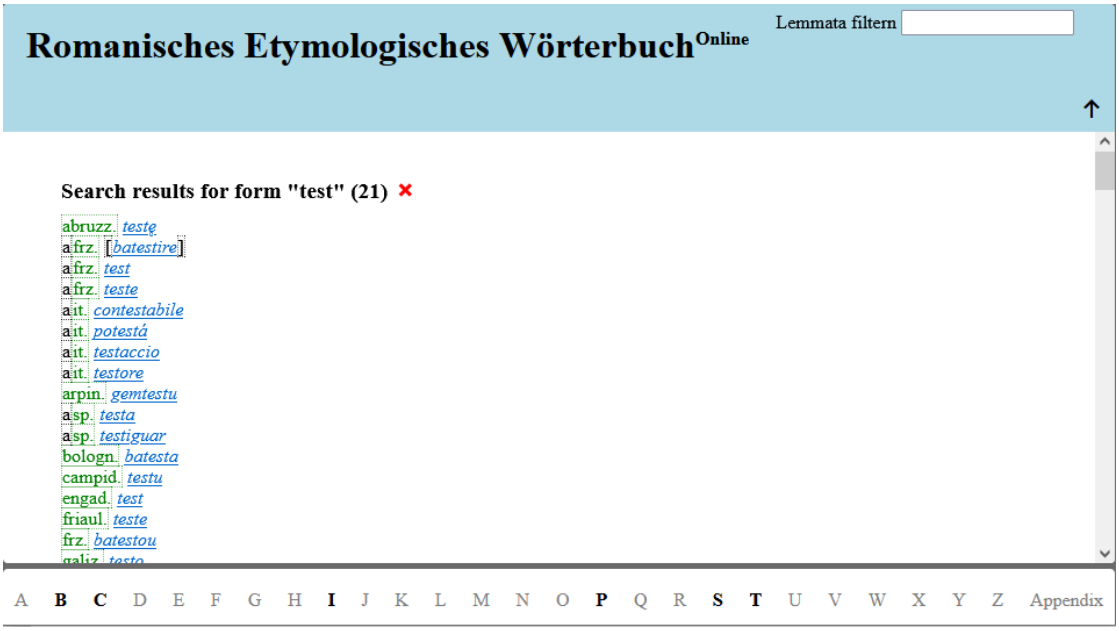


Abbildung 12.5: Suche nach sprachlichen Formen ohne Spezifikation einer Sprache (Link)

Im anderen Fall werden nur solche Formen zurückgegeben, die zur Sprache selbst oder einer untergeordneten Sprache / Mundart gehören. So werden im folgenden Beispiel nur französische Formen gesucht:

Abbildung 12.6: Suche nach sprachlichen Formen mit Spezifikation einer Sprache (Link)

Wie man sieht werden zu Beginn alle aufgefundenen Formen[2] gelistet, worauf die Artikel folgen, die mindestens eines der Suchergebnisse enthalten. An beiden Stellen sind die jeweiligen Detailseiten (vgl. Kap. 12.3) verlinkt. Analog werden die Ergebnisse einer Suche nach Bedeutungen dargestellt:

[2]Eine genauere Betrachtung der Darstellung der verschiedenen Elemente findet sich in Kap. 12.2.

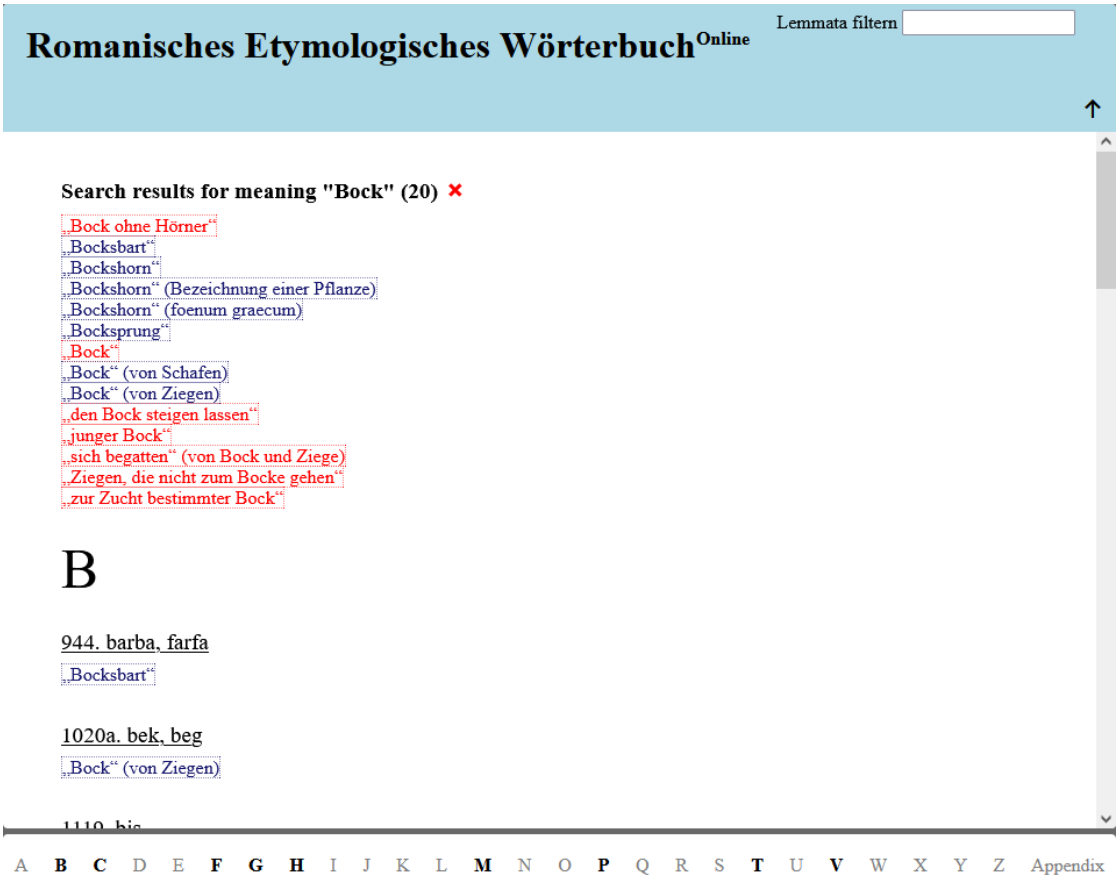


Abbildung 12.7: Suche nach einer Bedeutung (Link)

Zu beachten ist hier, dass die Suche orthographisch auf allen im REW vorkommenden Bedeutungsangaben arbeitet. Das heißt synonyme Varianten können auf dieser Ebene vorkommen. In diesem Fall sind die Bedeutungsangaben „Bockshorn“, „Bockshorn (Bezeichnung einer Pflanze)“ und „Bockshorn (foenum graecum)“ mit der gleichen *Wikidata*-QID verknüpft. Jede davon hat zwar eine eigene URL, inhaltlich sind die jeweiligen Detailseiten allerdings gleich (vgl. Kap. 12.3).

Mit der letzten Form der Suche kann die Herkunft einer bestimmten Form untersucht werden, d.h. es werden alle Etyma und die zugehörigen Artikel angezeigt:

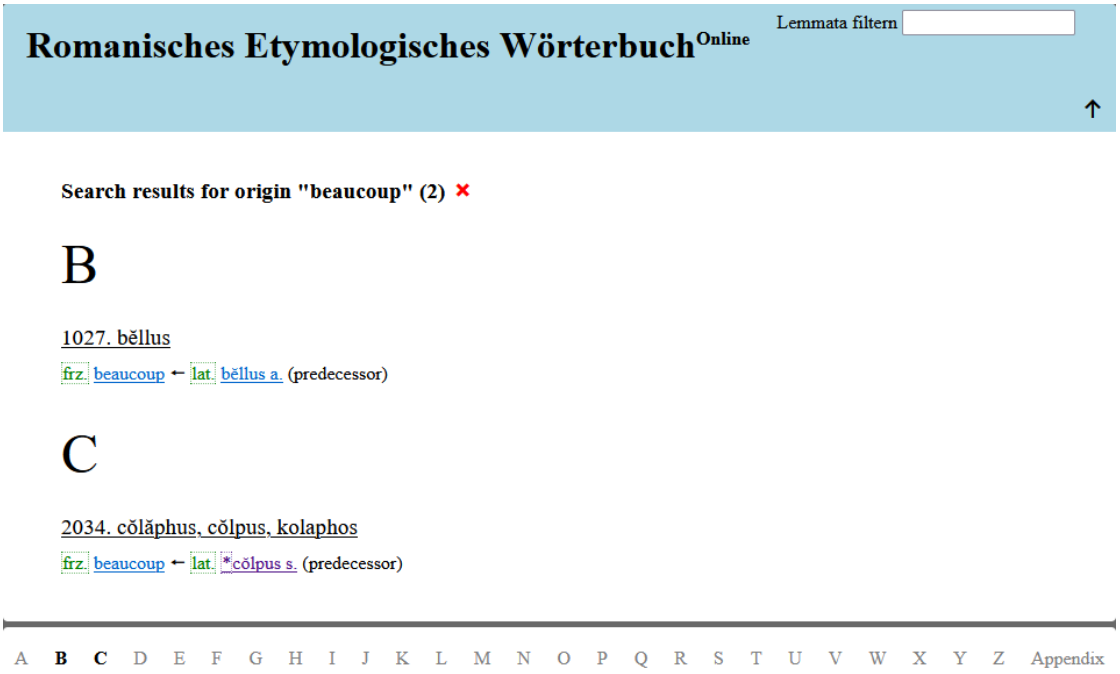


Abbildung 12.8: Suche nach der Wortherkunft (Link)

Zusammenfassend stehen somit verschiedene Zugangswege zur Verfügung, insbesondere ist auch ein onomasiologischer Zugang über die Suche nach Bedeutungen möglich. Einzelsprachliche Analysen (gerade in den Hauptsprachen) können ebenfalls einfach durchgeführt werden. In vielen Fällen kann auch nach einer übergeordneten Sprache wie „galloromanisch“ gesucht werden, auch wenn hier die Qualität der Suchergebnisse stark von der Vollständigkeit der entsprechenden Sprachhierarchien (vgl. auch Kap. 12.5.3) abhängt. Ein Zugang aus sprachhistorischer Perspektive ist nicht nur von den historischen Formen aus (anhand der Lemmatisierung des REW), sondern auch von den modernen Formen aus möglich.

12.2 Darstellung der Artikel

Eine erste Entscheidung, die bei der digitalen Darstellung von Wörterbuchartikeln getroffen werden muss, ist, ob die Artikel als unabhängige Seiten (vgl. z.B. TLIO) oder zusammen mit vorangegangenen und nachfolgenden Artikeln (vgl. z.B. Wörterbuchnetz) dargestellt werden. Je kleinteiliger die Lemmatisierung und je größer damit die Zusammenhänge zwischen aufeinanderfolgenden Artikeln sind, desto sinnvoller erscheint die zweite Variante. Im Fall des REW werden allerdings verhältnismäßig viele Lemmata zu einem Artikel zusammengefasst (vgl. Abb. 12.9), sodass hier die Artikel einzeln angezeigt werden.

4827. **lactaria** 1. „milchgebend“, 2. „Milchkuchen“, 3. **herba lactaria** „milchiges Kraut“.

Abbildung 12.9: Kopfzeile von REW, S. 4827

Abb. 12.10 zeigt die standardmäßige Ansicht eines solchen Artikels im Webportal. Im Gegensatz zum Originaltext, der aufgrund der Platzzwänge eines gedruckten Werks sehr dicht gedrängt gesetzt wurde, liegt hier der Fokus auf Übersichtlichkeit und Lesbarkeit. Die einzelnen Belege einer Belegliste werden als vertikale Aufzählungen dargestellt, während Entlehnungen entsprechend eingerückt auf die jeweilige Form folgen. Die Reihenfolge der Sprachbelege entspricht allerdings streng der im Ausgangsmaterial, um keine Information zu verlieren, die beispielsweise spezielle Separatoren zwischen den einzelnen Belegen enthalten. Die einleitenden Elemente zusätzlicher Listen (in diesem Fall die Liste der Ableitungen) werden prominenter dargestellt, sodass die einzelnen Bestandteile des Artikels klar unterschieden werden können. Für diskursive Elemente bietet sich keine besondere Formatierung an, sodass diese grundsätzlich wie im Quellentext dargestellt werden.

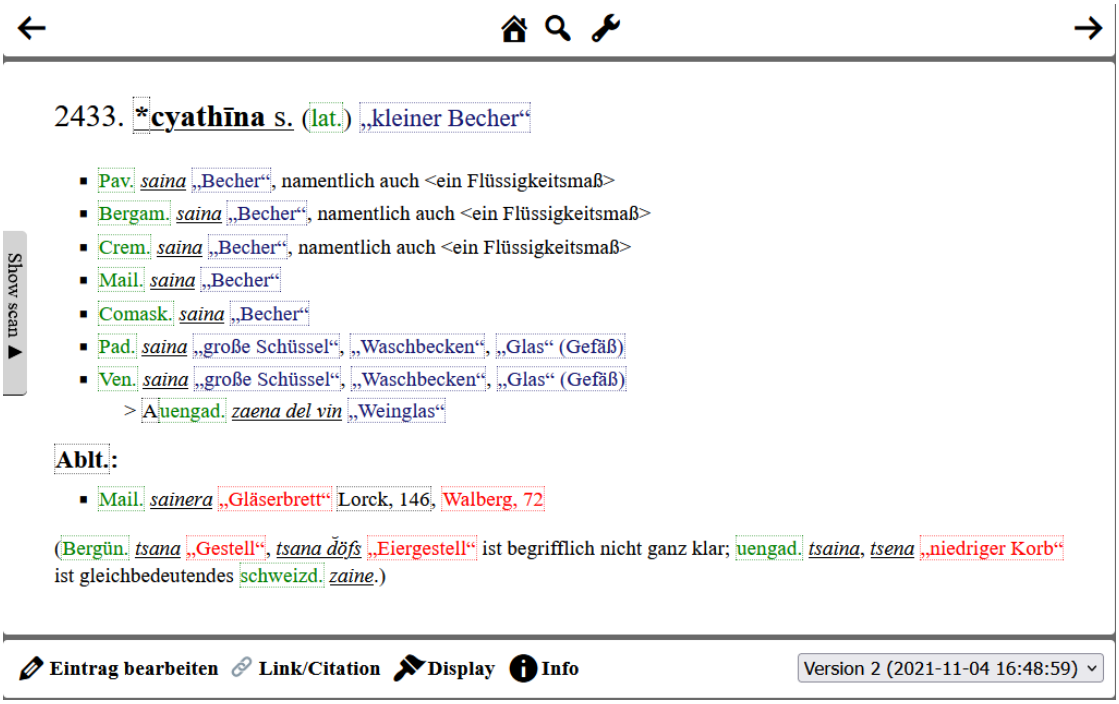


Abbildung 12.10: Standardansicht eines Artikels (Link)

Zusätzlich zu dieser Darstellung kann auch eine originalgetreuere Variante angezeigt werden, die zwar die verschiedenen Markierungen und Anreicherungen der

Standardansicht enthält, aber die Beleglisten in Satzform formatiert. Einzig die Abstände zwischen den Bestandteilen sind hier größer gewählt:

Abbildung 12.11: „Klassische Ansicht“ eines Wörterbuchartikels (Link)

Die Darstellung der Artikels an der Oberfläche wird nicht aus dem Quellentext, sondern aus der internen Repräsentation in der Datenbank generiert, die sehr kleinteilig strukturiert ist. Damit sind grundsätzlich weitere Darstellungsvarianten, z.B. eine tabellarische Ansicht o.ä. denkbar. Die Vorkommen aller zentralen Entitäten im Artikeltext sind entsprechend markiert. Grundsätzlich sind Elemente mit einem gepunkteten Rahmen interaktiv, d.h. beim Überfahren mit der Maus und Klick (bzw. Antippen auf Mobilgeräten) werden zusätzliche Informationen über einen *Tooltip* angezeigt und (falls vorhanden) eine Verlinkung mit externen Quellen bzw. der entsprechenden internen Detailseite (s. Kap. 12.3) angegeben. Alle Formen von Abkürzungen werden beispielsweise im Artikeltext entsprechend ihrer Darstellung in der Quelle angezeigt und bei einer entsprechenden Interaktion aufgelöst:

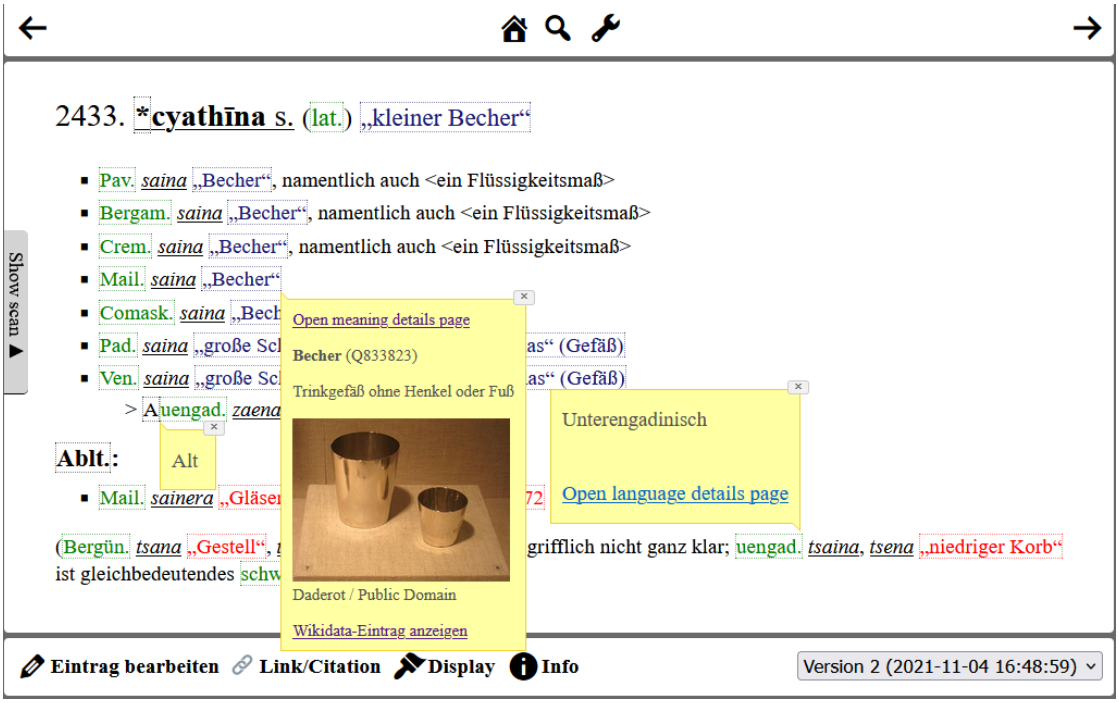


Abbildung 12.12: Verschiedene interaktive Elemente im Artikel

Findet die Verknüpfung mit *Wikidata* nicht über *exact match* statt (vgl. Kap. 10.2.2) werden die verschiedenen *Statements*, die ein Konzept beschreiben, tabellarisch angezeigt. Die folgende Abbildung zeigt dies am Beispiel der Bedeutung „kleiner, runder Käse“:

Abbildung 12.13: Mehrere *Statements* zur Beschreibung eines Konzepts an der Oberfläche

Zusätzlich gibt es bei der Darstellung der eigentlichen Bedeutungstexte an der Oberfläche eine Besonderheit. Da die Herleitung der Bedeutungen (vgl. Kap. 7.2.1) in komplexen Fällen zu falschen Ergebnissen führen kann, werden alle inferierten Formen teiltransparent dargestellt und im *Tooltip* explizit als inferiert ausgewiesen. Somit wird auf eine gewisse Unsicherheit in diesen Fällen hingewiesen. Im folgenden Beispiel sind (im strukturierten Teil) nur zwei explizite Bedeutungsangaben vorhanden. Dementsprechend werden alle weiteren teiltransparent dargestellt.

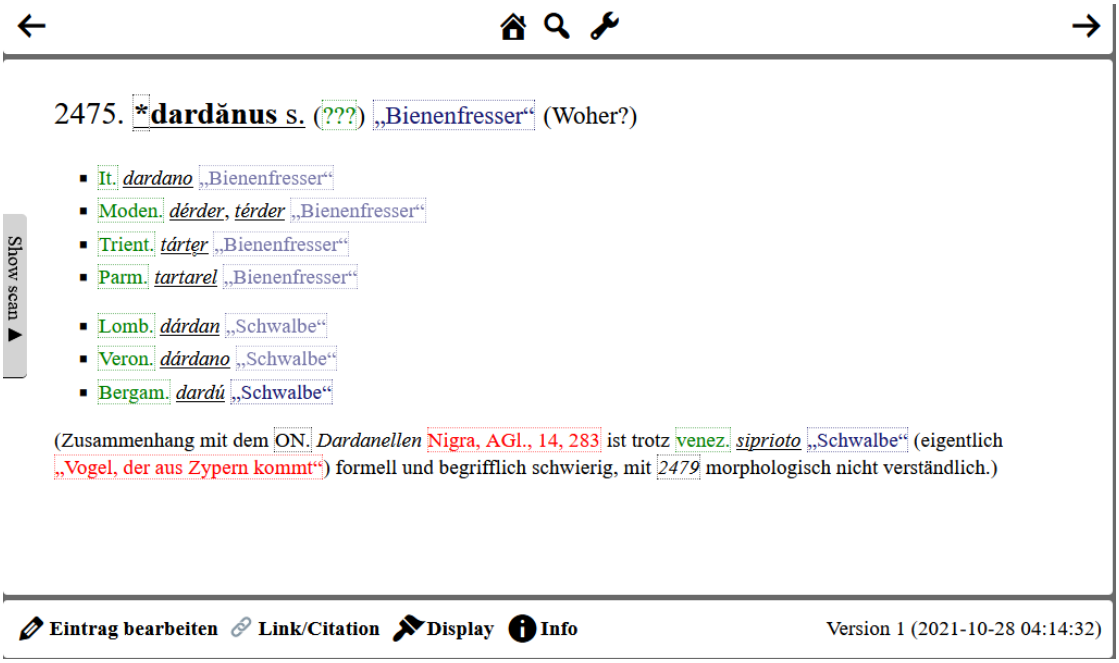


Abbildung 12.14: Spezielle Markierung von inferierten Bedeutungen (Link)

Bei Literaturverweisen wird im *Tooltip* seitengenau auf externe Digitalisierungen des jeweiligen Werks verwiesen, falls diese vorhanden sind und entsprechend verknüpft wurden (vgl. Kap. 11.1):

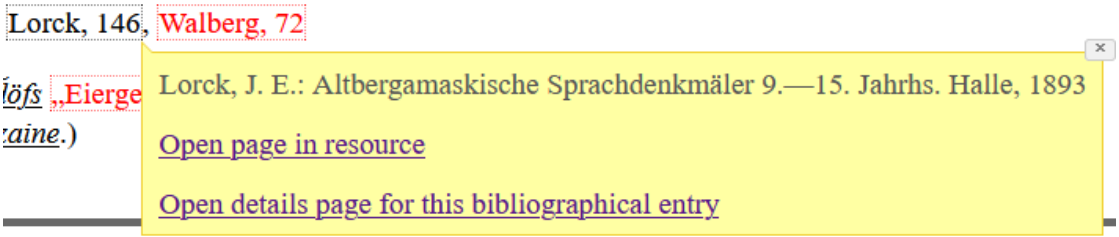


Abbildung 12.15: Auflösung einer bibliographischen Angabe mit externer und interner Verlinkung

Unabhängig von der sonstigen Darstellung des Artikels, kann jederzeit der Ausschnitt aus dem Scan des Originaltexts angezeigt werden (vgl. Abb. 12.16). Dies erleichtert den Abgleich mit dem Quellenmaterial erheblich und eventuelle Fehler bei der Erstellung der Artikeldaten können unkompliziert aufgefunden und behoben (s. Kap. 12.5.1) werden. Sind dem jeweiligen Artikel Nachträge oder Korrekturen aus dem Anhang zugeordnet, werden die entsprechenden Ausschnitte ebenfalls angefügt.

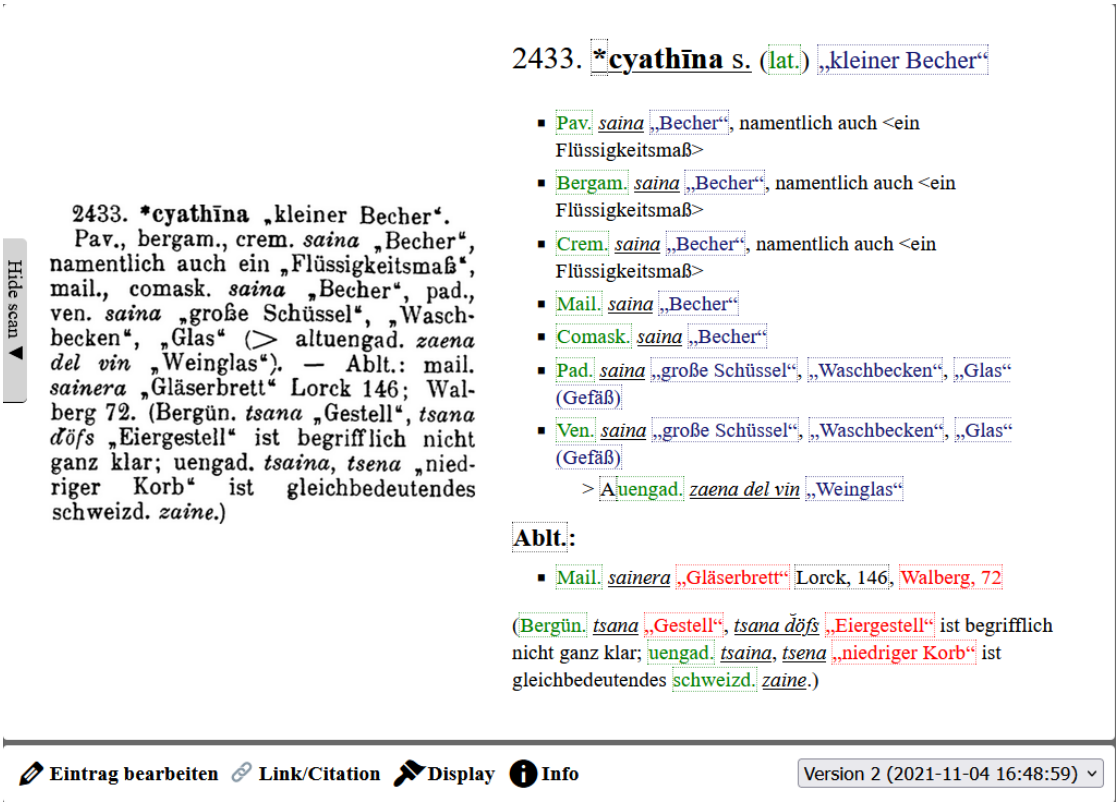


Abbildung 12.16: Eintrag mit Originalscan (Link)

Zusätzlich zu den angereicherten Darstellungen des Wörterbuchartikels, kann dieser auch als reiner Text angezeigt werden, was beispielsweise das Kopieren von Textpassagen erleichtert. Hierbei ist eine Fließtextvariante und eine Darstellung der ursprünglichen Blöcke im Ausgangstext möglich.

Im rechten unteren Eck ist (falls mehrere Versionen existieren, vgl. Kap. 3.4) eine Auswahl aller älteren Artikelversionen in der zum damaligen Zeitpunkt gültigen Fassung möglich. Über das Bedienelement „Link/Zitation“ können zwei verschiedene Formen von URLs erzeugt werden. Für wissenschaftliche Zitationen ist die explizite Verlinkung auf die aktuelle (oder eine ältere) Version des Artikels vorgesehen. Der Inhalt des Artikels bleibt dabei statisch. Für Anwendungsfälle, in denen das nicht notwendig ist, kann auf die jeweils neuste Version des Artikels verwiesen werden. Hierbei wird in der URL nicht eine konkrete ID einer Artikelversion, sondern seine Lemmanummer im Originalwerk verwendet. Befinden sich Nutzende in einer veralteten Artikelversion, wird dies über einen entsprechenden Warnhinweis angegeben.

12.3 Aggregierende Detailseiten

Alle hervorgehobenen Elemente in den Artikeln des vorherigen Kapitels (mit Ausnahme der allgemeinen Abkürzungen) können im Webportal über einen eigene URL angesprochen werden, die auf der ID des jeweiligen Objekts in der Datenbank basiert. Diese URLs können einerseits zur Identifizierung im Kontext des *Semantic Webs* genutzt werden, sie enthalten aber im Gegensatz zu anderen Anwendungen, bei denen oftmals alle URLs zum gleichen Dokument aufgelöst werden (vgl. z.B. lexinfo) konkrete Informationen. An dieser Stelle werden alle Vorkommen des entsprechenden Objekts aufgelistet, der Zustand der Vernetzung mit externen Ressourcen angezeigt und/oder Visualisierungen auf Basis der gesamten Daten zu diesem Objekt erstellt.

Für alle Detailseiten wird zu Beginn eine Liste von Varianten anzeigt, falls solche existieren. Im Fall einer Bedeutung sind dies synonyme Angaben, im Fall von Sprachen oder literarischen Quellen Abkürzungsvariationen und im Fall von sprachlichen Formen weitere Flexionsformen des gleichen Lexems oder orthographisch identische Formen in verschiedenen Dialekten einer übergeordneten Sprache. Die Detailseiten für die jeweiligen Varianten sind inhaltlich identisch, werden allerdings über eine eigene ID angesprochen. Somit bleiben bestehende Links immer gültig auch wenn bestimmte Objekte nachträglich zusammengelegt bzw. getrennt werden. Falls ein Objekt in keinem aktuellen Artikel mehr verwendet wird, beispielsweise im Fall von korrigierten Fehlern, wird ein entsprechender Hinweis und eine Verlinkung zu den veralteten Artikelversionen angeboten, sodass über diesen Umweg auch die jeweils korrigierte Version gefunden werden kann:

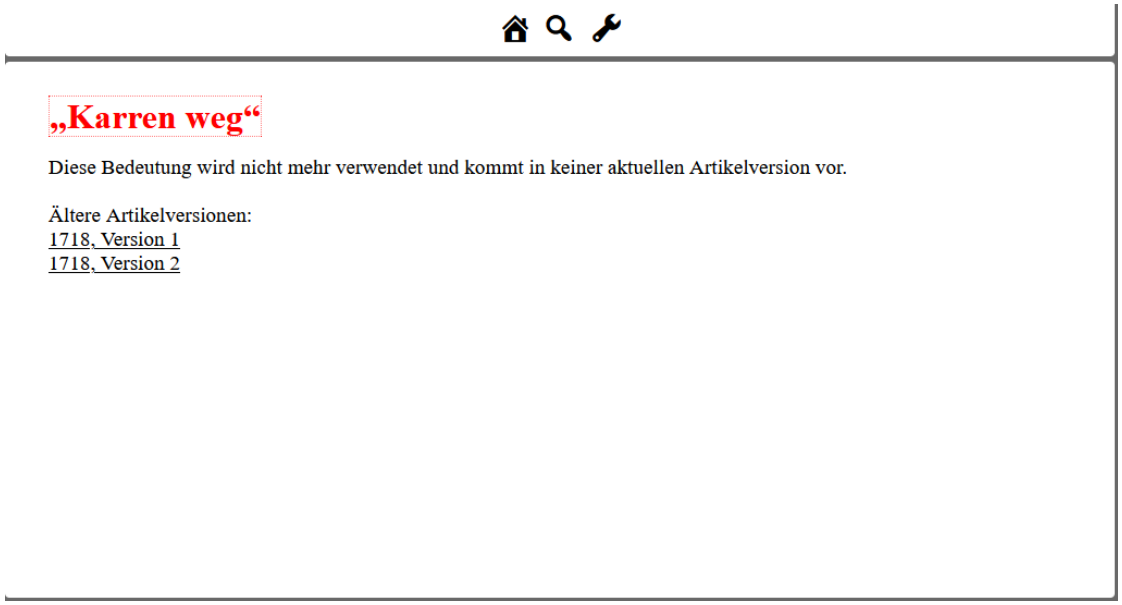


Abbildung 12.17: Detailseite für eine veraltete (weil fehlerhafte) Bedeutung (Link)

Bei aktuellen Objekten werden alle Vorkommen in den jeweiligen Artikeln aufgelistet. Somit findet grundsätzlich einer Verlinkung in beide Richtungen statt (von Artikel zu Einzelelement und zurück). Gleichzeitig hat man so Zugriff auf eine Übersicht, die beispielsweise angibt, welche Formen für eine bestimmte Bedeutung quellenübergreifend vorkommen:

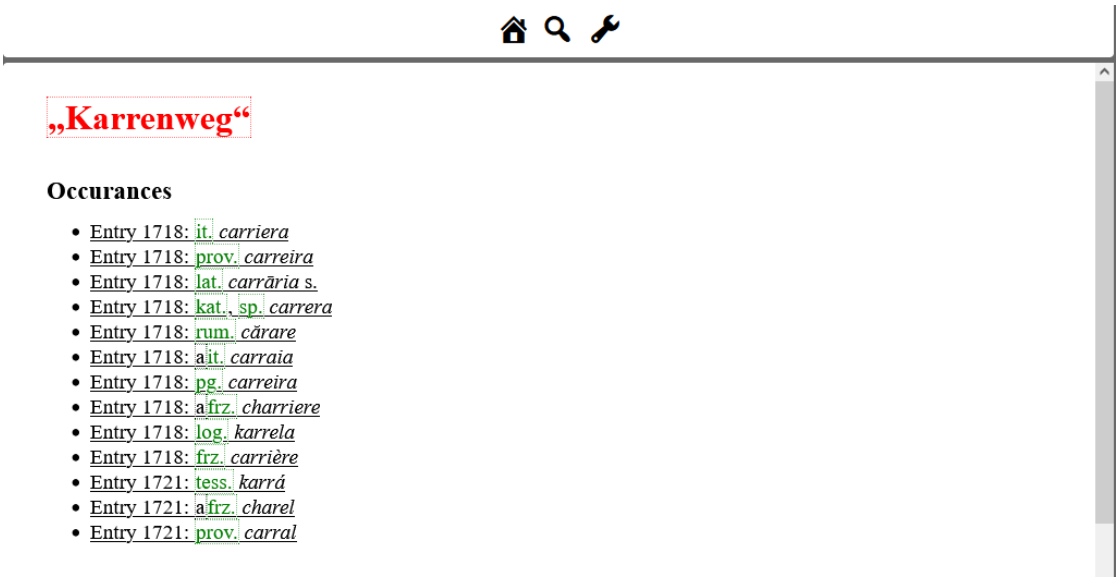
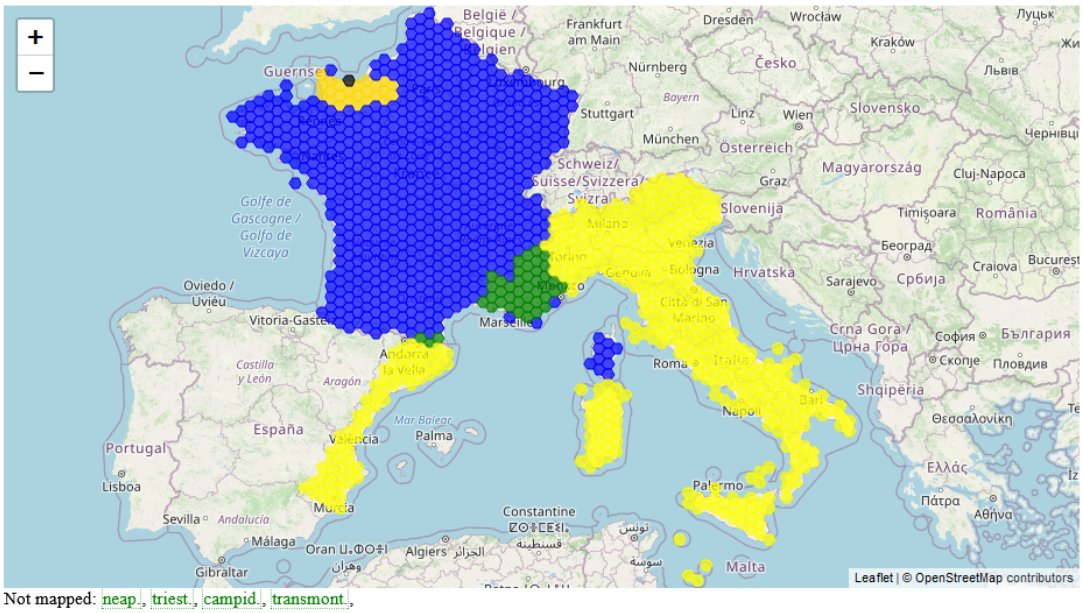


Abbildung 12.18: Die verschiedenen Formen für die Bedeutung „Karrenweg“ (Link)

Im Fall von Formen und Bedeutungen sind weiterhin auf Basis dieser gesammelten Daten verschiedene Formen der Visualisierung denkbar. Im Webportal werden hier drei Varianten prototypisch angelegt. Im Fall von Bedeutungen kann beispielsweise georaphisch abgebildet werden, in welchen Regionen welches Etymon bzw. welche Etyma Grundlage für die dort aktuell verwendeten Formen sind:

Etymon map



Not mapped: neap., triest., campid., transmont.

Legend

- breton. *gwemon s.* / anord. *tang s.*
- breton. *gwemon s.*
- lat. *alga s.* / breton. *gwemon s.*
- lat. *alga s.*
- anord. *tang s.*

Abbildung 12.19: Einfache geographische Visualisierung der Herkunft der Formen für die Bedeutung „Tang“ (Link)

Wenn umgekehrt alle Bedeutungen einer sprachlichen Form betrachtet werden, kann (falls sie in mehreren Regionen vorkommt) eine Kartierung erstellt, werden die die Bedeutungen räumlich darstellt:

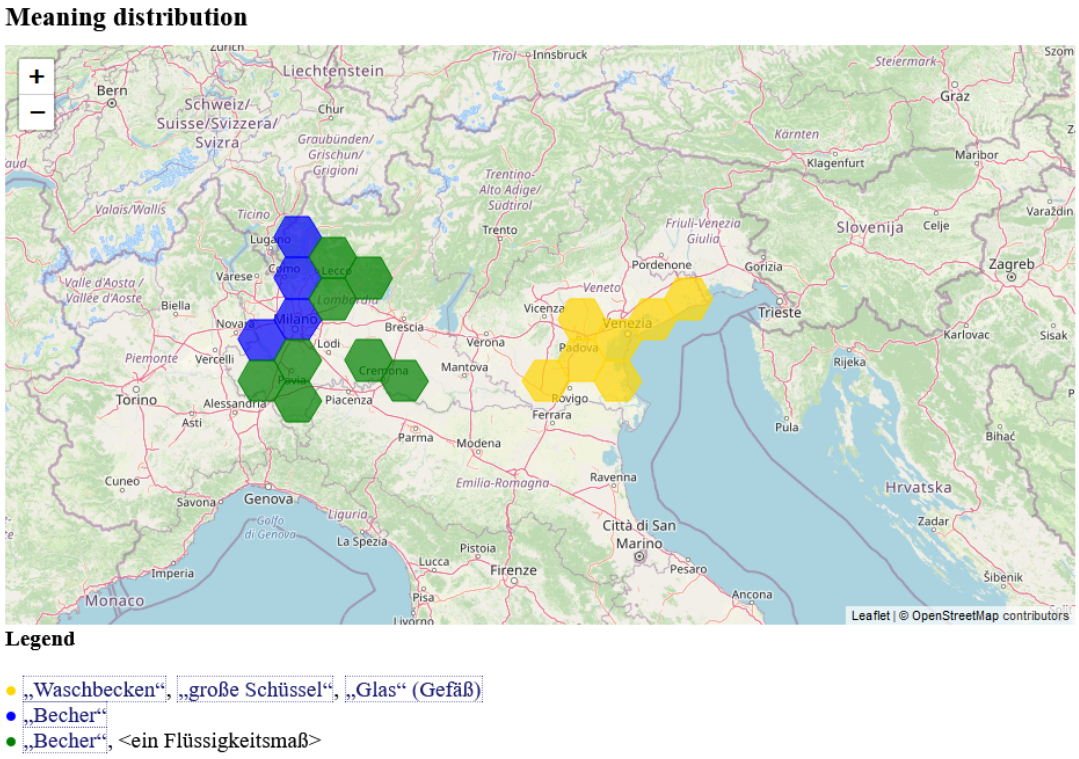


Abbildung 12.20: Verschiedene regional unterschiedliche Bedeutungen einer sprachlichen Form (Link)

Im REW sind solche detaillierte Angaben allerdings die Ausnahme, so dass auf Basis dieser Daten eine solche Visualisierung nur in seltenen Fällen hilfreich ist. Interessanter ist eine visuelle Darstellung der Kerninformation eines etymologischen Wörterbuchs, indem die verschiedenen Entlehnungswege dargestellt werden. Die folgenden beiden Abbildungen zeigen verschiedene Varianten der entsprechenden Graphen. Diese sind auf den Detailseiten der verschiedenen vorkommenden Formen jeweils identisch, wobei die aktuelle Form und ihr spezifischer Herleitungsweg speziell markiert werden.

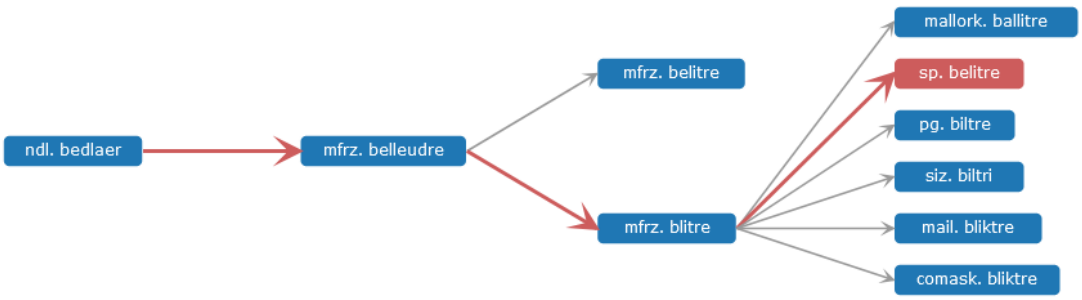


Abbildung 12.21: Herleitungsweg über mehrere Ebenen (Link)

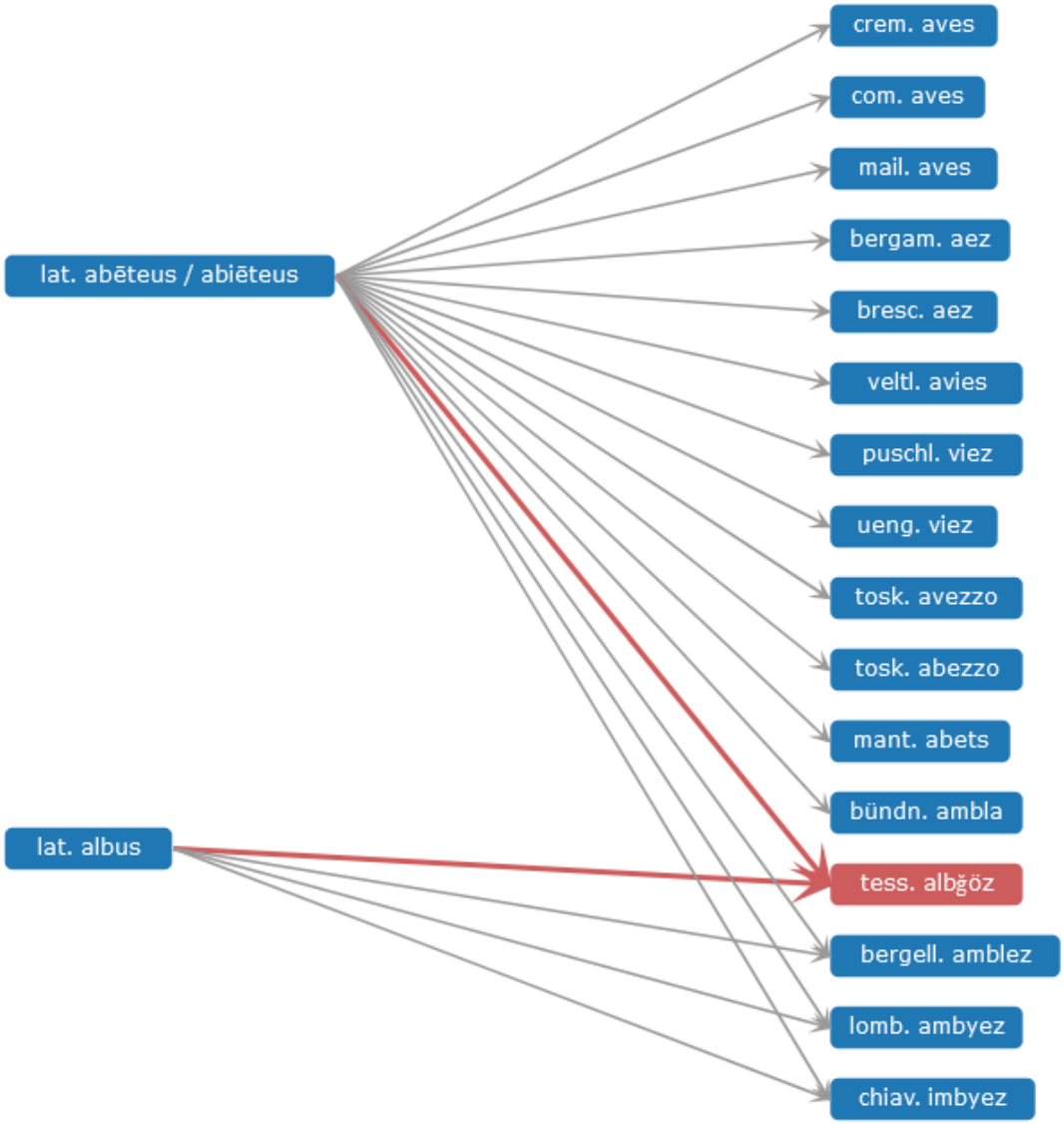


Abbildung 12.22: Etymologischer Graph im Falle von Kontaminationen (Link)

Die Detailseiten zu den Sprachen enthalten hauptsächlich interne und externe Vernetzungsdaten, geben also ihre hierarchische Einordnung und eventuelle geographische Informationen an. Für letzteres werden verschiedene Kombination von Sprachabkürzung und geographischer Angabe unterschieden (beispielsweise „frz.“, „nordfrz.“ etc.). Bibliographische Einträge listen vor allem die Vorkommen von Literaturverweisen in den jeweiligen Artikeln (nach Band gruppiert, falls mehrere vorhanden sind).

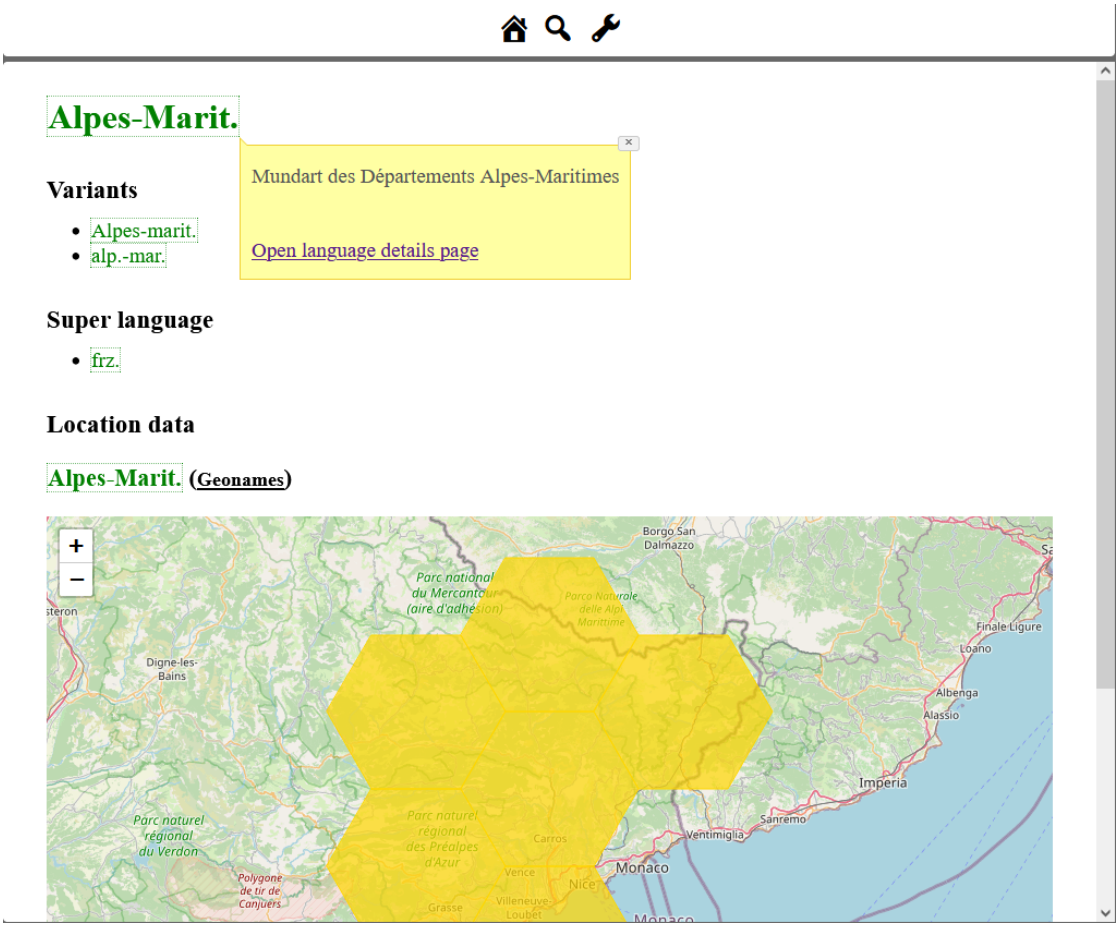


Abbildung 12.23: Sprachdetailseite mit hierarchischer und geographischer Information (Link)

12.4 Statistische Auswertung des Quellenmaterials

Eine weitere Herangehensweise an eine Visualisierung auf Basis des vollständigen Datenmaterials ist die Darstellung von statistischen Auswertungen der Quelle. Somit können bestimmte Merkmale der Quelle quantifiziert werden, die Meta-Analysen des Werks erlauben und somit auch bei der Einschätzung des Datenmaterials hilfreich sind. Auf Basis des Datenbankmodells aus Kapitel 6 sind verschiedenste Abfragen denkbar, exemplarisch wird hier eine Übersicht über die Sprache der jeweiligen Formen und eine quantitative Auswertung der verschiedenen literarischen Quellen gezeigt. Erstere kann Hinweise über die Abdeckung des gesamten romanischen Sprachraums durch das REW geben, während letztere eventuell hilfreich für die Einschätzung der Herkunft der angeführten Informationen ist.

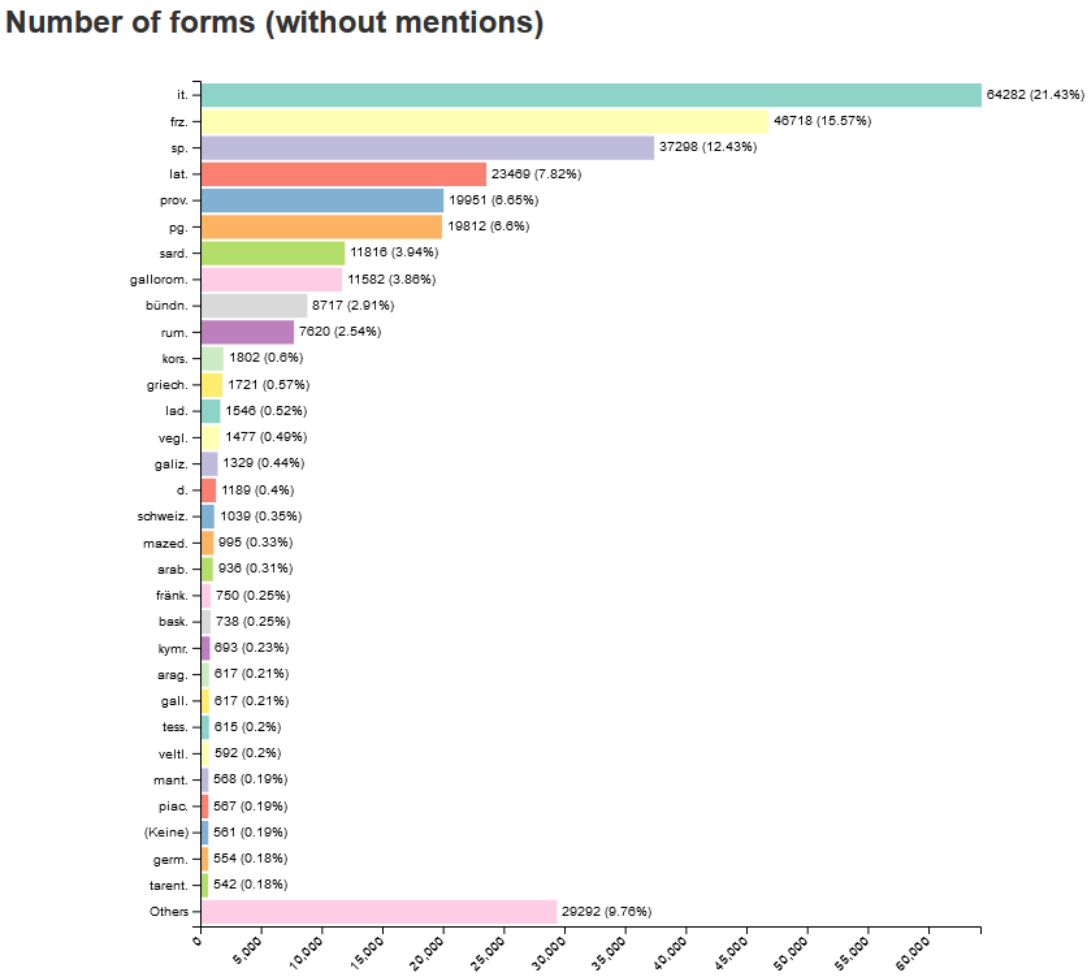


Abbildung 12.24: Anteil der einzelnen Sprachen an der Gesamtheit der Formen. Untergeordnete Dialekte werden mit der übergeordneten Sprache zusammengefasst (Stand 19.06.2022).

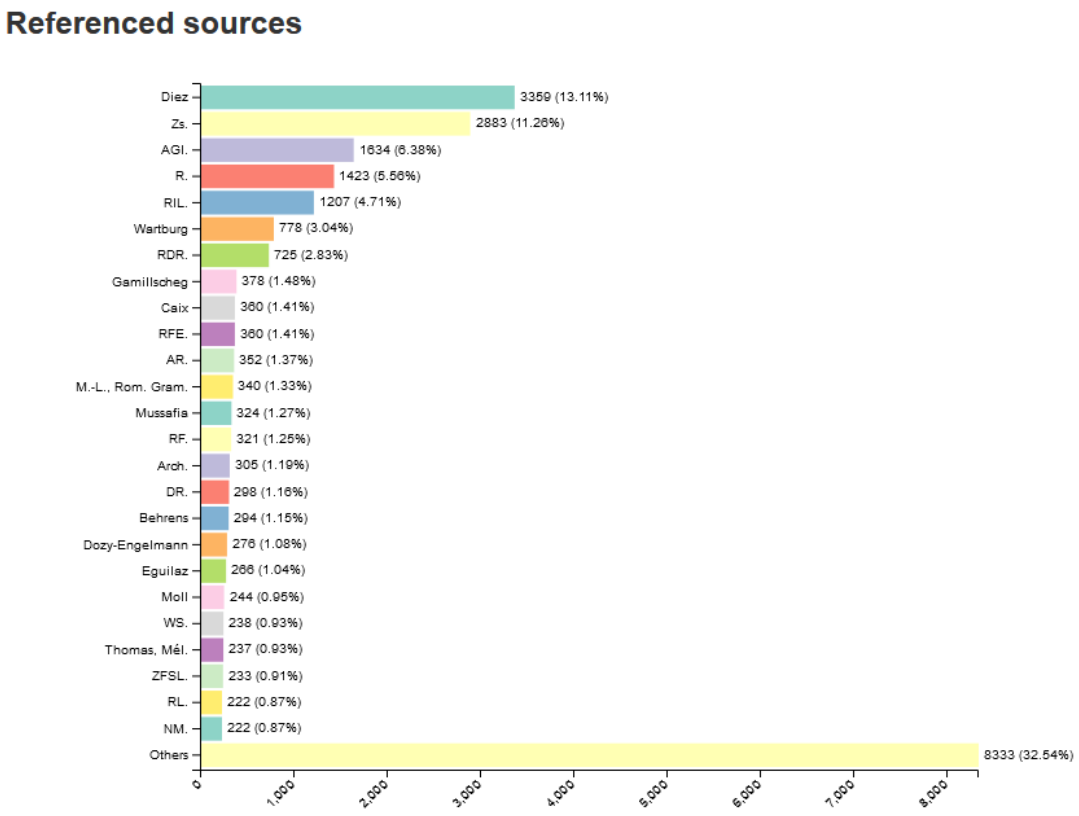


Abbildung 12.25: Anteil verschiedener literarischer Quellen an der Gesamtheit der Literaturverweise (Stand 19.06.2022).
Die jeweils aktuellen Diagramme (und weitere) können unter https://www.rew-online.gwi.uni-muenchen.de/index.php/statistics/ angezeigt werden.

12.5 Interaktionsmöglichkeiten

Ein wichtiger Bestandteil des Webportals ist die Möglichkeit zur Korrektur und Verbesserung der zugrundeliegenden Daten. Hauptsächlich ist hiermit der zentrale Prozess zur Erzeugung der Artikeldaten gemeint, es kann aber auch die Vernetzung mit externen Ressourcen vorgenommen werden. Alle hier vorgestellten Werkzeuge wurde mit der Grundidee konzipiert, dass sie von internen Mitarbeitenden, aber auch (zumindest in reduzierter Form) von Außenstehenden verwendet werden können. Somit ist die Kernidee nicht die eines Portals zur Online-Erhebung (vgl. z.B. Elspaß und Möller 2006), sondern die zusätzliche Möglichkeit zur Interaktion einer tendenziell eher auf lesenden Zugriff ausgelegten Plattform. Alle Interaktionsmöglichkeiten stehen registrierten Nutzenden zur Verfügung. Die grundsätzliche Möglichkeit zur Korrektur von Zeilen im Quellenmaterial wurde bereits in Kap. 8.1 beschrieben, hier wird nur die

Einbettung in den Prozess der Artikelverarbeitung und die zusätzlichen Eingriffmöglichkeiten beschrieben.

12.5.1 Korrekturmöglichkeiten für Nutzende

Im folgenden wird die Bearbeitungsansicht auf einen Artikel besprochen. In dieser kann der Artikel korrigiert, zusätzliche Ausnahmen hinzugefügt, die Erzeugung der einzelnen Teilresultate überprüft und der Artikel neu importiert werden. Sie ist aus den folgenden Bereichen zusammengesetzt:

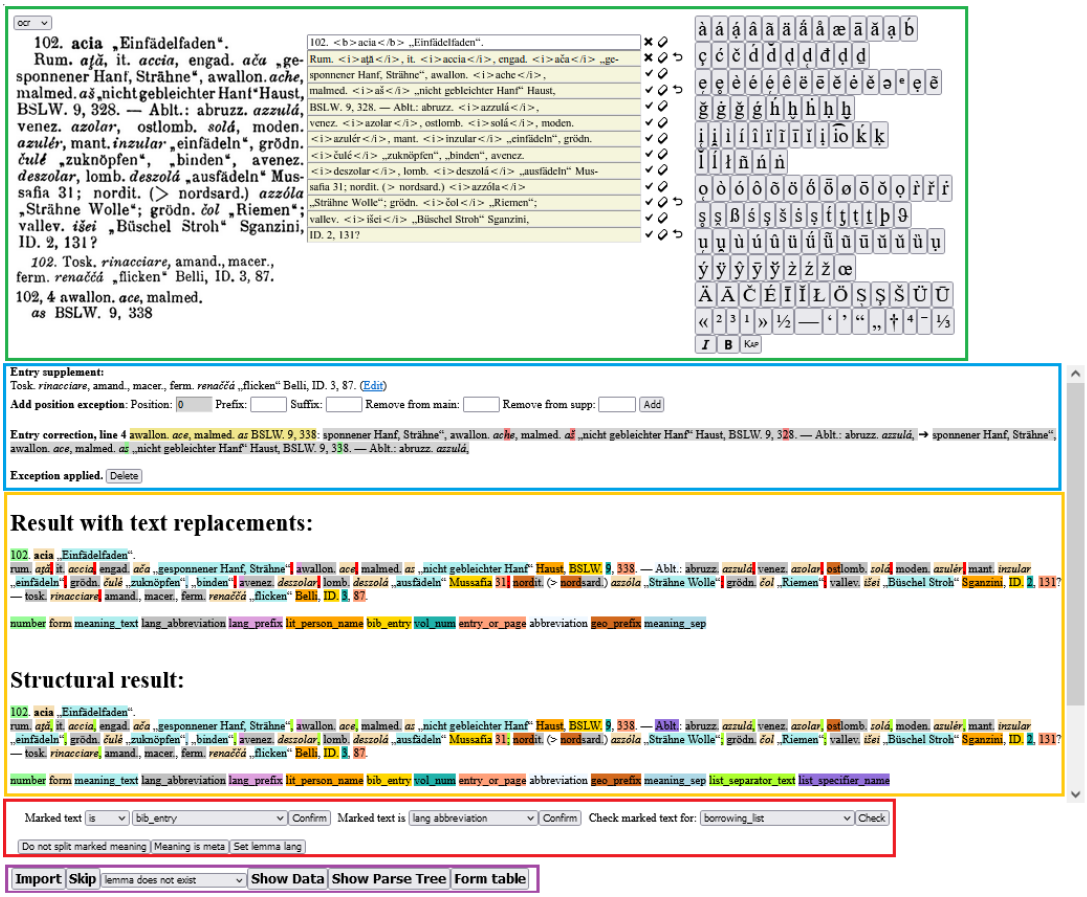


Abbildung 12.26: Verschiedene Elemente der Bearbeitungsansicht eines Artikels im Expertenmodus

Eingabemaske (grün): Diese enthält die Möglichkeit zur Korrektur aus Kap. 8.3 auf Basis der konkreten Zeilen, die einem Artikel zugeordnet sind. Weiterhin werden die Ausschnitte eventueller Zusatzangaben aus dem Anhang angezeigt, können aber an dieser Stelle nicht bearbeitet werden, da sie bei Änderungen gesondert re-importiert werden müssen, bevor sie auf den Artikel angewendet werden können (s. Kap. 8.3).

Zusätzlich können durch die Icons direkt rechts neben den Zeilen Ausnahmen hinzugefügt werden, die auf diesen basieren. In diesem Fall handelt es sich um die Behandlung von Absätzen. Ein solcher wird durch einen Wechsel der Hintergrundfarbe der Zeilen symbolisiert. Durch das -Icon kann angegeben werden, dass eine Zeile keinen neuen Absatz darstellt. Umgekehrt kann mit dem -Icon ein neuer Absatz eingefügt werden.

Anhangdaten und ungültige Formen (blau): An dieser Stelle wird das Einfügen zusätzlicher Angaben aus dem Anhang gesteuert. Für Ergänzungen kann die genaue Position der Einfügung über eine Ausnahme festgelegt (vgl. Kap. 8.3.1) oder diese entfernt werden, falls dies nötig ist. Für Korrekturen wird die aktuelle Anwendung von dieser auf die entsprechende Artikelpassage angegeben, indem Einfügungen und Auslassungen grün bzw. rot markiert werden. Falls die Ersetzungen inkorrekt sind (oder nicht ausgeführt werden können), ist hier ebenfalls die Möglichkeit zum Erstellen (oder Inaktiv-Setzen) einer entsprechenden Ausnahme vorhanden (vgl. Kap. 8.3.2). Zusätzlich werden hier alle fett bzw. kursiv markierten Elemente gelistet, die keine gültigen sprachlichen Formen sind. Um Fehler möglichst aufzufinden, sind hier die Regeln der Grammatik sehr strikt gefasst, z.B. sind keine Großbuchstaben erlaubt. Falls dies in Ausnahmefällen doch vorkommt (z.B. bei Toponymen) können die jeweiligen Elemente hier mit einem Klick über eine Ausnahme als sprachliche Form markiert werden.

Ergebnisse auf struktureller Basis (orange): In diesem Bereich wird die aktuelle strukturierte Verarbeitung des Artikeltexts durch farbliche Markierung der einzelnen erkannten Elemente visualisiert. Dabei werden zwei Verarbeitungsstufen dargestellt, die den beiden Phasen der strukturellen Erkennung (vgl. Kapitel 5) entsprechen. Oben wird das vollständige Resultat mit der Markierung einzelner Bestandteile in diskursiven Passagen angezeigt, während unten das Ergebnis der Erfassung der strukturierten Abschnitte dargestellt wird, in dem alle sonstigen Sätze bzw. Teilsätze rot markiert werden[3]. Vor allem die untere Ansicht ist oft hilfreich, um einzelne Fehler aufzufinden, die die strukturelle Erfassung eines Satzes verhindern (siehe Kap. 12.5.2). Bei jeder Änderung (also bei einer Korrektur der Zeilen oder dem Hinzufügen einer Ausnahme) wird dieser Bereich neu geladen und die aktualisierten Ergebnisse werden angezeigt.

Operationen auf markierten Textabschnitten (rot): Hier können vor allem Ausnahmen auf Grammatikebene erstellt werden. Dazu kann eine Passage aus einer der farblich hinterlegten Darstellungen des Artikeltexts markiert werden und diese als (positive oder negative) Grammatikausnahme zu einer bestimmten Regel verwendet werden (vgl. Kap. 5.2.6). Bestehende Ausnahmen werden ebenfalls hier gelistet und können entsprechend entfernt werden. Weiterhin sind auch bestimmte Ausnahmen für den Importprozess erstellbar, die ebenfalls auf Textausschnitten beruhen. Beispielsweise kann eine markierte Bedeutungsangabe als *Bedeutungsklasse* markiert werden (vgl.

[3] In der obigen Abbildung ist der vollständige Artikel strukturiert erfassbar, sodass sich beide Darstellungen in diesem Fall entsprechen.

Kap. 10.1), falls diese nicht also solche erkannt wurde. Eine weitere Möglichkeit, die ebenfalls auf der Markierung einer Textpassage beruht, ist das Hinzufügen von neuen Sprach- oder bibliographischen Abkürzungen. Wird dies durchgeführt wird eine Liste von ähnlichen bestehenden Abkürzungen der jeweiligen Kategorie angezeigt, sodass neue Varianten eines bestehenden Eintrags diesem zugeordnet werden können. Ist dies nicht der Fall wird ein neuer eigenständiger Eintrag (mit unbekannter Bedeutung) angelegt. Zuletzt können noch einzelne Textpassagen im Bezug auf eine bestimmte Regel der formellen Grammatik untersucht werden. Hierbei wird also ein Teil des Texts mit einer untergeordneten Regel geparst. Dies kann hilfreich bei der Anpassung der Grammatik und der Behebung von schlecht formulierten Regeln sein.

Steuerungselemente (lila): Dieser letzte Bereich enthält verschiedene Bedienelemente. Mit dem Button *Import* wird eine neue Artikelversion erstellt (falls die erzeugten Daten sich geändert haben) und im Anschluss zur Leseansicht weitergeleitet. Mit dem folgenden Button kann der Artikel (unter Angabe eines Grundes) übersprungen werden. Dies ist nur in der initialen Verarbeitungsphase sinnvoll und im Gegensatz zu allen weiteren Elementen nicht für Außenstehende sichtbar (vgl. auch Kap. 5.2.3). Die folgenden beiden Buttons dienen zur Anzeige der detaillierten Ergebnisse der beiden Hauptverarbeitungsssschritte, ohne die Speicherung der entsprechenden Daten. Mit ersterem kann der vollständige Strukturbaum in dynamisch aufklappbarer Art angezeigt werden, während zweiterer alle Datenzeilen in aufbereiteter Form anzeigt:

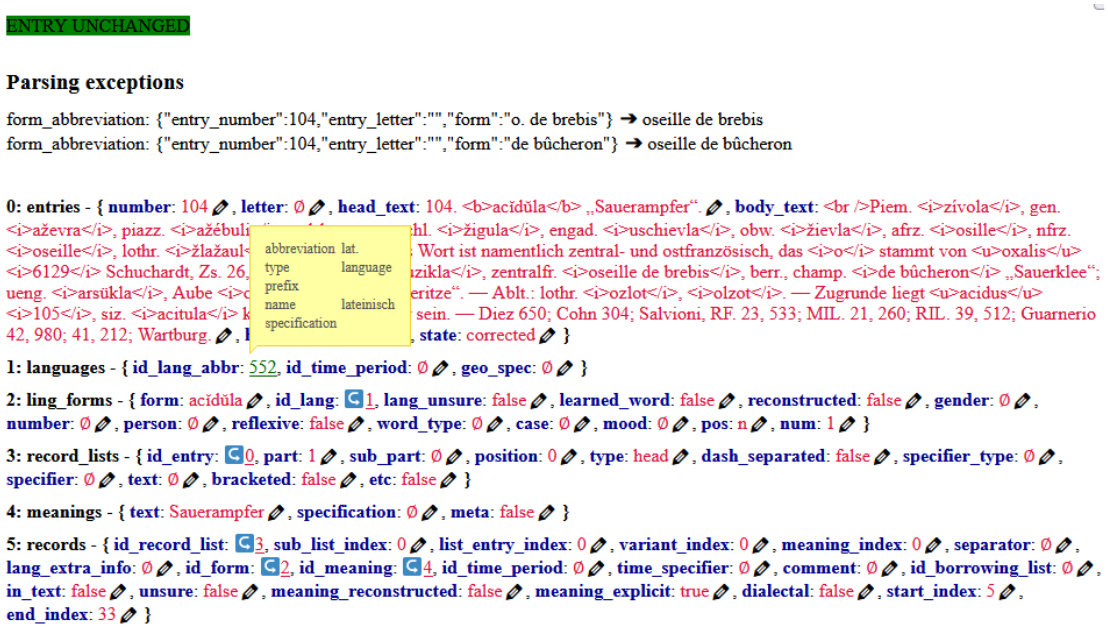


Abbildung 12.27: Ausschnitt aus der Anzeige der Resultatdaten

Zu Beginn wird der Status der Daten angegeben. Falls es Änderungen zur aktuellen Artikelversion gibt, werden diese aufgelistet (vgl. Kap. 3.4.1). Es folgt eine Liste aller

Prozessausnahmen, die bei der Erstellung der Daten angewendet werden. Der Hauptteil besteht dann aus den einzelnen Datenzeilen, die für diesen Artikel erzeugt wurden. Referenzen auf andere Zeilen werden entsprechend verlinkt und mit einem Pfeil markiert. Verweise auf bestehende Elemente in der Datenbank (im Beispiel die Sprachabkürzung) werden über einen *Tooltip* aufgelöst. Zusätzlich können an dieser Stelle Ausnahmen vom Typ *replace_text* (vgl. Kap. 7.3) erstellt werden, um einzelne Datenzeilen anzupassen. Dazu dient das Stift-Icon nach allen konstanten Elementen (d.h. Einträgen, die kein Verweis sind).

Der letzte Button erlaubt die Erstellung von lokalen Ausnahmen, die auf den Artikelkontext bezogen sind. Hier wird eine Tabelle aller Sprachbelege und ihrer Indizierung und der Schreibweise der zugehörigen Form angegeben:

part	sub_part	position	sub_list_index	in_text	list_entry_index	form
1		0	0	0	0	acĭdŭla
1		1	0	0	0	zívola
1		1	0	0	1	ažévra
1		1	0	0	2	ažébuli
1		1	0	0	3	žigula
1		1	0	0	4	žigula
1		1	0	0	5	uschievla
1		1	0	0	6	žievla
1		1	0	0	7	osille
1		1	0	0	8	oseille
1		1	0	0	9	žlažaul
1		3	0	0	0	uzikla
1		3	0	0	1	oseille de brebis
1		3	0	0	2	de bûcheron (oseille de bûcheron)
1		3	0	0	3	de bûcheron (oseille de bûcheron)
1		3	1	0	0	arsükla
1		3	1	0	1	o. de brebis (oseille de brebis)
1		4	0	0	0	ozlot
1		4	0	0	0	olzot
1		5	0	1	0	acitula

Exception type: Meaning exception

Meaning: Test Specification: Meta: +


```
▾Object
  context: "get_meaning"
  ▾value_in: Object
    entry_number: 104
    entry_letter: ""
    part: "1"
    sub_part: ""
    position: "3"
  ▾value_out: Array[1]
    ▾0: Object
      text: "Test"
      specification: ""
      meta: "0"
```


Add exception

Abbildung 12.28: Eingabemaske zur Erstellung von lokalen Prozessausnahmen

Grundsätzlich kann entweder die Schreibweise einer Form für die Definition einer Ausnahme verwendet werden, sodass sich diese auf alle orthographisch identischen

Formen im aktuellen Artikel bezieht, oder ein Kontext über die Indizierung der jeweiligen Sprachbelege gegeben werden. Es ist allerdings auch eine Kombination aus beidem möglich ist, falls ein Sprachbeleg mehrere Varianten hat und sich die Ausnahme nur auf eine von diesen bezieht. Im Beispiel in der obigen Abbildung würde die Ausnahme auf alle Sprachbelege in der Liste mit der Position 3 angewendet.

Bestimme Ausnahmefälle müssen auf struktureller und Verarbeitungsebene definiert werden. Das ist vor allem der Fall bei abkürzenden Schreibweisen, die nicht strukturell als solche erkannt werden. Im vorliegenden Fall gilt dies für die abgekürzte sprachliche Form *de bûcheron*, die für *oseille de bûcheron* steht. Diese werden erst (wie oben beschrieben) über eine Ausnahme auf Grammatikebene markiert und müssen während der Verarbeitung noch aufgelöst werden[4]. Hierzu wird beim Anstoß des Imports bzw. beim Anzeigen der Resultatdaten eine spezielle Meldung angezeigt, die die Eingabe des vollständigen Texts erlaubt:

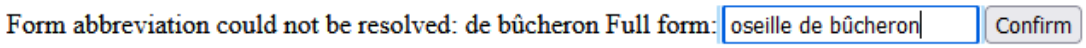


Abbildung 12.29: Eingabemöglichkeit einer Ausnahme für die Behandlung einer Ausnahme

Die Beschreibung aller Elemente hier beruht auf dem Expertenmodus, der alle Eingriffsmöglichkeiten bei der Artikelverarbeitung erlaubt. Standardmäßig ist eine vereinfachte Bearbeitungssicht aktiv, die nur die Korrektur der Zeilen bzw. Absätze ermöglicht.

12.5.2 Beispiele für die Korrektur von Artikeln

Ohne zusätzliche Hilfsmittel ist bei komplexen, grammatikbasierten Parsern das Auffinden der Gründe, warum bestimmte Texte nicht verarbeitet werden können, zum Teil extrem aufwendig. Dieser Abschnitt zeigt anhand eines ausführlichen Beispiels, wie die Bearbeitungsoberfläche dieses erleichtert. Als Beispiel dient hier REW, S. 9295, der aus einem längeren sehr strukturierten Abschnitt und einem natürlichsprachigen Text (in Klammern) besteht. Aufgrund von Fehlern im Ausgangstext können hier allerdings weite Teile nicht strukturell erfasst werden. Diese werden in der farbigen Ansicht des Artikeltexts rot markiert:

9295 . **vĭa** „ Weg “. it. *via* , log. *bia* , engad. *via* , friaul. *vie* , frz. *voie* , prov. , kat. , sp. , pg. *via* ; it. *via* „ Mal “ , *tre via quattro* „ dreimal vier “ , *vieppiù* „ weit mehr “ Spitzer , Zs. 40 , 421 ; friaul. *vie* „ Art und Weise “ ; siz. *ya* , bergam. *bya* „ vorwärts “ , mail. *via* , friaul. *ğa* „ Zuruf an die Pferde “. Ablt.: it. *viale* „Allee“, bergün. *vial* „unterirdischer Gang“; frz. *voyage*, prov. *viatge* (> it. *viaggio*, log. *biadzu*, sp.

[4] Eine Ausnahme stellen abkürzende Schreibweisen mit Bindestrichen bei Formen dar, die größtenteils automatisch behandelt werden können, vgl. Kap. 7.1.3.

viaje, pg. *viagem*) „Reise“; piazz. *a li viaği* „bisweilen“, wallis. *yadyo* „der Weg, der zum Holen des Heus zurückzulegen ist“, „Heubürde, die in einem Male getragen wird“, „Last“, „Mal“, kat. *viatje* „Mal“, it. *viaggiare*, frz. *voyager*, prov. *viatjjar*, pg. *viajar*, reisen“; impt. piver. *viaga* „rasch“. Zssg.: abruzz. *ebbi*, teram. *abbi*, log. *ebbia* „nur“, ursprünglich wohl „und weiter“; siz. *a ḍḍa via*, puschl. *lavía*, bellinz. *lala via* „dort“, puschl. *klavafora* „dort draußen“, *klavaínt* „dort drinnen Salvioni, RIL. 39, 612; it. *diviato* „in gerader Richtung“, „schnell“, kors. *imbia* „nützlich Guarnerio, AGl. 14, 162; asen. in *issa via*, aumbr. *essavia*, numbr. *savia* „sofort Salvioni, R. 39, 445; neuenb. *la vi* „ weg “ ; it. *avviare* , a frz. *avoier* „ auf den Weg bringen “ , sp. , pg. *aviar* „ die Reise vorbereiten “ , berg. *abyá* „ helfen “ , abruzz. *abbiyá* , march. *biáre* „ anfangen “ Salvioni , RDR. 4 , 100 , a tosk. *malabbiato* „ Übeltäter “ Flechia , RFICl. 1 , 378 ; rum. *îmbia* „ auffordern “ , „ einladen “ , eigentlich „ auf den Weg bringen “ , piver. *anaviar* „ auf den Weg bringen “ , *nave* „ Anstoß “ , *snavya* „ vorwärtsbewegen “ , *snav-te* , *snau-te* „ vorwärts “ Flechia , AGl. 18 , 277 , siz. *abbiari* „ wegjagen “ , march. *biare* , *miare* „ anfangen “ , a log. *inviare* „ ausstatten “ , *invíu* „ Ausstattung eines jungen Ehepaares “ ; frz. *envoyer* „ auf den Weg bringen “ , „ schicken “ , sp. , pg. *enviar* „ schicken “ , a frz. *desvoyer* „ vom Wege abbringen “ M. -L. Rom. Gram. 2, 189; Paris, R. 31, 148; it. *crocevia* „ Kreuzweg “ ; a frz. *toutes voies* , sp. , pg. *tcdavía* , salm. *entavia* , *entadia* , *entá* „ dennoch “ ; triest. *a la mata via* „ aufs Geratewohl “ Subak , ATriest. 30 , 161 . — Diez 341 ; Gauchat , Arch. 121 , 446 ; Hochuli 59 . (Bellinz. *üyoüya* „Zuruf an die Pferde gehört kaum hierher, ist vielmehr eine Erweiterung des internationalen *hü*; frz. *voyage* VIATICUM „Reisekost Diez 341 paßt begrifflich nicht; piver. *anviarase* „sich auf den Weg machen“, *a viará* „schnell Flechia, AGl. 18, 325 ist morphologisch nicht klar; triest. *a la mata via* zu sloven. *motovilo*, Strekelj, ASPh. 26, 423 kommt nicht in Betracht; rum. *învia* INVITARE Puşcariu, DR. 4, 1319 ist nicht möglich.) number form meaning_text lang_abbreviation form_sep lit_person_name bib_entry vol_num entry_or_page abbreviation record_sep meaning_sep lang_prefix grammar_spec latin_form list_separator_text bracketed_text

Um Probleme genauer eingrenzen zu können werden dabei für alle Sätze, die nicht vollständig strukturiert erfasst werden können, die einzelnen Satzbestandteile (vgl. hierzu Kap. 5.3.3) nochmals einzeln aufgeführt, so dass einfacher erkannt werden kann, an welchen Stellen der Text fehlerhaft ist bzw. die Importroutine zusätzliche Informationen benötigt. Vollständig erfasste Teilsätze werden entsprechend der erkannten Entitäten eingefärbt, während bei ungültigen Passagen markiert wird, bis zu welcher Stelle der Parser vorgedrungen ist, bevor keine weitere Regel mehr angewandt werden konnte:

Parts

— Ablt. : it. *viale* „ Allee “ , bergün. *vial* „ unterirdischer Gang “. frz. *voyage* , prov. *viatge* (> it. *viaggio* , log. *biadzu* , sp. *viaje* , pg. *viagem*) „ Reise “.
String not valid: „ Piazz. *a li viağı* „bisweilen“, wallis. *yadyo* „der Weg, der zum Holen des Heus zurückzulegen ist“, „Heubürde, die in einem Male getragen wird“, „Last“, „Mal“, kat. *viatje* „Mal“, it. *viaggiare*, frz. *voyager*, prov. *viatjjar*, pg. *viajar*, r eisen“; impt. piver. *viaga* „rasch“.“ for grammar „extra_list“
— Zssg. : abruzz. *ebbi* , teram. *abbi* , log. *ebbia* „ nur “ , ursprünglich wohl „ und weiter “. siz. *a ḍḍa via* , puschl. *lavía* , bellinz. *lala via* „ dort “ , puschl. *klavafora* „ dort draußen “ , *klavaínt* „ dort drinnen “ Salvioni , RIL. 39 , 612 .
String not valid: „ It. *diviato* „in gerader Richtung“, „schnell“, kors. *ímbia* „nützlich Guarnerio, AGl. 14, 162; asen. i n *issa via*, aumbr. *essavia*, numbr. *savia* „sofort“ Salvioni, R. 39, 445.“ for grammar „extra_list“
String not valid: „ - L.“ for grammar „extra_list“
String not valid: „ Rom. Gram . 2, 189; Paris, R. 31, 148.“ for grammar „extra_list“

In diesem Fall sind drei Probleme vorhanden:

- Vor der Bedeutung „reisen“ wurde anstatt eines öffnenden Anführungszeichen ein Komma erkannt.
- Das erste Token von *in issa via* wurde nicht als kursiviert markiert.
- Nach der Abkürzung M.-L. (= Meyer-Lübke) fehlt ein Komma, so dass sie nicht korrekt als Personenname vor einer Literaturangabe erkannt werden kann. Dies führt ebenfalls dazu, dass die beiden Punkte innerhalb der Abkürzungen als Satzende interpretiert werden, was zu der inkorrekten Aufteilung der Sätze führt.

In diesem Fall handelt es bei allen drei Problemen um Zeichen, die im Zuge der Texterkennung nicht korrekt gelesen wurde, sodass sie sehr einfach über die jeweiligen Textfelder korrigiert werden können, was zu folgender Darstellung des Artikels führt:

9295 . **vĭa** „ Weg “. it. *via* , log. *bia* , engad. *via* , friaul. *vie* , frz. *voie* , prov. , kat. , sp. , pg. *via* ; it. *via* „ Mal “ , *tre via quattro* „ dreimal vier “ , *vieppiù* „ weit mehr “ Spitzer , Zs. 40 , 421 ; friaul. *vie* „ Art und Weise “ ; siz. *ya* , bergam. *bya* „ vorwärts “ , mail. *via* , friaul. *ğa* „ Zuruf an die Pferde “. — Ablt. : it. *viale* „ Allee “ , bergün. *vial* „ unterirdischer Gang “ ; frz. *voyage* , prov. *viatge* (> it. *viaggio* , log. *biadzu* , sp. *viaje* , pg. *viagem*) „ Reise “ ; piazz. *a li viağı* „ bisweilen “ , wallis. *yadyo* „ der Weg, der zum Holen des Heus zurückzulegen ist “ , „ Heubürde, die in einem Male getragen wird “ , „ Last

“ , „ Mal “ , kat. *viatje* „ Mal “ , it. *viaggiare* , frz. *voyager* , prov. *viatjjar* , pg. *viajar* „ reisen “ ; impt. piver. *viaga* „ rasch “. — Zssg. : abruzz. *ebbi* , teram. *abbi* , log. *ebbia* „ nur “ , ursprünglich wohl „ und weiter “ ; siz. *a ḍḍa via* , puschl. *lavía* , bellinz. *lala via* „ dort “ , puschl. *klavafora* „ dort draußen “ , *klavaínt* „ dort drinnen “ Salvioni , RIL. 39 , 612 ; it. *diviato* „ in gerader Richtung “ , „ schnell “ , kors. *imbia* „ nützlich “ Guarnerio , AGl. 14 , 162 ; a sen. *in issa via* , a umbr. *essavia* , n umbr. *savia* „ sofort “ Salvioni , R. 39 , 445 ; neuenb. *la vi* „ weg “ ; it. *avviare* , a frz. *avoier* „ auf den Weg bringen “ , sp. , pg. *aviar* „ die Reise vorbereiten “ , berg. *abyá* „ helfen “ , abruzz. *abbiyá* , march. *biáre* „ anfangen “ Salvioni , RDR. 4 , 100 , a tosk. *malabbiato* „ Übeltäter “ Flechia , RFICl. 1 , 378 ; rum. *îmbia* „ auffordern “ , „ einladen “ , eigentlich „ auf den Weg bringen “ , piver. *anaviar* „ auf den Weg bringen “ , *nave* „ Anstoß “ , *snavya* „ vorwärtsbewegen “ , *snav-te* , *snau-te* „ vorwärts “ Flechia , AGl. 18 , 277 , siz. *abbiari* „ wegjagen “ , march. *biare* , *miare* „ anfangen “ , a log. *inviare* „ ausstatten “ , *invíu* „ Ausstattung eines jungen Ehepaares “ ; frz. *envoyer* „ auf den Weg bringen “ , „ schicken “ , sp. , pg. *enviar* „ schicken “ , a frz. *desvoyer* „ vom Wege abbringen “ M.-L., Rom. Gram. 2 , 189 ; Paris , R. 31 , 148 ; it. *crocevia* „ Kreuzweg “ ; a frz. *toutes voies* , sp. , pg. *todavía* , salm. *entavia* , *entadia* , *entá* „ dennoch “ ; triest. *a la mata via* „ aufs Geratewohl “ Subak , ATriest. 30 , 161 . — Diez 341 ; Gauchat , Arch. 121 , 446 ; Hochuli 59 . (Bellinz. *üyoüya* „Zuruf an die Pferde gehört kaum hierher, ist vielmehr eine Erweiterung des internationalen *hü*; frz. *voyage* VIATICUM „Reisekost Diez 341 paßt begrifflich nicht; piver. *anviarase* „sich auf den Weg machen“, *a viará* „schnell Flechia, AGl. 18, 325 ist morphologisch nicht klar; triest. *a la mata via* zu sloven. *motovilo*, Strekelj, ASPh. 26, 423 kommt nicht in Betracht; rum. *învia* INVITARE Puşcariu, DR. 4, 1319 ist nicht möglich.) number form meaning_text lang_abbreviation form_sep lit_person_name bib_entry vol_num entry_or_page abbreviation record_sep meaning_sep lang_prefix latin_form list_separator_text list_specifier_name bracketed_text

Die Anzeige der nicht strukturell erfassbaren Abschnitte (zusammen mit der Erkennung der einzelnen Sätze bzw. Subsätze, vgl. Kap. 5.3.3) führt auch im Falle tatsächlicher unstrukturierter Elemente meist zu einer leichten Erkennung. Im folgenden Beispiel wird auf den ersten Blick ersichtlich, dass die drei nicht erkannten Sätze tatsächlich diskursvie Elemente sind:

4559 . ***īva** (gall.) „ Günsel “. it. *iva* , frz. *ive* , prov. , sp. , pg. *iva* . — Ablt. : frz. *ivette* Rolland 175 . Da der wissenschaftliche Name z. T.

ajuga iva ist, kann sp., pg. *iva* durch Botaniker importiert sein. Das gall. **iva* dürfte identisch sein mit IVUS *4560*, vgl. den griech. Ausdruck *chamaipitys*, der als *pin terrestre* in das Frz. übersetzt ist. Ob die begriffliche und formelle Differenzierung der zwei Wörter von Galliern oder Romanen vorgenommen ist, läßt sich vorläufig nicht sagen.

number form lang_abbreviation meaning_text abbreviation bib_entry entry_or_page latin_form list_separator_text list_specifier_name

Parts

String not valid: „ D a der wissenschaftliche Name z. T. *ajuga iva* ist, kann sp., pg. *iva* durch Botaniker importiert sein.“ for grammar „extra_list“

String not valid: „ D as gall. **iva* dürfte identisch sein mit IVUS *4560*, vgl. den griech. Ausdruck *chamaipitys*, der als *pin terrestre* in das Frz. übersetzt ist.“ for grammar „extra_list“

String not valid: „ O b die begriffliche und formelle Differenzierung der zwei Wörter von Galliern oder Romanen vorgenommen ist, läßt sich vorläufig nicht sagen.“ for grammar „extra_list“

Ein weiterer häufiger Fehlerfall ist das Auftreten von unbekannten Varianten von einleitenden oder trennenden Elementen. In diesem Fall wird eine Liste mit der Formulierung „Mit anderem Ausgang“ eingeführt:

345 a . ***alīsia** „ Elsbeere “. frz. *alise* , saintong. *alie* , dauph. *arie* , langued. *alio* , toul. *aligo* , gask. *alige* , sp. *aliso* , germ. *aliza* , vgl. d. *Elsbeere* . Mit anderm Ausgang: poitev. *alü*, bourn. *olüz*, tann. *harlüs*; ost frz. *alu* , schweiz. *alutzo* ; oengad. *(a)lossa* , uengad. *alaussa* . — Ablt. : frz. *alisier* „ Elsbeerbaum “ , engad. *alosser* ; santand. *alisunas* „ Elsbeeren “. Ursprung und Suffixbildung sind unbekannt. Die *-s*-losen Formen reimen z. T. mit den Vertretern von CAMISIA, während keine nur mit *-isa* vereinbar sind. Das germ. Wort ist aus geographischen Gründen nicht die Grundlage des galloromanischen. Die Bedeutung „Erle scheint in sp. *aliso* vorzuliegen. — Bertoldi , ZCP. 17 , 84 ; Diez 420 ; Wartburg ; Gamillscheg . (Kalabr. *autsinu*, bask. *(s)altza* „Erle Schuchardt, Zs. Bhft. 6, 36 sind jenes geographisch, dieses formell fern zu halten.) number lemma_num_letter form meaning_text lang_abbreviation abbreviation record_sep geo_prefix latin_form lit_person_name bib_entry vol_num entry_or_page list_separator_text extra_info list_specifier_name bracketed_text **Parts**String not valid: „ M it anderm Ausgang: poitev. *alüž*, bourn. *olüz*, tann. *harlüs*.“ for grammar „extra_list“String not valid: „ U rsprung und Suffixbildung sind unbekannt.“ for grammar „extra_list“String not valid: „ D ie *-s*-losen Formen reimen z. T. mit den Vertretern von camisia, während keine nur mit *-isa* vereinbar sind. Das germ. Wort ist aus geographischen Gründen nicht die Grundlage des galloromanischen.“ for grammar „extra_list“String not valid: „ D ie Bedeutung „Erle“ scheint in sp. *aliso* vorzuliegen.“ for grammar „extra_list“

Über eine Markierung der entsprechenden Passage kann wie im vorherigen Kapitel beschrieben eine Ausnahme (in diesem Fall für die Regel *list_specifier_literal*) erstellt werden, sodass diese Passage auch als Beleg(teil)liste erkannt wird. Zum Einfügen solcher Ausnahmen ist allerdings eine gewisse Kenntnis der zugrundeliegenden Grammatik notwendig, wobei dies bei der großen Anzahl an Regeln in vielen Fällen wohl nicht realistisch ist. Die überwiegende Mehrheit der Grammatikausnahmen beschränkt sich allerdings auf wenige Regeln, während andere nur sehr vereinzelt (wenn überhaupt) über Ausnahmen angepasst werden. Die folgenden Tabelle zeigt eine quantitative Auswertung der aktuell vorhanden (lokalen) Grammatikausnahmen für das REW:

Typ	ID	Anzahl
is	extra_info	37.91%
is	form_it/form_it_sub	7.88%
is	record_list_sep	7.17%
is	list_specifier_literal	4.71%
is	text_insertion	4.42%
is	form_bold	3.49%
is	german_sentence	3.30%
is	head_comment	3.20%
is	list_comment	2.47%
is	meaning_record_spec	1.95%
is	record_sep	1.79%
is	meaning_abbr	1.73%
is	form_it_abbr	1.60%
is	form_sep	1.57%
is	meaning_sep	1.50%
is_not	extra_info	1.47%
is	given_name_type_literal	1.38%
is	lang_extra_info	1.28%
is	text_part_dot	1.15%
is	list_specifier_without_col_literal	1.09%

Tabelle 12.1: Lokale Grammatikausnahmen, die mehr als 1% der Gesamtzahl ausmachen

Die Regeln beziehen sich hierbei in der überwältigenden Mehrheit auf die Markierung von diskursiven Teilabschnitten, die nicht als solche erkannt wurden, oder die Markierung von Formen, die eine ungewöhnliche Transkription aufweisen. Somit kann mit einem verhältnismäßig geringem Einarbeitungsaufwand ein großer Anteil von Artikeln nachbearbeitet werden.

12.5.3 Weitere Interaktionsmöglichkeiten

Vor allem die Anreicherung und Vernetzung sollte sinnvollerweise ebenfalls von Nutzenden erweitert und korrigiert werden können. Dies bezieht sich auf die folgenden Anwendungsfälle:

- Verknüpfung von Bedeutungen mit Konzepten aus *Wikidata*
- Anpassung der Hierarchie zwischen verschiedenen Sprachen[5]
- Verknüpfung von Sprachen mit geographischen Polygonen
- Zuordnung von Abkürzungsvarianten zueinander bzw. die Ergänzung von Bedeutungen von Abkürzungen, die nicht im Verzeichnis genannt werden.

Für solche Aufgaben können kleinere *Tools* erstellt werden, die deren Durchführung möglich macht. Die folgende Abbildung zeigt eine einfache Oberfläche um Sprachen oder Regionen durch Erstellung eines Polygons geographisch einzuordnen. Im Anschluss kann eine Erzeugung der entsprechenden Hexagone stattfinden (vgl. Kap. 11.2).

[5] Diese ist aus linguistischer Perspektive nicht unbedingt immer eindeutig. Es sollten somit nur dann Beziehungen angelegt werden, wenn ein gewisser wissenschaftlicher Konsens besteht.

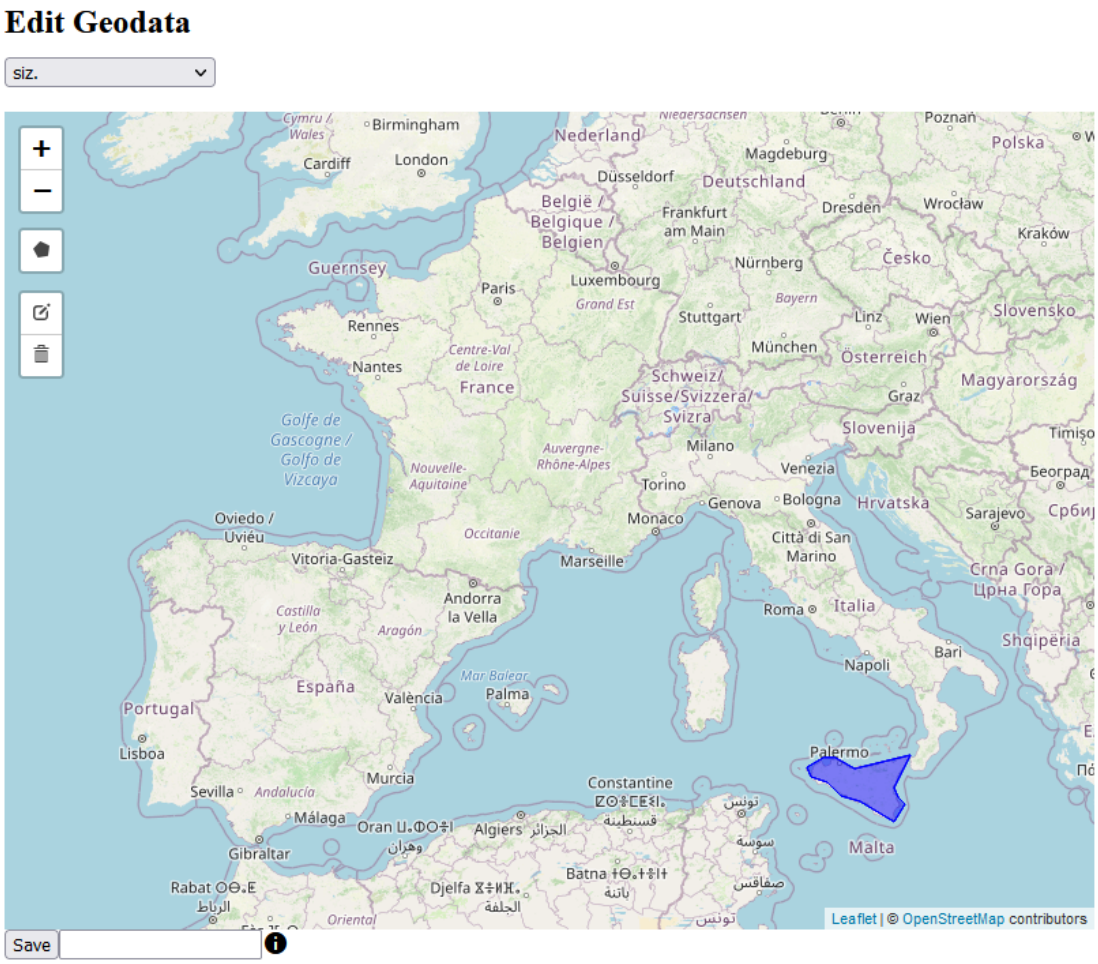


Abbildung 12.30: Einfaches Tool zum Anlegen von Polygondaten für eine Region

Mit Ausnahme der Zuordnung der QIDs von *Wikidata*, die an jeder Stelle, an der die entsprechende Bedeutung angezeigt wird, vorgenommen werden kann, findet der Zugriff zu den anderen Tools über die entsprechenden Detailseiten statt, z.B. hier zur Bearbeitung der Verknüpfung von Bibliographie-Einträgen und externen Ressourcen:

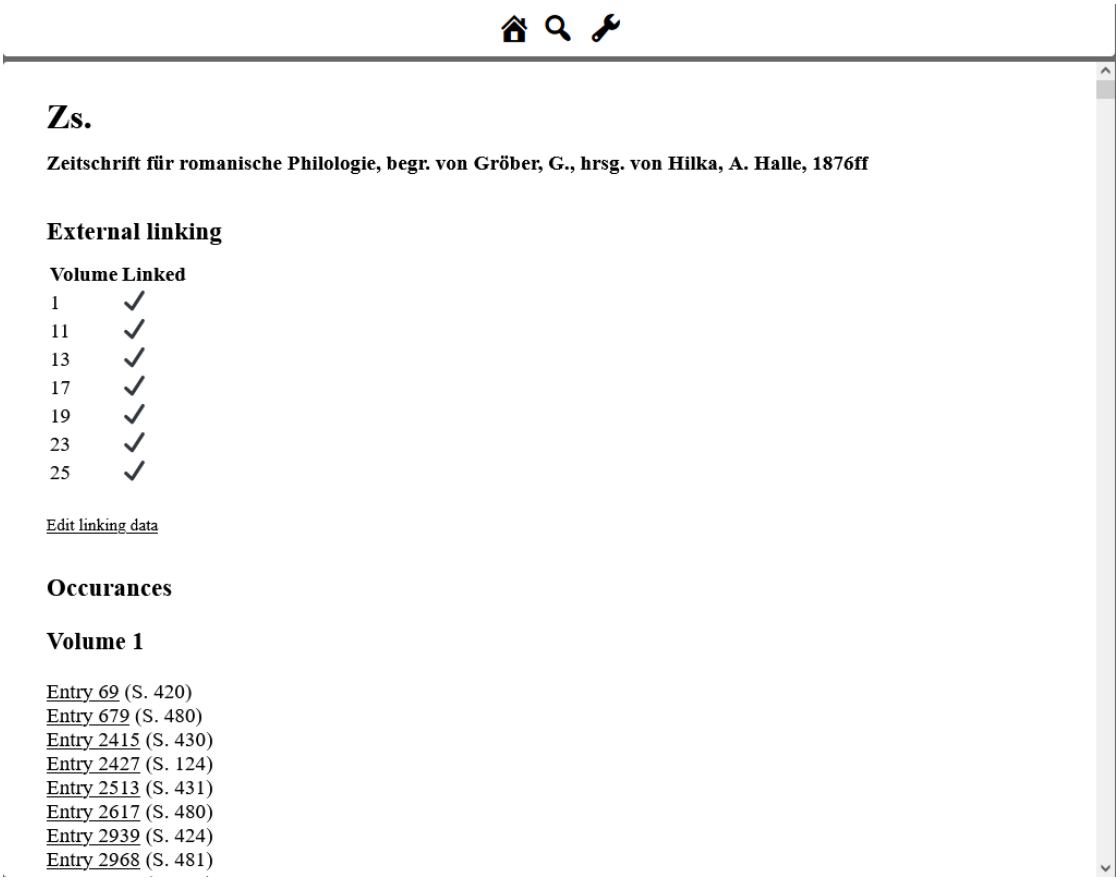


Abbildung 12.31: Detailseite für einen bibliographischen Eintrag (Link)

13 Technischer Zugriff

Dieses Kapitel beschäftigt sich schließlich damit, welche Daten in welcher Form bereitgestellt werden können, um in anderen Kontexten wiederverwendet zu werden. Im ersten Abschnitt (Kap. 13.1) wird der Export der Eingangs- und Ausgangsdaten besprochen, während im weiteren zusätzliche Sekundärdaten beschrieben werden, die sinnvollerweise veröffentlicht werden sollten (Kap. 13.2).

13.1 Verschiedene Datenformate für den Export

Im folgenden werden hier Möglichkeiten aufgezeigt die Information aus dem REW in verschiedenen Datenformaten zum Export anzubieten. Da spätere Anwendungsfälle schwer vorherzusehen sind, werden die Informationen in sehr verschiedenen Verarbeitungsstufen zur Verfügung gestellt. Manche davon basieren auf den nicht oder kaum veränderten Eingangsdaten, während andere rein auf den daraus generierten relationalen Daten aufbauen. Die Beispiele sind dabei vereinfacht und aus Normierungsperspektive zum Teil nicht unbedingt optimal, da sie stark auf den Notationen des REW aufbauen. Sie dienen zur Illustration der Vielseitigkeit der möglichen Exportformate und sind nicht immer in der Praxis völlig unverändert sinnvoll. Als Grundlage für alle Beispiele dient hier ein Ausschnitt, der zwei kurze Artikel enthält, die zu Beginn von Seite 267 des REW vorkommen.

3050. **explanāre** „ebnen“.
It. *spianare*, südit. *škanata* „großes Brot“, teram., mark. *spyanata* „Art Kuchen“ Goidanich 120; schweiz. *epňaná* „Knospen abbrechen“ Gignoux, Zs. 26, 49. — Merlo.
3051. **explētus** „voll“.
Altaquil. *spleto* „vollgültig“.

Abbildung 13.1: Anfang von Seite 267 im REW

Ein erstes einfaches Exportformat stellt der Volltext der Quelle dar. In diesem Fall werden die einzelnen Artikel unverändert aneinandergehängt und im HTML-Format ausgegeben:

```
[...]
3050. <b>explanāre</b> „ebnen".
<br />
It. <i>spianare</i>, südit. <i>škanata</i> „großes Brot", teram., mark. <i>spyanata
Merlo.
<br />
<br />
3051. <b>explētus</b> „voll".
<br />
Altaquil. <i>spleto</i> „vollgültig".
[...]
```

Eine Repräsentation, die noch näher am originalen Quellenmaterial ist, stellt die rein typographische Sicht auf dieses dar. Hierbei werden die zugrundeliegenden Seiten und die jeweiligen Spalten beschrieben. Als Format wird hier (und in den zwei folgenden Beispielen) *TEI* verwendet, da es einer der bekanntesten Standards zur Darstellung von digitalem Text ist (vgl. Kap. 2.3):

```
[...]
<pb n="267" />
<cb />
<p>
   <lg>
      <l>3050. <hi rend="bold">explanāre</hi> „ebnen".</l>
   </lg>
   <lg>
      <l>It. <hi rend="italic">spianare</hi>, südit. <hi rend="italic">škanata</hi>
      <l>Brot", teram., mark. <hi rend="italic">spyanata</hi> „Art</l>
      <l>Kuchen" Goidanich 120; schweiz. <hi rend="italic">epɦaná</hi></l>
      <l>„Knospen abbrechen" Gignoux, Zs. 26,</l>
      <l>49. - Merlo.</l>
   </lg>
</p>
<p>
   <lg>
      <l>3051. <hi rend="bold">explētus</hi> „voll".</l>
   </lg>
   <lg>
```

```
      <l>Altaquil. <hi rend="italic">spleto</hi> „vollgültig".</l>
   </lg>
</p>
[...]
<cb />
[...]
<pb n="268" />
[...]
```

Der Seitenwechsel wird hier durch entsprechende *pb*-Tags (= *page break*) markiert, während die jeweiligen Textsektionen über *cb*-Tags (= *column break*) dargestellt werden. Die Artikel selbst werden in Absätze (*p)* eingefügt, während einzelne Abschnitte innerhalb der Artikel als *lg*-Tags (= *line group*) modelliert werden. Die Auszeichnung geht bis hin zur einzelne Zeile (*l*), was diese Art der Darstellung wohl zur orignalgetreuesten macht, falls eine völlig unverarbeitete textuelle Repräsentation des REW benötigt wird.

Eine etwas weiter verarbeitete Fassung, kann eine annotierte Repräsentation des Quellentexts sein. Diese lässt sich einfach aus dem in Kapitel 5 beschriebenen Strukturbaum erzeugen, wenn unter Einfügung bestimmter XML-Tags der Ursprungstext wieder zusammengesetzt wird. Eine mögliche Darstellung mit den Elementen aus dem *dictionary*-Modul von *TEI* könnte diese Form haben:

```
[...]
<entryFree>
3050. <form><hi rend="bold">explanāre</hi></form> <sense>„ebnen"</sense>.
<lang>It.</lang> <form><hi rend="italic">spianare</hi></form>, <lang>südit.</lang> <form><hi r
<ref>Merlo</ref>.
</entryFree>
<entryFree>
3051. <form><hi rend="bold">explētus</hi></form> <sense>„voll"</sense>.
<lang>Altaquil.</lang> <form><hi rend="italic">spleto</hi></form> <sense>„vollgültig"</sense>.
</entryFree>
[...]
```

TEI kann allerdings genauso für eine mehr datenorientierte Darstellung verwendet werden, die sich allerdings weiter vom Quellenmaterial entfernt, da dieses nicht vollständig strukturiert abgebildet werden kann. Eine strukturiere Darstellung könnte beispielsweise folgende Form haben:

```
[...]
<entry n="3050">
```

```
    <form type="lemma">
        <orth>explanāre</orth>
    </form>

    <sense n="1">
        <def xml:lang="de-DE">ebnen</def>
    </sense>

     <dictScrap>
        <lang xml:lang="it-IT">it.</lang>
        <form n="1">
            <orth>spianare</orth>
        </form>
        <sense n="1">
            <def xml:lang="de-DE">ebnen</def>
        </sense>
     </dictScrap>

    <dictScrap>
        <lang xml:lang="it-IT">südit.</lang>
        <form n="1">
            <orth>škanata</orth>
        </form>
        <sense n="1">
            <def xml:lang="de-DE">großes Brot</def>
        </sense>
     </dictScrap>
     [...]
</entry>
<entry n="3051">
    <form type="lemma">
        <orth>explētus</orth>
    </form>

    <sense n="1">
        <def xml:lang="de-DE">voll</def>
    </sense>

    <dictScrap>
        <lang xml:lang="it-IT">aaquil.</lang>
        <form n="1">
            <orth>spleto</orth>
        </form>
        <sense n="1">
```

```
            <def xml:lang="de-DE">vollgültig</def>
         </sense>
      </dictScrap>
</entry>
[...]
```

Hier wird der grundsätzliche Aufbau der Artikel noch wiedergegeben, indem zuerst die Lemmata und ihre Bedeutungen genannt werden und mit Hilfe des Tags *dictScrap* im folgenden die einzelnen Sprachbelege. Wie man sieht wird hier allerdings beispielsweise die ausgelassene Bedeutung der ersten Form bereits explizit angegeben. Somit kann diese Darstellung schon leichter Grundlage einer technischen Verarbeitung sein, auch wenn diese weiterhin gewisse Nachteile mit sich bringt (s. Kap. 2.3.1).

Eine vollständig datenorientierte Variante wäre der tabellarische Export der Kerndaten (beispielsweise im *CSV*-Format). Entsprechend des Datenmodells aus Kap. 2.2 könnte sie beispielsweise auf folgende (verkürzte und vereinfachte) Weise exportiert werden:

ID	**id_form**	**lang**	**form**	**id_meaning**	**meaning**	**time**	**location**
2208240	37019	lat.	explanāre	548	ebnen		
2208243	37020	it.	spianare	548	ebnen		
2208246	37021	it.	škanata	10827	großes Brot		südit.
2208249	37022	it.	spyanata	4875	Art Kuchen		teram.
2208252	37023	it.	spyanata	4875	Art Kuchen		mark.
2208255	37024	schweiz.	epḥaná	10828	Knospen abbrechen		schweiz.
1764099	37025	lat.	explētus	9259	voll		
1764102	37026	it.	spleto	10829	vollgültig	ait.	aquil.

relation	**form_1**	**form_2**
predecessor	37020	37019
predecessor	37021	37019
predecessor	37022	37019
predecessor	37023	37019
predecessor	37024	37019
predecessor	37026	37025

Als letztes Export-Format sollen hier schließlich Tripel-Daten für die Verwendung im *Semantic Web* genannt werden. In diesem Fall werden die Daten im RDF Turtle Format dargestellt, welches eine Möglichkeit zur kompakten Repräsentation von RDF-Tripeln bildet. Als Ontologien wurden hier *ontolex Lemon* zusammen mit *lexinfo* verwendet (vgl. Kap. 2.3.2).

```
@prefix rew-form: <rew-online.gwi.uni-muenchen.de/?id_form=>.
@prefix rew-meaning: <rew-online.gwi.uni-muenchen.de/?id_meaning=>.
@prefix rew-record: <rew-online.gwi.uni-muenchen.de/?id_record=>.
@prefix ontolex: <http://www.w3.org/ns/lemon/ontolex#> .
@prefix lexinfo: <http://www.lexinfo.net/ontology/3.0/lexinfo#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

rew-record:2208246 a ontolex:LexicalEntry;
    ontolex:lexicalForm rew-form:37021 ;
    ontolex:sense rew-meaning:10827-37021 ;
    lexinfo:etymologicalRoot rew-form:37019 .

rew-form:37021 a ontolex:Form;
    ontolex:writtenRep "škanata"@it .

rew-form:37019 a ontolex:Form;
    ontolex:writtenRep "explanāre"@lat .

rew-meaning:10827-37021 a ontolex:LexicalSense;
    ontolex:reference <https://www.wikidata.org/wiki/Q7802> ;
    ontolex:usage [rdf:value "großes Brot"@de] .
```

Aus Übersichtlichkeitsgründen wurden hier nur zwei sprachliche Formen und die etymologische Relation zwischen beiden dargestellt. Weiterhin wurde zur Vereinfachung die Bedeutung „großes Brot“ mit dem *Wikidata*-Konzept für BROT verknüpft, da *ontolexLemon* nur eine einfache Identitätsbeziehung erlaubt (vgl. Kap. 10.2.2). Auf Basis dieses minimalen Datensatzes könnten grundsätzlich bereits SPARQL-Abfragen ausgeführt werden. So könnte mit folgendem Code das Etymon für die italienische Form zurückgegeben werden:

```
SELECT ?ename
WHERE {
    ?f ontolex:writtenRep "škanata"@it .
    ?r ontolex:lexicalForm ?f .
    ?r lexinfo:etymologicalRoot ?e .
    ?e ontolex:writtenRep ?ename .
}
```

13.2 Weitere exportierbare Daten

Zusätzlich zu den eigentlichen Daten können weitere Exporte zur Verfügung gestellt werden. Im Fall des REW ist das eine Auflistung aller Lemmata, sowie die unterschiedlichen Vernetzungsdaten aus den Kapiteln 10.2 und 11.1. Gerade die Daten für literarische Quellen (also die Verlinkungsinformationen und die Abbildung von Seitenzahlen auf Lemmata) könnten grundsätzlich an anderer Stelle in ähnlicher Weise direkt wiederverwendet werden. Dies gilt auch bei der Zuordnung von Bedeutungen zu QIDs, die nur zu einem geringen Teil automatisiert werden kann (vgl. Kap. 10.2.1), sodass vor allem manuelle zusätzliche Verknüpfungen möglichst zur Verfügung gestellt werden sollten, damit ein Teil der Arbeit für andere Projekte entfällt.

14 Ausblick

Die vorliegende Arbeit behandelt die verschiedenen Aspekte, die für die technische Tiefenerschließung von traditionellen Werken relevant sind. Aufgrund der vielen behandelten Teilaspekte konnte nicht auf alle in entsprechender Tiefe eingegangen werden. Somit sind viele der vorgeschlagenen Lösungsvorschläge für bestimmte Teilprobleme innerhalb des Gesamtprozesses sicherlich weiter optimierbar. Das trifft sowohl für die technische Effizienz (z.B. die Verwendung eines optimierten Parsers für die formellen Grammatiken) als auch für geringere Fehlerraten und Optimierung der manuelle Nachbearbeitung benötigenden Schritte zu.

Das logische Modell, welches über die formelle Grammatik definiert wird, weist eine hohe Mächtigkeit auf und kann in der ganz großen Mehrzahl auch sehr ungewöhnliche und inkonsequente Abschnitte strukturiert erfassen. Der Preis hierfür ist allerdings eine hohe Anzahl an (manuell erstellten) Ausnahmen. Dies könnte ein möglicher Ansatzpunkt für eine deutliche Verbesserung des Systems sein, indem zumindest das Erstellen häufiger Varianten von grammatikbasierten Ausnahmen in einem gewissen Ausmaß automatisiert wird. Denkbar wäre beispielsweise ein Ansatz, der auf Basis von maschinellem Lernen aus bereits vorhandenen Ausnahmen vor dem Parsen des Artikeltexts neue Ausnahmen erstellt und zur formellen Grammatik hinzufügt. Die Erkennung, ob eine Passage einer bestimmten Regel entspricht, wäre in diesem Fall allerdings wohl leichter, als das Auffinden entsprechender Kandidaten aus dem Artikeltext. Auch das manuelle Anlegen von neuen Abkürzungsvarianten könnte durch einen ähnlichen Ansatz zumindest zum Teil ersetzt werden, auch wenn dabei in Betracht gezogen werden müsste, dass unter Umständen bestehende OCR-Fehler oder ähnliches repliziert werden könnten. Weitere mögliche Anwendungsfälle für maschinelles Lernen könnten generalisierbare Probleme wie die Aufspaltung von Bedeutungsbeschreibungen (vgl. Kap. 7.1.2) oder die Auflösung von abgekürzten Formen (vgl. Kap. 7.1.3) sein. Allein auf Basis des REW sind dazu allerdings kaum die notwendigen Mengen an Trainingsdaten vorhanden, sodass dies wohl eher nur bei der Kombination von Daten aus verschieden Quellen möglich ist.

Was die erschlossen Daten angeht, wäre es außerdem sicherlich interessant auch die von der Quelle verworfenen Etymologien strukturiert zu erfassen. Somit könnten „Negativrelationen“ angelegt werden, die beschreiben, dass die Quelle eine bestimmte Wortherkunft explizit ablehnt. Im Vergleich mit anderen Quellen könnten somit widersprüchliche Etymologien besser quantifiziert werden, da auf Basis der aktuell erstellen Daten der Unterschied zwischen einer abgelehnten Etymologie für eine bestimmte Form und dem völligen Fehlen einer Aussage nicht erkannt werden kann.

Dem entgegen steht vor allem die wenig strukturierte Angabe der meisten abgelehnten Etymologien. Trotzdem wäre grundsätzlich zumindest eine exemplarische Behandlung von häufigen Varianten wie „... ist begrifflich ausgeschlossen“ denkbar. Auch widersprüchliche Herkunftsangaben innerhalb der Quelle sollten systematischer behandelt werden. Dazu müsste vor allem Fälle, in denen tatsächlich mehrere Etyma angegeben werden soll, von solchen unterschieden werden, in denen einen Form fälschlicherweise in mehreren Artikeln vorkommt.

Verbesserungspotential ist auch im Bezug auf die Korrektur der sprachlichen Formen vorhanden. Aktuell können nur Fehler behoben werden, die aus Zeichen bestehen, die nicht im vorgegebenen Transkriptionssystem oder Alphabet vorhanden sind (vgl. Kap. 8.2.2). Formell gültige Transkriptionen, die aber trotzdem inkorrekt sind, können nicht systematisch aufgefunden werden, obwohl viele davon (beispielsweise die häufig vorkommende Vertauschung von *u* und *n* durch das Texterkennungssystem) für das menschliche Auge auf den ersten Blick ersichtlich sind. Ein möglicher Lösungsansatz könne ein stochastischer auf Basis von N-Grammen sein, wie sie häufig in der Dialektometrie zum quantitativen Vergleich verwendet werden (vgl. z.B. Zastrow 2011). Wenn auf Basis der gesamtem Formen mit einer gewissen Sprachzugehörigkeit Wahrscheinlichkeiten für das Vorkommen gewisser Zeichenkombinationen errechnet würden, könnten somit ungewöhnliche Konstruktionen und somit in vielen Fällen wohl auch Fehler aufgefunden werden. Dies könnte den regelbasierten Ansatz ergänzen, der im Gegensatz zu einem solchen Modell bei vielen systematischen Fehlern in der Texterkennung gut funktioniert.

Sigle

AIS	Karl Jaberg und Jakob Jud (1928–1940). *Sprach- und Sachatlas Italiens und der Südschweiz*. Bd. 8. Zofingen. URL: http://www3.pd.istc.cnr.it/navigais/.
DRG	Florian Melcher und Robert De Planta, Hrsg. (1939–). *Dicziunari Rumantsch Grischun*. Cuoira: Società Retorumantscha. URL: http://online.drg.ch/.
Duden „Nachschlagwerk“	Dudenredaktion (o. D.). *„Nachschlagwerk“ auf Duden online*. URL: https://www.duden.de/node/238851/revision/484309.
DWDS	DWDS (2004). *Das digitale Wörterbuch der deutschen Sprache*. Berlin.
DWDS: Homonym	DWDS (o. D.). *DWDS: Homonym*. URL: https://www.dwds.de/wb/Homonym.
FAQ Wörterbuchnetz	Kompetenzzentrum – Trier Center for Digital Humanities (o. D.). *FAQ Wörterbuchnetz*. URL: https://woerterbuchnetz.de.
FEW	atilf (o. D.). *Französisches etymologisches Wörterbuch*. URL: https://apps.atilf.fr/lecteurFEW/.
Glottopedia Languoid	Glottopedia (o. D.). *Languoid*. URL: http://www.glottopedia.org/index.php/Languoid.
MariaDB: System-Versioned Tables	MariaDB (o. D.). *System-Versioned Tables*. URL: https://mariadb.com/kb/en/system-versioned-tables/.
NavigAIS	Graziano Tisato (2017). *NavigAIS. AIS Digital Atlas and Navigation Software*. Padua. URL: https://navigais-web.pd.istc.cnr.it/.
OED Online	OED Online, Hrsg. (2020). *Oxford English Dictionary Online*. 3. Aufl. Oxford, UK: Oxford University Press. URL: https://www-oed-com.emedien.ub.uni-muenchen.de.
REW	Wilhelm Meyer-Lübke (1935). *Romanisches etymologisches Wörterbuch 3., vollst. neubearb. Aufl.* Heidelberg: Winter.

Sigle

SchweizId.	SchweizId. (1881-lfd.). *Schweizerisches Idiotikon. Wörterbuch der schweizerdeutschen Sprache. Gesammelt auf Veranstaltung der Antiquarischen Gesellschaft in Zürich unter Beihülfe aus allen Kreisen des Schweizervolkes. Hg. mit Unterstützung des Bundes und der Kantone. Begonnen von Staub, Friedrich und Tobler, Ludwig. Fortges. unter der der Leitung von Bachmann, Albert, u.a.* Frauenfeld. URL: https://www.idiotikon.ch/.
TLIO	Lino Leonardi (2017). *Tesoro della Lingua Italiana delle Origini Il primo dizionario storico dell'italiano antico che nasce direttamente in rete fondato da Pietro G. Beltrami. Data di prima pubblicazione: 15.10.1997.* URL: http://tlio.ovi.cnr.it/TLIO/.
VerbaAlpina	Thomas Krefeld und Stephan Lücke, Hrsg. (2014–). *VerbaAlpina. Der alpine Kulturraum im Spiegel seiner Mehrsprachigkeit.* München. DOI: http://dx.doi.org/10.5282/verba-alpina. URL: https://www.verba-alpina.gwi.uni-muenchen.de/it/?page_id=10&db=182.
Wikidata Help:Description	Wikidata (o. D.[a]). *Help:Description.* URL: https://www.wikidata.org/wiki/Help:Description.
Wikidata Startseite	Wikidata (o. D.[b]). *Wikidata Startseite.* URL: https://www.wikidata.org/wiki/Wikidata:Main_Page.
Wikidata:Notability	Wikidata (o. D.[c]). *Wikidata:Notability.* URL: https://www.wikidata.org/wiki/Wikidata:Notability.
Wörterbuchnetz	Kompetenzzentrum - Trier Center for Digital Humanities (o. D.). *Wörterbuchnetz.* Trier. URL: https://woerterbuchnetz.de.

Literatur

Abgaz, Yalemisew (2020). "Using OntoLex-Lemon for representing and interlinking lexicographic collections of Bavarian dialects". In: *Proceedings of the 7th Workshop on Linked Data in Linguistics (LDL-2020)*, S. 61–69.

Abromeit, Frank u. a. (2016). "Linking the Tower of Babel: modelling a massive set of etymological dictionaries as RDF". In: *Proceedings of the 5th Workshop on Linked Data in Linguistics (LDL-2016): Managing, Building and Using Linked Language Resources*, S. 11–19.

Aggarwal, Charu C (2018). *Machine learning for text*. Bd. 848. Springer.

atilf (o. D.). *Französisches etymologisches Wörterbuch*. URL: `https://apps.atilf.fr/lecteurFEW/`.

Bosque-Gil, Julia, Jorge Gracia, John McCrae u. a. (o. D.). *The OntoLex Lemon Lexicography Module*. 2019. URL: `https://www.w3.org/2019/09/lexicog/`.

Bosque-Gil, Julia, Jorge Gracia und Elena Montiel-Ponsoda (2017). "Towards a Module for Lexicography in OntoLex." In: *LDK Workshops*, S. 74–84.

Bowers, Jack und Philipp Stöckle (2018). "TEI and Bavarian dialect resources in Austria: updates from the DBÖ and WBÖ". In: *Proceedings of the Second Workshop on Corpus-Based Research in the Humanities (CRH-2)*. Hrsg. von Andrew U. Frank u. a. Wien: Gerastree Proceedings, S. 45–54.

Burch, Thomas und Andrea Rapp (2006). "Das Wörterbuch-Netz: Verfahren-Methoden-Perspektiven". In: *Geschichte im Netz: Praxis, Chancen, Visionen. Beiträge der Tagung. hist*, S. 607–627.

Bürgermeister, Martina (2019). "Extending Versioning in Collaborative Research". In: *Versioning Cultural Objects : Digital Approaches*. Hrsg. von Roman Bleier. Bd. 13. Norderstedt: BoD, S. 171–190. URL: `https://kups.ub.uni-koeln.de/10654/`.

Cantara, Linda (2005). "The text-encoding initiative: Part 1". In: *OCLC Systems & Services: International digital library perspectives*.

Carstensen, Kai-Uwe (2010). "Anwendungen". In: *Computerlinguistik und Sprachtechnologie: Eine Einführung*. Hrsg. von Kai-Uwe Carstensen u. a. Heidelberg: Spektrum Akademischer Verlag, S. 553–658. DOI: `10.1007/978-3-8274-2224-8_5`. URL: `https://doi.org/10.1007/978-3-8274-2224-8_5`.

Chapman, Nigel P (1987). *LR parsing: theory and practice*. CUP Archive.

Chavula, Catherine und C Maria Keet (2014). "Is lemon Sufficient for Building Multilingual Ontologies for Bantu Languages?" In.

Chiarcos, Christian, Sebastian Hellmann und Sebastian Nordhoff (2011). "Towards a Linguistic Linked Open Data cloud: The Open Linguistics Working Group." In: *Trait. Autom. des Langues* 52.3, S. 245–275.

Chomsky, N. (1956). “Three models for the description of language”. In: *IRE Transactions on Information Theory* 2.3, S. 113–124. DOI: 10.1109/TIT.1956.1056813.

Cimiano, Philipp, Paul Buitelaar u. a. (2011). “LexInfo: A declarative model for the lexicon-ontology interface”. In: *Journal of Web Semantics* 9.1, S. 29–51.

Cimiano, Philipp, John P. McCrae und Paul Buitelaar (2016). *Lexicon Model for Ontologies: Community Report*.

Consortium, Text Encoding Initiative (2021). *P5: Guidelines for Electronic Text Encoding and Interchange*. URL: https://tei-c.org/release/doc/tei-p5-doc/en/html/index.html.

Crist, Sean (2011). “Processing the text of bilingual print dictionaries”. In: URL: http://www.sean-crist.com/all/crist_dictionaries_20111210.pdf.

Damerau, Fred J (1964). “A technique for computer detection and correction of spelling errors”. In: *Communications of the ACM* 7.3, S. 171–176.

Declerck, Thierry (2017). “Encoding Lexicographic Data in Ontolex: Lessons Learned and Open Questions”. In: *LDK Workshops*.

Digital Humanities, Kompetenzzentrum - Trier Center for (o. D.). *Wörterbuchnetz*. Trier. URL: https://woerterbuchnetz.de.

Digital Humanities, Kompetenzzentrum – Trier Center for (o. D.). *FAQ Wörterbuchnetz*. URL: https://woerterbuchnetz.de.

Dudenredaktion (o. D.). *„Nachschlagwerk“ auf Duden online*. URL: https://www.duden.de/node/238851/revision/484309.

DWDS (2004). *Das digitale Wörterbuch der deutschen Sprache.* Berlin.

— (o. D.). *DWDS: Homonym*. URL: https://www.dwds.de/wb/Homonym.

Ekbal, Asif, Rejwanul Haque und Sivaji Bandyopadhyay (2007). “Bengali part of speech tagging using conditional random field”. In: *Proceedings of seventh international symposium on natural language processing (SNLP2007)*, S. 131–136.

Elspaß, Stephan und Robert Möller (2006). “Internet-Exploration: zu den Chancen, die eine Online-Erhebung regional gefärbter Alltagssprache bietet”. In: *Osnabrücker Beiträge zur Sprachtheorie* 71, S. 141–156.

Ford, Bryan (2002). “Packrat Parsing : a Practical Linear-Time Algorithm with Backtracking”. Diss.

— (2004). “Parsing Expression Grammars: A Recognition-Based Syntactic Foundation”. In: *Symposium on Principles of Programming Languages*. ACM Press, S. 111–122. DOI: 10.1145/982962.964011.

Glottopedia (o. D.). *Languoid*. URL: http://www.glottopedia.org/index.php/Languoid.

Good, Jeff und Calvin Hendryx-Parker (2006). “Modeling contested categorization in linguistic databases”. In: *Proceedings of the EMELD 2006 Workshop on Digital Language Documentation: Tools and standards: The state of the art*, S. 20–22.

Hallig, Rudolf und Walther von Wartburg (1963 [1952]). *Begriffssystem als Grundlage für die Lexikographie. Versuch eines Ordnungssystems*. 2. Aufl. Berlin: Akademie-Verlag.

Hauser, Ralf und Angelika Storrer (2017). “Probleme und Lösungen beim Parsen von Wörterbüchern”. In: *Lexikon und Text. Wiederverwendebare Methoden und Ressourcen zur linguistischen Erschließung des Deutschen*. Niemeyer, S. 53–67.

Heyn, Matthias (1992). *Zur Wiederverwendung maschinenlesbarer Wörterbücher: Eine computergestützte metalexikographische Studie am Beispiel der elektronischen Edition des „Oxford advanced learner's dictionary of current English“*. Max Niemeyer Verlag. DOI: doi:10.1515/9783111341101. URL: https://doi.org/10.1515/9783111341101.

Horák, Aleš, Adam Rambousek u. a. (2017). “Lexicography and natural language processing”. In: *The Routledge handbook of lexicography*, S. 179–196.

Huang, Silu u. a. (2017). “OrpheusDB: Bolt-on Versioning for Relational Databases”. In: *CoRR* abs/1703.02475. URL: http://arxiv.org/abs/1703.02475.

Jaberg, Karl und Jakob Jud (1928–1940). *Sprach- und Sachatlas Italiens und der Südschweiz*. Bd. 8. Zofingen. URL: http://www3.pd.istc.cnr.it/navigais/.

Jean-Caurant, Axel u. a. (2017). “Lexicographical-Based Order for Post-OCR Correction of Named Entities”. In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Bd. 01, S. 1192–1197. DOI: 10.1109/ICDAR.2017.197.

Khan, Fahad (2020). “Representing Temporal Information in Lexical Linked Data Resources”. In: *Proceedings of the 7th Workshop on Linked Data in Linguistics (LDL-2020)*. Marseille, France: European Language Resources Association, S. 15–22. URL: https://aclanthology.org/2020.ldl-1.3.

Krcmar, Helmut (2015). *Informationsmanagement*. 6. Aufl. Heidelberg: Springer-Verlag Berlin.

Krefeld, Thomas und Stephan Lücke, Hrsg. (2014–). *VerbaAlpina. Der alpine Kulturraum im Spiegel seiner Mehrsprachigkeit*. München. DOI: http://dx.doi.org/10.5282/verba-alpina. URL: https://www.verba-alpina.gwi.uni-muenchen.de/it/?page_id=10&db=182.

— (2018). “Typisierung”. In: *Methodologie*. VerbaAlpina-de 21/2. URL: https://doi.org/10.5282/verba-alpina?urlappend=%3Fpage_id%3D493%26db%3D212%26letter%3DT%2358.

— (2021). “Crowdsourcing”. In: *Methodologie*. VerbaAlpina-de 21/2. URL: https://doi.org/10.5282/verba-alpina?urlappend=%3Fpage_id%3D493%26db%3D212%26letter%3DC%2312.

Krefeld, Thomas und Florian Zacherl (2022). “Ontologie”. In: *Methodologie*. VerbaAlpina-de 22/1. URL: https://doi.org/10.5282/verba-alpina?urlappend=%3Fpage_id%3D493%26db%3D221%26letter%3DO%23180.

Kudo, Taku, Kaoru Yamamoto und Yuji Matsumoto (2004). “Applying conditional random fields to Japanese morphological analysis”. In: *Proceedings of the 2004 conference on empirical methods in natural language processing*, S. 230–237.

Kunze, Claudia und Lothar Lemnitzer (2007). *Computerlexikographie. Eine Einführung*. Tübingen: Narr.

Lafferty, John, Andrew McCallum und Fernando CN Pereira (2001). “Conditional random fields: Probabilistic models for segmenting and labeling sequence data”. In.

Lehmann, Jens u. a. (2015). “Dbpedia-a large-scale, multilingual knowledge base extracted from wikipedia”. In: *Semantic web* 6.2, S. 167–195.

Leonardi, Lino (2017). *Tesoro della Lingua Italiana delle Origini Il primo dizionario storico dell'italiano antico che nasce direttamente in rete fondato da Pietro G. Beltrami. Data di prima pubblicazione: 15.10.1997*. URL: http://tlio.ovi.cnr.it/TLIO/.

Levenshtein, Vladimir I u. a. (1966). “Binary codes capable of correcting deletions, insertions, and reversals”. In: *Soviet physics doklady*. Bd. 10. 8, S. 707–710.

Lücke, Stephan (2016). “Digitalisierung”. In: *Methodologie*. VerbaAlpina-de 21/2. URL: https://doi.org/10.5282/verba-alpina?urlappend=%3Fpage_id%3D493%26db%3D212%26letter%3DD%2315.

— (2021a). “Normdaten”. In: *Methodologie*. VerbaAlpina-de 21/2. URL: https://doi.org/10.5282/verba-alpina?urlappend=%3Fpage_id%3D493%26db%3D212%26letter%3DN%23114.

— (2021b). “Versionierung”. In: *Methodologie*. VerbaAlpina-de 21/2. URL: https://doi.org/10.5282/verba-alpina?urlappend=%3Fpage_id%3D493%26db%3D212%26letter%3DV%2361.

Lüschow, Andreas (2020). “Automatische Extraktion und semantische Modellierung der Einträge einer Bibliographie französischsprachiger Romane.” In: *DHd*.

MariaDB (o. D.). *System-Versioned Tables*. URL: https://mariadb.com/kb/en/system-versioned-tables/.

McCrae, John, Guadalupe Aguado-de-Cea u. a. (2012). “Interchanging lexical resources on the semantic web”. In: *Language Resources and Evaluation* 46.4, S. 701–719.

McCrae, John, Dennis Spohr und Philipp Cimiano (2011). “Linking lexical resources and ontologies on the semantic web with lemon”. In: *Extended Semantic Web Conference*, S. 245–259.

McCrae, John P u. a. (2017). “The Ontolex-Lemon model: development and applications”. In: *Proceedings of eLex 2017 conference*, S. 19–21.

Melcher, Florian und Robert De Planta, Hrsg. (1939–). *Dicziunari Rumantsch Grischun*. Cuoira: Società Retorumantscha. URL: http://online.drg.ch/.

Meyer-Lübke, Wilhelm (1935). *Romanisches etymologisches Wörterbuch 3., vollst. neubearb. Aufl.* Heidelberg: Winter.

Miller, George A. (1995). “WordNet: A Lexical Database for English”. In: *Communications of the ACM* 38, S. 39–41. DOI: 10.1145/219717.219748.

Möller, Robert und Stephan Elspaß (2014). “Zur Erhebung und kartographischen Darstellung von Daten zur deutschen Alltagssprache online: Möglichkeiten und Grenzen”. In: *Fabio Tosques (ed.)* 20, S. 121–131.

Moss, Aaron (2017). “Derivatives of Parsing Expression Grammars”. In: *Electronic Proceedings in Theoretical Computer Science* 252, S. 180–194. DOI: 10.4204/eptcs.252.18. URL: https://doi.org/10.4204%2Feptcs.252.18.

Nadeau, David und Satoshi Sekine (2007). “A survey of named entity recognition and classification”. In: *Lingvisticae Investigationes* 30, S. 3–26.

Neumann, Gerald (2007). “Wörterbücher als digitale Ressourcen für Mensch und Maschine – Die Wörterbuchprojekte der Berlin-Brandenburgischen Akademie der

Wissenschaften". In: *Geschichte im Netz: Praxis, Chancen, Visionen. Beiträge der Tagung .hist 2006*. Hrsg. von Daniel Burckhardt, Rüdiger Hohls und Claudia Prinz. Berlin: Humboldt-Universität zu Berlin. DOI: http://dx.doi.org/10.18452/17821.

Nikula, Henrik (2013). "Perspektivität und Polysemie im Lexikon und Wörterbuch". In.

Online, OED, Hrsg. (2020). *Oxford English Dictionary Online*. 3. Aufl. Oxford, UK: Oxford University Press. URL: https://www-oed-com.emedien.ub.uni-muenchen.de.

Parr, T. J. und R. W. Quong (1995). "ANTLR: A predicated-LL(k) parser generator". In: *Software: Practice and Experience* 25.7, S. 789–810. DOI: 10.1002/spe.4380250705.

Prätor, Klaus (2011). "Zur Zukunft des Zitierens. Identität, Referenz und Granularität digitaler Dokumente". In: *Editio* 25, S. 170–183.

Rabinowitz, Adam u. a. (2016). "Making sense of the ways we make sense of the past: The PeriodO project". In: *Bulletin of the Institute of Classical Studies* 59.2, S. 42–55. DOI: 10.1111/j.2041-5370.2016.12037.x. URL: https://doi.org/10.1111/j.2041-5370.2016.12037.x.

Renders, Pascale (2011). "Modélisation d'un discours étymologique. Prolégomènes à l'informatisation du Französisches Etymologisches Wörterbuch." Diss. Liège & Nancy, Belgique & France: Université de Liège & Nancy-Université. URL: https://hdl.handle.net/2268/94407.

Romary, Laurent und Toma Tasovac (2018). "TEI Lex-0: A Target Format for TEI-Encoded Dictionaries and Lexical Resources". In: *TEI Conference and Members' Meeting*. Tokyo, Japan. URL: https://hal.inria.fr/hal-02265312.

Scherrer, Yves und Owen Rambow (2010). "Natural Language Processing for the Swiss German Dialect Area". In: *Semantic Approaches in Natural Language Processing - Proceedings of the Conference on Natural Language Processing 2010 (KONVENS)*. Hrsg. von M. Pinkal u. a. Saarbrücken, Germany: Universaar, S. 93–102. URL: https://archive-ouverte.unige.ch/unige:22826.

SchweizId. (1881-lfd.). *Schweizerisches Idiotikon. Wörterbuch der schweizerdeutschen Sprache. Gesammelt auf Veranstaltung der Antiquarischen Gesellschaft in Zürich unter Beihülfe aus allen Kreisen des Schweizervolkes. Hg. mit Unterstützung des Bundes und der Kantone. Begonnen von Staub, Friedrich und Tobler, Ludwig. Fortges. unter der der Leitung von Bachmann, Albert, u.a.* Frauenfeld. URL: https://www.idiotikon.ch/.

Steinwart, Ingo und Andreas Christmann (2008). *Support vector machines*. Springer Science & Business Media.

Sutton, Charles und Andrew McCallum (2010). *An Introduction to Conditional Random Fields*. DOI: 10.48550/ARXIV.1011.4088. URL: https://arxiv.org/abs/1011.4088.

Tasovac, Toma (2020). *The Historical Dictionary as an Exploratory Tool: A Digital Edition of Vuk Stefanovic Karadzic's Lexicon Serbico-Germanico-Latinum*. Trinity College Dublin.School of Linguistic, Speech & Communication Sciences. URL: http://hdl.handle.net/2262/92750.

team, opencv dev (2014). *Morphological Transformations*. URL: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html.

tesseract-ocr (o. D.). *4.0-Accuracy-and-Performance*. URL: https://tesseract-ocr.github.io/tessdoc/tess4/4.0-Accuracy-and-Performance.html.

Thiemann, Peter und Matthias Neubauer (2008). "Macros for Context-Free Grammars". In: *PPDP '08*. New York, NY, USA: Association for Computing Machinery, S. 120–130. DOI: 10.1145/1389449.1389465. URL: https://doi.org/10.1145/1389449.1389465.

Tisato, Graziano (2017). *NavigAIS. AIS Digital Atlas and Navigation Software*. Padua. URL: https://navigais-web.pd.istc.cnr.it/.

Tittel, Sabine und Christian Chiarcos (2018). "Historical Lexicography of Old French and Linked Open Data: Transforming the resources of the Dictionnaire étymologique de l'ancien français with OntoLex-Lemon". In.

Unterstein, Michael und Günter Matthiessen (2012). "Normalformen in relationalen Datenbanken". In: *Relationale Datenbanken und SQL in Theorie und Praxis*. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 237–268. DOI: 10.1007/978-3-642-28986-6_11. URL: https://doi.org/10.1007/978-3-642-28986-6_11.

Veen, Theo van (2019). "Wikidata. From "an" Identifier to "the" Identifier". In: *Information Technology and Libraries* 38.2, S. 72–81. DOI: 10.6017/ital.v38i2.10886. URL: https://ejournals.bc.edu/index.php/ital/article/view/10886.

Voß, Jakob u. a. (2014). *Normdaten in Wikidata*. Hochschule Hannover Fakultät III Abteilung Information & Kommunikation. URL: https://hshdb.github.io/normdaten-in-wikidata/.

Vossen, Gottfried und Kurt-Ulrich Witt (2002). "Anwendungen kontextfreier Sprachen". In: *Grundlagen der Theoretischen Informatik mit Anwendungen: Eine Einführung für Studierende der Informatik, Wirtschaftsinformatik und Technischen Informatik*. Wiesbaden: Vieweg+Teubner Verlag, S. 203–231. DOI: 10.1007/978-3-322-96901-9_7. URL: https://doi.org/10.1007/978-3-322-96901-9_7.

Wikidata (o. D.[a]). *Help:Description*. URL: https://www.wikidata.org/wiki/Help:Description.

— (o. D.[b]). *Wikidata Startseite*. URL: https://www.wikidata.org/wiki/Wikidata:Main_Page.

— (o. D.[c]). *Wikidata:Notability*. URL: https://www.wikidata.org/wiki/Wikidata:Notability.

Zacherl, Florian (2022). "Linguistische Online-Ressourcen auf Basis traditioneller Werke. Anforderungen und digitale Möglichkeiten am Beispiel des Romanischen Etymologischen Wörterbuchs". In: *apropos [Perspektiven auf die Romania]* 9, S. 254–275. DOI: https://doi.org/10.15460/apropos.9.1895.

— (In Vorb.). “Automatisierte Erschließung von strukturierten Daten aus Wörterbuchtexten”. In: *Digitale romanistische Sprachwissenschaft: Stand und Perspektiven*. Hrsg. von Lidia Becker u. a. Tübingen: Narr Francke Attempto.
Zastrow, Thomas (2011). “Neue Analyse-und Visualisierungsmethoden in der Dialektometrie”. Diss. Universität Tübingen.
Zimmermann, Ralf (2006). “BAYDAT - Die bayerische Dialektdatenbank”. Diss. Universität Würzburg. DOI: 10.25972/OPUS-1938.