Models and Algorithms for Automatic Detection of Language Evolution

Towards Finding and Interpreting of Content in Long-Term Archives

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des akademischen Grades
Doktorin der Naturwissenschaften
Dr. rer. nat.
genehmigte Dissertation von

M.Sc. Nina N. Tahmasebi

geboren am 29. Oktober 1982 in Tehran, Iran

2013

Referent: Prof. Dr. Wolfgang Nejdl
Koreferent: Prof. Dr. Erich Neuhold
Vorsitzender: Prof. Dr. Udo Lipeck
Tag der Promotion: 13. November 2013

Abstract

With advances in technology and culture and through high impact events, our language changes. We invent new words, add or change meanings of existing words and rename existing things. This results in a dynamic language that progresses with our needs and provides us with the possibilities to express ourselves and describe the world around us. This phenomenon is called **language evolution**.

Unfortunately, our language does not carry a memory; words, expressions and meanings used in the past are forgotten over time. Therefore, language evolution limits us when we want to find and interpret information about the past from historical documents.

The primary goals of this thesis are the following: (1) to provide deeper insight into the problems of language evolution; (2) to take the first steps towards fully automated methods of detecting language evolution; and (3) to discuss future directions to fully utilize language evolution.

We begin by analyzing the problems language evolution causes on two high-level objectives; the **finding** and **interpreting** of content in long-term archives. We present a classification of language evolution and a model, called **term concept graphs**, to describe different types of evolution. We continue with an in-depth analysis of two specific types of evolution, namely word sense evolution and named entity evolution.

The first step in finding **word sense evolution** is to discover word senses present in a collection of text. We do this using word sense discrimination and start by evaluating the applicability of such algorithms to historical data. We then continue by formally defining word sense evolution, and present models for finding evolution that build on iteratively merging term concept graphs. We evaluate using a set of terms with known sense changes, and find that the corresponding evolution can successfully be found for most of these terms. We can track evolution within specific senses, including narrowing and broadening, and group senses into concepts. In addition, the evolution is detected at the time of the actual change, or with a slight delay of 2–10 years.

We then consider **named entity evolution** and go beyond existing methods for finding different names used for the same entity over time. Our methodology builds on the use of **change periods** with a high likelihood of name changes and searches for evolution only in these periods. Our method avoids comparing arbitrary term contexts and recurrent computations, and shows promising results.

Because our problem deals with large datasets, long time spans and diverse domains, we opt for automatic methods that do not require human input or existing resources such as dictionaries. For our experiments, we make use of *The Times Archive* (1785-1985) and the *New York Times Annotated Corpus* (1987-2007). The former provides us with a large sample of modern English in a realistic setting with noisy, unstructured text. The latter is a modern, error free collection and serves as a comparison corpus. It is also used to extend the time span for the word sense evolution experiments, resulting in 222 years of text. For each of the two classes, we provide example **applications for search and browsing**.

Zusammenfassung

Durch den technologischen und kulturellen Fortschritt sowie durch Ereignisse mit großem Einfluss verändert sich unsere Sprache. Es werden neue Wörter erfunden, es werden neue Bedeutungen zu Wörtern hinzugefügt oder geändert und es werden die Namen von Dingen geändert. Dadurch entsteht eine Dynamik in der Sprache, die mit unseren Bedürfnissen Schritt hält und uns die Möglichkeit gibt, uns und die Welt um uns herum auszudrücken. Das entstehende Phänomen heißt **Sprachevolution**.

Leider hat Sprache kein Gedächtnis; in der Vergangenheit genutzte Worte, Ausdrücke und Bedeutungen werden im Laufe der Zeit vergessen. Die Evolution der Sprache begrenzt daher die Möglichkeiten Informationen über die Vergangenheit in historischen Dokumenten finden und zu interpretieren.

Die primären Ziele dieser Arbeit sind die Folgenden: (1) einen tiefere Einblicke in das Problem der Sprachevolution zu geben, (2) ersten Schritte zur Entwicklung einer automatisierten Methode zur Erkennung von Sprachevolution zu machen, und (3) Vorschläge für die weitere Vorangehensweise zur vollständigen Nutzbarmachung von Sprachevolution zum Auffinden und Interpretieren von Inhalten in Langzeitarchiven zu diskutieren.

Wir analysieren das Problem der Sprachevolution (1) beim **Finden**, und (2) zum **Interpretiere**n von Inhalten in Langzeitarchiven. Wir stellen eine Klassifikation von Sprachevolution vor sowie ein **Term-Concept-Graph** genanntes Modell zur Beschreibung unterschiedlicher Evolutionstypen. Anschließend analysieren wir zwei spezifische Evolutionstypen: die Evolution von Wortbedeutungen sowie von benannten Entitäten.

Der erste Schritt zum Finden von **Wortbedeutungsevolution** ist die Identifikation von Wortbedeutungen in einer Textkollektion. Wir benutzen dafür Word Sense Discrimination zur Abgrenzung von Wordbedeutungen. Zunächst evaluieren wir die Anwendbarkeit solcher Algorithmen auf historische Textkorpora. Anschließend präsentieren wir ein formales Modell für die Wortbedeutungsevolution für die Problemdefinition und zum Finden von Evolution, die auf dem iterativen Zusammenführen Term-Concept-Graphen basieren. Für die Evaluation verwenden wir eine Liste von Begriffen, deren Bedeutungsänderungen wir kennen und prüfen inwieweit die Begriffe erfolgreich im Testdatensatz erkannt werden.

Als nächstes betrachten wir die **Evolution von benannten Entitäten** und gehen über existierende Methoden hinaus, die zum Finden verschiedener Namen für dieselbe Entität über die Zeit entwickelt wurden. Unsere Methode basiert auf **Veränderungsperioden**, in denen mit hoher Wahrscheinlichkeit Namensänderungen stattfinden, und suchen in diesen Perioden nach Evolution. Unsere Methode vermeidet wiederholte Berechnungen und verhindert, dass der Kontext von Wörtern vergleichen wird, die keinen Bezug zueinander haben. Die Ergebnisse sind vielversprechend.

Für die beschriebene Problematik ist es notwendig große Datenmengen über lange Zeiträume sowie in unterschiedlichen Domänen zu analysieren. Wir haben uns deshalb für automatisierte Methoden entschieden, für die weder manuelle Eingaben noch vorhandene externe Ressourcen als Hintergrundwissen nötig sind. Für unsere Experimente verwenden wir das Archiv von *The Times* (1785-1985) und den *New York Times Annotated Corpus* (1987-2007). Ersterer gibt uns ein großes, repräsentatives Beispiel von modernem Englisch in einem realistischen Umfeld mit fehlerbehaftetem unstrukturiertem Text. Letzteres ist eine moderne, fehlerfreie Sammlung und dient als Vergleichskorpus. Gleichzeitig ergänzen sich die Archive um den Zeitraum für die Experimente zur Wortbedeutungsevolution auf insgesamt 222 Jahre zu verlängern. Zusätzlich präsentieren wir beispielhafte **Anwendungen für die Suche und das Browsing** in historischen Kollektionen.

Keywords: Language Evolution,Word Sense Evolution, Named Entity Evolution

Stichwörter: Sprachevolution, Wordbedeutungsevolution, Zeitliche Namensevolution

Foreword

The algorithms and experiments presented in this thesis are based upon work that has been published or is under review at various conferences and journals. Below follows a chapter by chapter presentation of these works.

Motivation

Nina Tahmasebi, Gerhard Gossen, and Thomas Risse. Which Words Do You Remember? Temporal Properties of Language Use in Digital Archives. In *Proceedings of the Second International Conference on Theory and Practice of Digital Libraries - TPDL 2012, Paphos, Cyprus*, volume 7489, pages 32–37. Springer, 2012c. doi: 10.1007/978-3-642-33290-6_4.

Language Evolution Model and Classification

Nina Tahmasebi, Gerhard Gossen, Nattiya Kanhabua, Helge Holzmann, and Thomas Risse. NEER: An Unsupervised Method for Named Entity Evolution Recognition. In *Proceedings of COLING 2012*, pages 2553–2568, Mumbai, India, December 2012a. The COLING 2012 Organizing Committee. URL `http://www.aclweb.org/anthology/C12-1156`.

Nina Tahmasebi. Automatic Detection of Terminology Evolution. In *Proceedings of On the Move to Meaningful Internet Systems: OTM 2009 Workshops, Vilamoura, Portugal*, volume 5872 of *Lecture Notes in Computer Science*, pages 769–778. Springer, 2009. doi: 10.1007/978-3-642-05290-3_93.

Nina Tahmasebi, Tereza Iofciu, Thomas Risse, Claudia Niederée, and Wolf Siberski. Terminology Evolution in Web Archiving: Open Issues. In *Proceedings of the 8th International Web Archiving Workshop (IWAW'08), Aarhus, Denmark, 18th & 19th September*, 2008. `http://iwaw.net/08/IWAW2008-Tahmasebi.pdf`.

Finding and Evaluating Word Senses

Nina Tahmasebi, Kai Niklas, Gideon Zenz, and Thomas Risse. On the applicability of word sense discrimination on 201years of modern English. *International Journal on Digital Libraries*, 13(3-4):135–153, 2013. doi: 10.1007/s00799-013-0105-8.

Nina Tahmasebi, Thomas Risse, and Stefan Dietze. Towards Automatic Language Evolution Tracking, A Study on Word Sense Tracking. In *Proceedings of Workshop on Knowledge Evolution and Ontology Dynamics (EvoDyn'11)*, 2011.

Nina Tahmasebi, Kai Niklas, Thomas Theuerkauf, and Thomas Risse. Using word sense discrimination on historic document collections. In *Proceedings of the 2010 Joint International Conference on Digital Libraries, (JCDL'10), Gold Coast, Queensland, Australia*, pages 89–98, 2010a. doi: 10.1145/1816123.1816137.

Finding Word Sense Evolution

Nina Tahmasebi, Kai Niklas, Gideon Zenz, and Thomas Risse. On the applicability of word sense discrimination on 201years of modern English. *International Journal on Digital Libraries*, 13(3-4):135–153, 2013. doi: 10.1007/s00799-013-0105-8.

Gideon Zenz, Nina Tahmasebi, and Thomas Risse. Towards mobile language evolution exploitation. *Multimedia Tools and Applications*, 66(1):147–159, 2013. doi: 10.1007/s11042-011-0973-0.

Accepted with revision:

Nina Tahmasebi and Thomas Risse. Models and Algorithms for Automatic Detection of Language Evolution. *ACM Transactions on Speech and Language Processing, 2013.*

Finding Named Entity Evolution

Nina Tahmasebi, Gerhard Gossen, Nattiya Kanhabua, Helge Holzmann, and Thomas Risse. NEER: An Unsupervised Method for Named Entity Evolution Recognition. In *Proceedings of COLING 2012*, pages 2553–2568, Mumbai, India, December 2012a. The COLING 2012 Organizing Committee. URL `http://www.aclweb.org/anthology/C12-1156`.

Helge Holzmann, Gerhard Gossen, and Nina Tahmasebi. *fokas*: Formerly Known As – A Search Engine Incorporating Named Entity Evolution. In *Proceedings of COLING 2012: Demonstration Papers*, pages 215–222, Mumbai, India, December 2012. The COLING 2012 Organizing Committee. URL `http://www.aclweb.org/anthology/C12-3027`.

Remaining publications

Bogdan Pogorelc, Artur Lugmayr, Björn Stockleben, Radu-Daniel Vatavu, Nina Tahmasebi, Estefanía Serral, Emilija Stojmenova, Bojan Imperl, Thomas Risse, Gideon Zenz, and Matjaž Gams. Ambient bloom: new business, content, design and models to increase the semantic ambient media experience. *Multimedia Tools and Applications*, pages 1–26, 2012. doi: 10.1007/s11042-012-1228-4.

Christopher Kunz, Nina Tahmasebi, Thomas Risse, and Matthew Smith. Detecting Credential Abuse in the Grid Using Bayesian Networks. In *Proceedings of the 12th IEEE/ACM International Conference on Grid Computing (GRID), 2011*, pages 114 –120, sept. 2011. doi: 10.1109/Grid.2011.23.

Nina Tahmasebi, Gideon Zenz, Tereza Iofciu, and Thomas Risse. Terminology Evolution Module for Web Archives in the LiWA Context. In *Proceedings of the 10th International Web Archiving Workshop (IWAW'10) in conjunction with iPRES in Vienna, Austria*, 2010.

Gideon Zenz, Nina Tahmasebi, and Thomas Risse. Language Evolution On The Go. In *Proceedings of the 3rd International Workshop on Semantic Ambient Media Experience (NAMU Series) (SAME 2010) 10th-12th November 2010 in conjunction with AmI-10 in Malaga, Spain*, 2010.

Nina Tahmasebi, Sukriti Ramesh, and Thomas Risse. First Results on Detecting Term Evolutions. In *Proceedings of 9th International Web Archiving Workshop (IWAW'09) in conjunction with ECDL*, 2009.

The work in this thesis was partly funded by the European Commission under the projects ARCOMEM (ICT 270239) and LiWA (IST 216267).

Contributions and Thanks

I have many people to thank for their valuable help and insight in the work that contributed to this thesis. For the papers where I am first author, the main ideas, analysis and execution were mine. However, I still owe thanks to the following people.

- *On the Applicability of Word Sense Discrimination on 201 Years of Modern English*
  In this paper Mr. Kai Niklas contributed with the OCR error correction algorithm *OCR-Key* that resulted from his Master's thesis work. Mr. Gideon Zenz ran the large scale experiments on the Rrnz server cluster. I owe great thanks to both Mr. Niklas and Mr. Zenz, without their valuable contribution this work would not have been finished in time. We all owe great thanks to Mrs. Gertrud Erbach for her time and effort with fine tuning the text in this paper.
- *NEER: An Unsupervised Method for Named Entity Evolution Recognition*
  In this paper an initial seed of names that have undergone evolution where contributed by Dr. Nattiya Kanhabua. These names became the starting point of the testset used for the experiments. In addition, an index of the New York Times Corpus was provided by Dr. Kanhabua. I also owe great thanks to Mr. Gerhard Gossen for his valuable insights and discussions as well as his help with fine tuning of the text. The work performed on burst detection is fully contributed to Mr. Gossen.
- *Which Words Do You Remember? Temporal Properties of Language Use in Digital Archives*
  In this paper Mr. Gossen contributed immensely in the work of adapting the paper and fine tuning the language. Without his efforts this paper would not have been accepted for publication.
- *Terminology Evolution in Web Archiving: Open Issues*
  For the mathematical formulation of the model in this paper I owe thanks to Prof. Dr. Wolf-Tilo Balke whose insights and discussions made the model possible.

Across all my papers I owe great thanks to my mentor, Dr. Thomas Risse for his constant encouragement and generous time. He has been a integral part of all discussions leading up to the implementation of the ideas in this thesis. He has also taken a large part in the writing process and helped fine tune language and strategies in most papers.

Acknowledgments

Many people deserve thanks for their valuable contributions that made this thesis possible. From the person who gave me my first job to those who made the office a place for more than just work. A place that I have been happy to call home.

My first thanks goes to Prof. Wolfgang Nejdl who gave me the chance to work at L3S and who placed me in the best group possible. I owe thanks to Julien Gaugaz who informed me about the position at L3S, welcomed me to Hannover and for his friendship after that. A thanks to Wolf-Tilo Balke who made me feel right at home. To Eelco, George and Enrico with whom I shared offices. To Christopher Kunz who included me in his work and helped me when I needed it. To Kim, who helped me study for a class and then became a close friend. To Kim, Enrico and their lovely daughters who all have a special place in my heart. To Dimitar and the admin office that offered support for much more than computers and always had time to listen. To Claudia Niederée who is one of the nicest people I know and a great role model for women in science. To my students who taught me so much; Zhivko and Sallam who let me be a part of their lives and to Helge who will reach great heights. To Gerhard Gossen, without whom, I would most surely have gone mad. For all the discussions and conversations, the coffees and the lunches, I owe him great thanks! To Elena, for her friendliness and for brightening up every day with a smile.

My warmest thanks go to Thomas Risse, who was the best mentor I could ever have hoped for. He believed in me, helped me, supported me and pushed me. Without his support this thesis would not have been possible! I am eternally grateful.

I would like to thank Prof. Erich Neuhold for agreeing to be my second advisor and for his help in improving this thesis. I owe great thanks to him and all the professors at the On The Move Academy who inspired me and taught me so much.

Also a warm thanks to Anca Vais, for her help, patience and kindness and for always making me feel welcome.

My thanks go to Times Newspapers Limited for providing the archive of The Times for the research in this thesis. A special thanks to Gertrud Erbach for her kindness and patience.

To the people from the past, who are still my motivation. Peter Hegarty, Aila Särkää, Michael Patriksson, Patrik Albin, Hans Westergren, Ulla Digner, Lennart Falk, Tommy Gustafsson, Jan-Alve Svensson, Ivar Gustafsson and many more.

To Håkan Hansson, Mattias Widbom and Markus Wittebo who taught me the true meaning of *uthållighet* and *okuvlig anda*.

Finally, I owe the greatest thanks to my family who carried me here and supported me at every step of the way. To my brother to whom I wanted to prove that anything is possible. To my husband, who is the greatest person in my world and the brightest star on my sky, without whom, I would be lost. To my Mum and Dad, who worked so hard to change my fate and to give me a chance to write my own. I hope I have made you all proud.

This one is for you.

Contents

Chapter 1

Introduction

The need to understand and study our culture drives us to read books, explore Web sites and query search engines or encyclopedias for information and answers. Today, with the huge increase of historical documents being made available we have a unique opportunity to learn about the past, from the past itself. Using the collections of projects like Gutenberg (Gut) or Google books (Goo), we can access the historical source rather than read modern interpretations. Access is offered online and often minimal effort is required for searching and browsing. There is however a major limitation; unlike on the Web, where we are offered countless tools for information searching, when searching historical documents there are few resources available.

Throughout most of history, that which one generation knows has been different from the generation before. Because we seek to renew the existing and embrace the new and exciting, our language evolves. Where older people say *fine*, *modern*, *exciting* and *searching*, young people say *cool*, *awesome*, *sick* and *googling*. Therefore, there are no guarantees that what was written down many years ago and stored in our archives can be correctly interpreted today.

In the past, published and preserved content was stored in repositories such as national libraries and access was simplified with the help of librarians. These experts would read hundreds of books to help students, scholars or the interested public to find relevant information expressed using any language, modern or old. Today, because of the easy access to digital content, we are no longer limited to physical hard copies stored in one library. Instead, we can aggregate information and resources from any online repository stored at any location. The sheer volume of content prevents librarians from keeping up and thus there are no experts to help us to find and interpret information.

There are two major problems that we face when searching for information in long-term archives: firstly, *finding* content and secondly, *interpreting* that content. When things, locations and people have different names in the archives from those we are familiar with, we cannot find relevant documents by means of simple string matching techniques. The strings matching the modern names will not correspond to the strings matching the names stored in the archive. And, even if we are able to find relevant documents, there is no guarantee that we can interpret the content. Words and expressions reflect our culture and evolve over time. Without explicit knowledge about these changes, we risk placing modern meanings on these expressions which leads to wrong interpretations.

Currently, we must find missing links and information on our own. To aid us, the Web offers resources like Wikipedia and WordNet that compile and summarize a great deal of current and past knowledge. Information about important people, entities and events of the past can be found in these resources. However, most resources are typically general, limited in scope and

do not explicitly reveal historical changes. They rarely cover specific domains, do not cover ephemeral changes in language or jargon that was used in the past.

To help overcome these limitations, in this thesis we tackle the problem of automatically detecting language evolution. Our objectives are finding and interpreting content in long-term archival search. We therefore start by investigating different classes of language change with respect to these objectives. We then investigate models for describing each class. For two specific classes we investigate methods for automatically finding evolution to map current words and meanings to those of the past.

1.1 Goals of this thesis

In this thesis we focus on finding models and algorithms to describe and categorize language evolution from a *computational* point of view in an attempt to answer the question of *what* rather than *why*. We seek algorithms that can find and handle language evolution in such a way that *more information can be uncovered* and what is found *can be interpreted*.

Because of the vast amount of digitized information now available to us, we have a unique possibility to develop and test methods for detecting language evolution. However, the amount of data limits the possibility of using expert help and manual efforts. Therefore, we aim to find unsupervised, statistical methods that can be applied to any dataset without requiring external resources or manual input. As much as possible we aim to reuse algorithms and tools developed within the field of computational linguistics so that we have a sound base grounded in linguistic theory.

To maximize the chances of finding evidence of language evolution in our dataset we need large collections of text that span extended periods of time. To achieve this we have restricted the experiments in this thesis to two main corpora, *The Times Archive* (London) and the *New York Times Annotated Corpus*, hence forth referred to as *Times* and *NYTimes*. The Times is a large sample of modern English that spans 1785–1985, a long enough time span to guarantee word sense evolution. The collection is OCRed and offers a realistic setting with noisy, unstructured data. The NYTimes is a modern and error free collection with data from 1987–2007, and is used for comparison purposes as well as for finding evolution.

The primary goal of this thesis is to provide a deeper insight into the problem of language evolution regarding the finding and interpreting of content in long-term archives and to take the first steps towards fully automated methods of detecting language evolution. We target three main areas and our goals can be expressed as follows:

Language Evolution Model and Classification: To find a classification of language evolution with respect to the objectives of finding and interpreting content in long-term archives. To provide the relation between different classes, as well as the complexity of each class. Based on the requirements, we aim to find a model that can be used to represent different types of language evolution.

Word Sense Evolution: To investigate the properties of word sense evolution and, using these, find a suite of algorithms appropriate for automatically finding word sense evolution, and also to investigate the applicability of these algorithms to historical data. Furthermore, with the help of experiments, we aim to provide a proof-of-concept for automatic word sense evolution detection.

Named Entity Evolution: To find a method for detecting named entity evolution that overcomes limitations of previous work, by (1) eliminating dependencies on external resources; and (2) being applicable to all types of named entities, regardless of their context.

1.2 Outline of the Thesis

The structure of this thesis is as follows: In Chapter 2 we provide a motivation for our work, discuss our objectives, our use cases and perspectives taken in this thesis.

In Chapter 3 we present our definitions, terminology and our model for language evolution, entitled term concept graphs. Following that, we introduce our language evolution classification and highlight application areas for each class. We conclude the chapter by modeling of each class using term concept graphs.

In Chapter 4 we present a review of state-of-the-art in the fields and technologies related to word sense evolution as well as named entity evolution. We conclude the chapter by presenting a summary and motivation for our choices.

Our work on word sense evolution is split between two chapters. In Chapter 5 we cover our method for finding word senses. We present experimental results on Times (in both the original and OCR corrected version) and on NYTimes for comparison purposes. We evaluate the applicability of the algorithms to historical data. In Chapter 6 we analyze word sense evolution in detail, and present definitions of evolution that expand on linguistic theory. We also present experimental results for word sense evolution. As an example application we present a user interface, the *TeVo* - the *Terminology Evolution* browser.

Our work on named entity evolution is presented in Chapter 7 where we compare our method to state-of-the-art. As an example application, we present *fokas* - the *FOrmerly Known AS* search engine.

Chapter specific discussions, conclusions and contributions are provided at the end of Chapters 5 – 7. Finally, in Chapter 8 we conclude our work and present an outlook.

Chapter 2

Motivation

With advancements in technology, culture and high impact events our language changes. We invent new words, add or change meanings of existing words and change names of existing things. This results in a dynamic language that keeps up with our needs and provides us with the possibility to express ourselves and describe the world around us. The resulting phenomenon is called **language evolution**.[1]

For all contemporary use, language evolution is trivial as we are constantly made aware of the changes. At each point in time, we know the most current version of our language and, possibly, some older changes. However, our language does not carry a memory; words, expressions and meanings used in the past are forgotten over time. Thus, language evolution limits us when we want to find and interpret information about the past from historical documents. Formally, the problems caused by language evolution (illustrated in Figure 2.1) can be described with the following: Assume a long-term archive where each document d_i in the archive is written at some time t_i prior to current time t_{now}. The larger the time gap is between t_i and t_{now}, the more likely it is that current language has experienced evolution compared to the language used in document d_i. For each word w and its intended sense s_w at time t_i in d_i there are two possibilities; (1) the word can still be in use at time t_{now}; and (2) the word can be out of use (outdated) at time t_{now}.

Each of the above options opens up a range of possibilities that correspond to different types of language evolution that affect finding and interpreting in long-term archives. In this chapter we will discuss the types of evolution that are targeted in this thesis, other types of evolution are discussed in Chapter 3.

Word w at time t_i in use at t_{now}

> **No Evolution:** The word is in use at time t_{now} and has the *same sense* s_w and thus there has been no evolution for the word. The word and its sense are stable in the time interval $[t_i, t_{now}]$ and no action is necessary to understand the meaning of the word or to find content.
>
> **Word Sense Evolution:** The word is still in use at time t_{now} but with a *different sense* s'_w. The meaning of the word has changed, either to a completely new sense or to a sense that can be seen as an evolution of the sense at time t_i. The change occurred at some point in the interval (t_i, t_{now}). We consider this to be the manifestation of word sense evolution.

1 We use the term *language evolution* synonymous with *language change* in linguistics.

Word w from t_i out of use at t_{now}

Word Sense Evolution - Outdated Sense: The word is out of use because the word sense is outdated and the word is no longer needed in the language. This can follow as a consequence of, among others, technology, disease or occupations that are no longer present in our society. The word w as well as the associated word sense s_w have become outdated during the interval (t_i, t_{now}). To be able to interpret the word in a document from time t_i it becomes necessary to detect the active sense s_w at time t_i. Because it is necessary to recover a word sense that is not available at time t_{now} we consider this to be a case of word sense evolution.

Term to Term Evolution: The word w is outdated but the sense s_w is still active. Therefore, there must be another word w' with the same sense s_w that has replaced the word w. That means, different words, in this case w and w' are used as a representation for the sense s_w and the shift is made somewhere in the time interval (t_i, t_{now}). We consider this to be term to term evolution where the same sense is being represented by two different words.

2.1 Word Sense Evolution

When it comes to interpreting words that are found in historical documents we can make use of any available dictionary. Most dictionaries state all valid senses for a word and some include rare as well as outdated senses. However, only very few dictionaries keep time information attached to each word sense. The Oxford English Dictionary (Oxford University Press, 2000) is an example of such a dictionary where each word sense is accompanied by quotations from published work indicating the appearances of a particular sense for a given word. Most dictionaries do not carry this information, meaning that, even if all word senses are present in the dictionary, it is not known in which periods each sense was active. This leads to certain problems when interpreting documents, the following quote can be used to illustrate some problems that can arise.

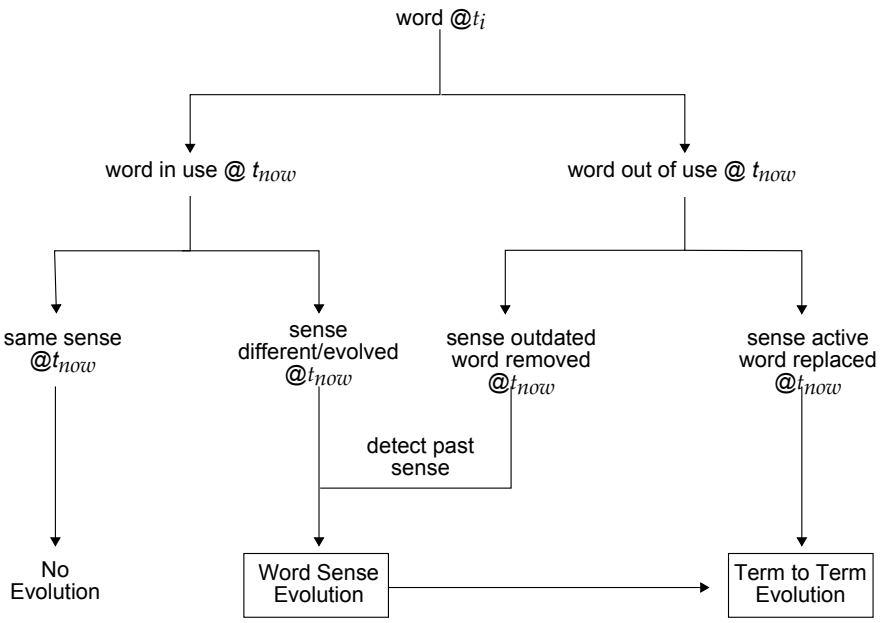


Figure 2.1: Diagram of Word Evolution

> “ Sestini's benefit last night at the Opera-House was overflowing with the fashionable and *gay*. ”

The above quote was published on April 27, 1787 in The Times (The Times, 1787) and carries a reference to a word that has since changed its primary meaning. When read today, the word *gay* will most likely be interpreted as *homosexual* because of the popularity of the sense today. However, this particular sense of the word was not introduced until the early 20th century and instead, in this context the word should be interpreted with the sense of *happy*.

The above is an example of word sense evolution. The word stays the same but is used in a different sense at different points in time. Removing the time component this problem is well known and referred to as *word sense disambiguation*. Navigli (2009) defines word sense disambiguation as the task of identifying the (intended) meaning of words in context in a computational manner. Given a short textual context, for example a sentence or a bag-of-words, identify the intended meaning of an ambiguous word in that context.[2] In the sentence *I like rock music.* the word *music* helps to disambiguate the word *rock* which otherwise could be interpreted in its *stone* sense. For humans, the task of disambiguating a word in a given context is natural; we do it on a regular basis and without much effort. However, for a computer the task is thought of as *AI complete* which is considered to be as hard as making computers as intelligent as people.

Word sense disambiguation contains two main difficulties. The first is disambiguation; a system must automatically make a decision regarding which sense that is most appropriate in the specified context. This can be viewed as a classification task; given a word w in a context and a set of senses $S = \{s_1, \ldots, s_n\}$, determine which sense $s_i \in S$ is the intended sense s_w.

The second difficulty lies in the senses. In the above classification task we assume that the set of senses S is given. However, it is far from straight forward to define the senses for a given word; The domain is highly relevant as well as the granularity of the senses. Terms can have vastly different meanings in different domains. Therefore, a general dictionary that is appropriate for one document might not be appropriate for another, both because of the domain as well as the granularity of the senses in the dictionary. If the senses cannot be found manually, like using manually created machine readable dictionaries, they must be extracted automatically. Automatic extraction can be applied to any collection of text to allow the senses to reflect the content in the collection rather than to be predefined. However, automatic extraction of word senses is a complex and difficult task. The alternative with manual creation of knowledge resources is unfortunately expensive and time consuming and must take the domain and the collection into account. Therefore, for many applications, we must use automatic sense extraction.

The task to automatically find word sense evolution builds upon automatically discovered word senses with the addition of time, senses cannot only be extracted once and assumed static. So for a given word w in a context, the set of senses S are dependent on the time t_i from which the word and the context stems. And, as we concluded above, there is limited time information available in existing knowledge resources and therefore we must rely on methods for automatic extraction of senses. Such methods are referred to as word sense discrimination and are described in detail in Chapter 4. We consider word sense evolution to consist of the following sub tasks.

1. Find word senses for each available time point.
2. Find the evolution of word senses over time.
3. Disambiguate between the word senses to attach the correct word sense to each instance of a word.

2 A bag-of-words is a set of unique words without the implied order or grammar that is present in e.g., a sentence.

Step 2 and 3 can be done in reverse order, but regardless of order, to find word sense evolution we must rely on word sense disambiguation which is considered to be AI complete. Therefore also word sense evolution can be considered AI complete and, thus, in this thesis we cannot hope to solve the full problem of word sense evolution. Instead we limit ourselves to (i) finding word senses with the help of word sense discrimination and (ii) determining which word senses that have evolved and how.[3] We focus on the senses and leave it as future work to apply word sense disambiguation in order to attach the correct senses to each instance of a word.

There are two different use cases for word sense evolution. The first one is to alert a user to a change in the word sense in a document compared to the modern word sense. More formally, the problem is the following; for any time t_i determine if word sense s_w corresponding to w has changed compared to time t_{now} for $t_i \leq t_{now}$. This problem contains comparisons only between two time periods and one decision: the sense s_w is changed or the sense s_w remained unchanged. An extension to this problem is to provide the word sense of w at time t_i. The second use case is to find all word sense evolution, that is, the full history of the term w. This task is more complex than the first one and requires $n-1$ comparisons to reconstruct all changes, where n is the number of time periods.

Sentiments

In many of the examples of word sense evolution, sentiments or the semantic orientation of words can be used as an indication of the sense change. The words *awesome* and *nice* are both terms that have changed their sentiment over time because both have gained a positive sentiment. The latter is no longer used with a negative sentiment while *awesome* can still be used to express a negative sentiment. However, using only sentiments is not enough to help users interpret a historical mentioning of these words. By assigning both negative sentiment to both *awesome* and *nice* at one point in time and then later assigning a positive sentiment, we can lead the user to understand that something has happened. However, we cannot use the semantic orientation to fully express *what* has happened. We can only highlight that both have become more positive in sentiment. To fully express the change we need to find and express the word senses over time and thus word sense evolution is more appropriate and complete for our purposes than using sentiment analysis to find sentiment evolution.

2.2 Term to Term Evolution

Term to term evolution considers *different words* used to express or reference *the same entities or concepts* at different periods in time. This means that if a word w is found in a document at time t_i with the sense s_w there is a modern replacement for w, considered as w', at time t_{now} with the same sense s_w. For some words in this class we can make use of dictionaries and knowledge resources to find w', however, we run in to the same problems with modern resources as for word sense evolution. Lack of time information and coverage results in a need to automatically find these term to term evolutions. Some of the difficulties in this class are illustrated by the following quote.

> “The Germans are brought nearer to *Stalingrad* and the command of the lower Volga.”

The above quote was published on July 18, 1942 in The Times (The Times, 1942) and refers to the Russian city that figures often in the context of World War II. In reference to World War II

3 We consider only nouns and noun phrases as current word sense discrimination techniques primarily cover these word classes. More on this in Chapter 4.

people speak of *the city of Stalingrad* or the *Battle of Stalingrad*, however, the city cannot be found on a modern map. In 1961, *Stalingrad* was renamed to *Volgograd* and has since been replaced on maps and in modern resources. Not knowing of this change leads to several problems. Knowing only about *Volgograd* means that the history of the city becomes inaccessible as documents that describe its history only contain the name *Stalingrad* (or *Tsaritsyn* as the city was named before 1925). Reversely, knowing only about *Stalingrad* makes is difficult to find information about the current state and location of the city.

The above was an example of a special case of term to term evolution, namely named entity evolution. However, the class is broader and contains words from all part-of-speech. In the quote from Chapter 2.1 the words *gay* and *happy* can be considered evolutions of each other because they have been used to express the same feeling at different points in time. Detecting this type of term to term evolution is considered significantly more difficult than detecting named entity evolution.

The main difficulty with term to term evolution, analog to that for detecting word sense evolution, regards the definition and disambiguation of word senses. Detecting that w and w' are used at different points in time to refer to the same sense s_w requires first finding the set of all senses S and then disambiguating between all possible senses to determine $s_w \in S$. For named entities, defining the senses can be reduced to representing an entity, a task which we consider easier and better defined. People, companies and places can be defined by means of, for example, geographic location or full name and birth date. Also named entities must be disambiguated as there, for example, can be several different persons with the same name and the same birth date.

A second difficulty is that term to term evolution can occur in combination with word sense evolution. In the case of *Stalingrad*, the name of the city is out of use and when used, it refers only to the same entity. However, all words that experience term to term evolution need not necessarily be out of use. A word w can still be in use and experience term to term evolution and for this to happen, w must have experienced word sense evolution as well.

We illustrate the above claim with the following example: assume that a word w is in use and has been replaced by a word w' without experiencing a change in word sense. This would mean that there are two words in the language that refer to the same sense at the same point in time. We consider two such words to be synonyms. Thus, in order for word w and w' to be term to term evolutions, the word w must have experienced word sense evolution and exist in the language with another word sense. A concrete example of a word in this class is the word *nice*. The word is currently in use and today it would be interpreted as *considerate* or *kind in behavior* and is mainly seen as a positive adjective. However, during the 13th century the word *nice* would be interpreted as *foolish, silly* and *simple*. We conclude that the word has experienced word sense evolution; the sense of *foolish* is no longer valid for the word *nice*. Today, we would use the word *silly* to express the same sense as the word *nice* had in the 13th century. Therefore, we can consider *silly* to be a term to term evolution of the word *nice* in the sense of *foolish* while the word *nice* is still present in the language with the sense of *kind*.

Because term to term evolution consists of defining and disambiguating word senses and can be coupled with word sense evolution, we consider the class of term to term evolutions to be AI complete. However, for named entities, term evolutions are better defined; in many domains, references to named entities are unique and entities are easier to define than word senses. Also, named entities do not experience word sense evolution and therefore, on average, we consider named entity evolution to have a lower complexity than the more general term to term evolution. For this reason, in this thesis we focus on automatic detecting of named entity evolution and leave term to term evolution as future work.

2.3 Historical Data vs. Modern Data

When working with language evolution from a computational point of view there are two main perspectives available. The first considers today as the point of reference and searches for all types of language evolution that has occurred until today. In this perspective the language that we have today is considered as common knowledge and understanding past language and knowledge is the primary goal.

In the second perspective the goal is to prepare today's language and knowledge for interpretation in the future. We monitor the language for changes and incrementally map each change to what we know today. We can assume that knowledge banks and language resources are available and all new changes are added to the resources. In the next paragraphs we will discuss the differences between the two perspectives in detail.

2.3.1 Looking to the Past – The Backward Perspective

When looking to the past we assume that we have the following scenario. A user is accessing a long-term archive and wants to find and interpret information from the past. There are several problems which the user must face. Firstly, there are few or no machine readable dictionaries or other resources like Wikipedia, which sufficiently cover language of the past. The user must rely on his or her own knowledge or search extensively in other resources like encyclopedias or the Web in order to find an appropriate reformulation for modern words. Once the resource is found the user must repeat the process to find the meanings of words, phrases and names in the document. Because of the low coverage of the past, the user can find only limited amount of help in this process.

In order to help users in their research of the past we need to automatically find and handle language evolution. This can be done by making use of existing algorithms and tools or by developing new ones. For both existing and new tools there are severe limitations caused by the lack of digital, high quality, long-term collections. Most existing tools have been designed and trained on modern collections and can have difficulty with problems caused by language evolution. For example, part-of-speech tagging, lemmatization and entity recognition can be affected by the age of the collection and thus limit the accuracy and coverage of language evolution detection which relies on the mentioned technologies.

There is much work being done currently to overcome this lack of resources by digitizing historical documents by means of optical character recognition (OCR). However, many older collections have been stored for a long time which leads to less than perfect quality of the resulting text. Degraded paper, wear or damage as well as old fonts cause errors in the OCR process. This leads to problems in the processing, for example to detect word boundaries or to recognize characters, as well as to verify the results. If words cannot be understood by humans then it is next to impossible to judge the correctness of the algorithms. Because of the historical nature of the language, it is also difficult to find people that are qualified to verify, improve or help detect language evolution on such collections.

2.3.2 Looking to the Future – The Forward Perspective

When looking to the future to find language evolution we have many advantages compared to when looking to the past. The largest advantage is that most resources are born digitally today and thus many of the problems with degraded paper quality and OCR errors are avoided. In addition, there is an abundance of available data. Most concepts, senses and entities are described and referenced over and over again which makes it easier to gather evidence for each one individually.

Table 2.1: Processing Comparison - Looking to the Past and Future

Aspect	Past	Future
Content	Digitized after creation, risk of decreased quality.	Increasingly born digital no need for digitization.
Resources	Limited availability	Increasing availability, WordNet, LinkedData etc.
Tools	Mostly modern tools few specialized NLP tools	Existing tools, will be continuously updated
Quality	OCR errors, outdated terms	Noise in user generated text, abbreviations, slang
Crowd sourcing	Limited possibility requires experts	Possible to make use of crowd sourcing

In addition to the higher amount and quality of the text, there are plenty of tools and resources available that can solve many smaller tasks automatically. Natural language processing tools, machine readable dictionaries, and encyclopedias form an army of resources which can be used to tackle current language. Changes in our world are captured and questions like *What is the new name of the city XYZ?* can be answered using machine readable resources like Yago (Suchanek et al., 2007) or DBpedia (Bizer et al., 2009). To prevent information loss in the future, resources like Wikipedia, WordNet and natural language processing tools can be stored alongside the archives and can significantly simplify finding and verifying language evolution in the future.

In the perspective of looking to the future we assume that current language is common knowledge and therefore we can employ humans to help detect language evolution. *Crowd sourcing* (Howe, 2006) is collaborative work performed by large amounts of people and is the mechanism behind creating and maintaining Wikipedia. Such mechanisms could be used to monitor language and detect evolution. If models for representing and storing language evolution are provided, crowd sourcing could be used to detect language evolution manually or to verify automatically detected language evolution.

There are however several limitations. The first limitation is noisy data being published on the Web. With increasing amounts of user generated text and lack of editorial control, there are increasing problems with grammars, misspellings, abbreviations, etc. To which level this can be considered as real noise like with OCR errors is debatable, however, it is clear that this noise reduces the efficiency of tools and algorithms available today. This in turn limits the quality of evolution detection as we depend on existing tools and their efficiency. The second limitation is the restricted nature of resources like Wikipedia. As with dictionaries, Wikipedia does not cover all entities, events and words that exist. Instead, much is left out or only mentioned briefly which limits to which extent we can depend exclusively on these resources.

In this thesis we take the backwards perspective and introduce novel methods for finding and interpreting, allowing the users to utilize the potential of current digitization efforts.

2.4 Problem Statement and Contributions

Much of our culture and history is stored in the form of written records. Today, more and more effort and resources are spent digitizing and making available these historical resources that were previously available only as physical hard copies. However, making the resources available to the users has little value in itself; the broad public cannot fully understand or utilize

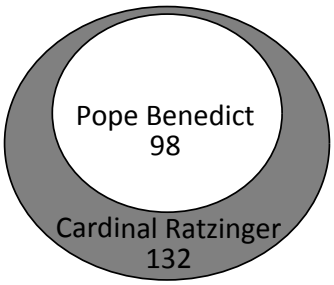


Figure 2.2: Search in NYTimes for Cardinal Ratzinger and Pope Benedict.

the content because the language used in the resources has changed over time. Instead, this vast pool of knowledge should be made semantically accessible and interpretable for the broad public to gain full utility. Our aim is to help the users find and understand the content and thus be able to explore their history and culture. Modern words should be "translated" into their historical counterparts and words should be represented with their historical meanings and senses. The full utilization of language evolution, for example by means of information retrieval technologies, temporal indexing techniques and visualization, is left as future work.

Finding

Users are interested in past entities or events and would like to learn about them from the original resources, for example by means of the first publication in the matter. The majority of people that are interested in the past are everyday people without in-depth knowledge of the language used at that time. In addition to everyday people, scholars are interested in certain historical aspects of, for example, politics, art or technology without being experts also in the corresponding language.

We illustrate the problem of finding content with an example shown in Figure 2.2. A search in the New York Times in 2005 for the term *Cardinal Ratzinger* results in 132 documents. On 98 of these documents, the name *Pope Benedict* is present. This means that without explicitly knowing that *Joseph Ratzinger*, and as an extension *Cardinal Ratzinger*, is the birth name of *Pope Benedict*, 34 documents about the earlier life of the Pope cannot be found. Thus presenting the user with query reformulation options or automatically adding previous names of entities to a query would help users find information that might otherwise be lost to them.[4]

In this thesis we take the first steps towards helping users find information in long-term archives by proposing a method that automatically finds different names used for the same entity over time. Information retrieval systems that utilize these different names to expand queries do not fit in the scope of this thesis and are left as future work.

Understanding

In addition to finding more content, users can benefit from automatically detected language evolution with regards to the meanings of words. When reading a historical document the user can be warned that words had different meanings during the time when the document was written and, thus, should not be interpreted with their modern sense.

4 After resigning the papacy in March 2013, the pope will officially be known as *His Holiness Benedict XVI, Pope emeritus.*

General Scope

In this thesis we set out to classify and model language evolution and to find word sense and named entity evolution in an automatic way. We take a historical perspective and look to the past to find evolution. We aim to reuse as many existing tools and algorithms as possible to build upon existing knowledge and experience. To find algorithms that are general and domain independent we choose to rely only upon the dataset in which we wish to find evolution. For this reason external resources like dictionaries or ontologies are excluded and because of the enormous amounts of textual resources available, and the heterogeneity of those resources, we also exclude human input. We aim to provide algorithms that serve as proof-of-concepts and thus largely ignore scalability and complexity issues. To try our algorithms we use newspaper corpora as they are (1) available and with long enough time span; (2) are time stamped; and (3) use a high quality, normalized language. To limit the scope of this thesis, we concentrate our work on nouns and noun phrases because there are existing word sense discrimination algorithms for these word classes. We exclude visualization of evolution, disambiguation, searching using discovered instances of evolution and data from other sources than newspapers, for example Web content, and leave this as future work.

2.4.1 Contributions

The main contributions presented in this thesis are:

- A **classification for language evolution** that is based on the objectives of finding and interpreting of content in long-term archives. We introduce **term concept graphs** as the relation between terms and their concepts (word senses or meanings) or contexts and show how different types of evolution can be modeled using term concept graphs. (Chapter 3).
- Investigation of the **applicability of word sense discrimination on historical text**. We verify that the outcome of word sense discrimination indeed corresponds to word senses and thus can be used as a basis for detecting evolution. We show that word sense evolution can be captured by the automatically extracted word senses. (Chapter 5).
- A **definition of word sense evolution** by means of term concept graphs and an algorithm to identify word sense evolution by merging term concept graphs and grouping word senses into concepts. The algorithm has the potential to capture broadening, narrowing, splitting and merging of senses in addition to polysemy and homonymy. The method is unsupervised, independent of external resources and takes the first steps towards fully **automatic word sense evolution detection**. (Chapter 6).
- Methodology to **identify named entity evolution** by analyzing the context of entities during time periods of evolution. The proposed method is independent from external knowledge sources and is able to find name changes without requiring re-current computation. (Chapter 7).
- A testset with all name changes used in Chapter 7, as well as an extended version, to encourage further research and comparison of results in the area of named entity evolution.

Chapter 3

Language Evolution Model and Classification

Within the linguistic field, the studies performed in this thesis are considered as **language change** rather than **language evolution**. In this thesis however, we define language evolution as synonymous to language change and choose the former term. The linguistic field of language evolution is broad and contains many different sciences and fields. Long-term evolutions beyond one language or the evolution of one language into another lie outside of the scope of this thesis. Instead, we focus our attention on word sense evolution and named entity evolution within one language.

We present a graph model for describing the relation between terms and their meanings called **term concept graphs**. These graphs constitute the building blocks for all types of evolution. We define four major classes of language evolution and, using term concept graphs, we show how each class can be modeled. In addition, for each class we outline application areas.

3.1 Definitions and Terminology

In this section we provide definitions and explain the terminology that we use throughout this thesis. All introduced terms will be used with and without *temporal* interchangeably. More specific definitions are given in the chapters where they are needed.

3.1.1 General

We begin with the general definitions needed for both word sense evolution as well as named entity evolution.

We consider a **term** w to be a single or multi-word lexical representation of a noun or noun phrase. The work in this thesis is limited on nouns and noun phrases and therefore we limit our definition of terms to only include this part-of-speech.

A **set** is a collection of objects such that each object is distinct and well defined. The set is in it self considered an object. A cluster is considered to be a set because all terms are unique. There are two set operations that we make use of in this thesis, *set union* denoted $\cup$ and *set intersection* denoted $\cap$. Set intersection refers to all elements in set A that are also present in set B, shown as the overlapping part in Figure 3.1. The union refers to all elements in either A or B and because sets consider only unique elements, $A \cup B$ can be seen as $A + B - (A \cap B)$.

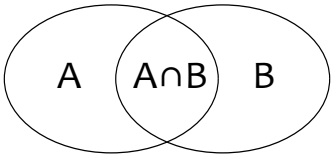


Figure 3.1: Two sets A and B. The overlapping elements are in $A \cap B$.

A **cluster** is a set of items such that each item in the cluster is more similar to the other items in the cluster than to items outside of the cluster. Similarity between items is measured, for example, by means of cosine similarity (Eq. 3.3). In our case a cluster consists of a set of terms such that the terms represent a word sense or a named entity. Clusters are found automatically using unsupervised clustering algorithms which do not require human input or interaction.

We consider **term co-occurrence** if terms are found in close proximity of each other or are grammatically related within a given context, e.g., sentence, paragraph or whole document. In this thesis we consider two different methods for finding term co-occurrences: *sliding window* and *grammatical*, both considered to be first order co-occurrences. A sliding window method considers all terms within a distance of k terms to be co-occurring. A grammatical method considers all terms that have a grammatical relation to be co-occurring. Example of grammatical relations are Hearst patterns (Hearst, 1992) like *such as*. In the sentence "A fruit such as an apple or a banana..." the grammatical method would consider following term co-occurrences (*fruit*, *apple*) and (*fruit*,*banana*). The sliding window method would, in addition, also consider (*apple*, *banana*).

We choose a **co-occurrence matrix** to represent term co-occurrences in a document collection D. The matrix M is an $N \times N$ matrix where N is equal to the number of terms in the vocabulary W_D corresponding to D. Each entry $m_{ij} \in M$ represents the frequency with which terms w_i and w_j co-occur in D. Each row M_i can be seen as a **feature vector** corresponding to term w_i. In Equation 3.1 we consider the first row to corresponding to the term *apple*. Each entry in the row corresponds to one term in the vocabulary and the entry is larger than zero only if the corresponding term co-occurs with *apple* in the collection. If entry two corresponds to the term *banana*, the entry m_{12} would have the value $fr_{(\text{apple, banana})}$.

$$\mathbf{M} = \begin{pmatrix} m_{11} & m_{12} & \dots \\ m_{21} & m_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \tag{3.1}$$

A co-occurrence matrix can also be viewed as a **co-occurrence graph** G where each node n represents a term from the vocabulary W_D. There exists an edge between two nodes if the corresponding terms co-occur in D. Edge weights can be considered as binary; 1 if the terms co-occur and 0 otherwise. It is also possible to let the edge weight correspond to the frequency with which terms co-occur, either in absolute values or as a normalized frequency. Figure 3.2 shows a minimal graph with the nodes *apple* and *banana*.

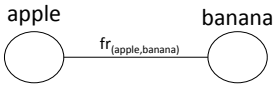


Figure 3.2: Co-occurrence graph with two nodes: *apple* and *banana* and edge weight corresponding to the co-occurrence frequency of the terms.

The **Jaccard similarity** of two sets of terms A and B is considered to be the number of terms that exist in both sets, out of all unique terms, defined formally as:

$$\texttt{JaccardSim}(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\#\text{terms in both } A \text{ and } B}{\#\text{unique terms in } A \text{ and } B} \tag{3.2}$$

Jaccard similarity can be used to measure similarity between clusters. The terms in each cluster constitute a set and the similarity between two clusters becomes the Jaccard similarity between the term sets.

There are different ways to measure relatedness or closeness of terms. Firstly, two terms w_i and w_j can be considered similar if they co-occur often. This would mean that the terms *car* and *tire* are similar because they often co-occur. This similarity can be found directly by means of the co-occurrence matrix.

Secondly, two terms can be considered similar if they co-occur with similar terms. The terms *car* and *truck* would be similar because they both co-occur with terms like *tire*, *motor* and *vehicle*. This similarity can be found by measuring the similarity between the feature vectors (i.e., rows in the co-occurrence matrix) corresponding to *car* and *truck*, for example by means of Jaccard similarity. In this case, we consider the feature vector corresponding to two terms w_i and w_j. All terms corresponding to non zero elements are placed in the respective set S_{w_i} and S_{w_j} and the Jaccard similarity is then used directly on the sets.

An alternative method for measuring similarity between terms is **cosine similarity**. Each term w_i and w_j is represented by its feature vector M_i and M_j and similarity is measured as the angle between the feature vectors. The higher the similarity, the smaller the angle.

$$\texttt{CosineSim}(M_i, M_j) = \frac{M_i \cdot M_j}{\|M_i\| \cdot \|M_j\|} = \frac{\sum_{k=1}^{n} m_{ik} \cdot m_{jk}}{\sqrt{\sum_{k=1}^{n} (m_{ik})^2} \cdot \sqrt{\sum_{k=1}^{n} (m_{jk})^2}} \tag{3.3}$$

3.1.2 Word Sense Evolution

To find word sense evolution we make use of **word sense discrimination** algorithms that are clustering algorithms used to automatically derive word senses present in a document collection. Thus far, discrimination algorithms have focused on nouns and noun phrases and we can therefore extend the general definitions to word sense evolution.

A **word sense** can be approximated using a cluster of terms such that these terms represent one sense or meaning of a term. Each sense corresponds to the collection that is used for clustering and carries the same notion of time as the collection. Throughout this thesis we use the terms *sense* and *cluster* interchangeably to refer to approximations of word senses by means of word sense discrimination.

A **concept** represents one or several related word senses. An in-depth discussion on the relation between linguistic concepts, meanings and word senses can be found in Chapter 6.1.

When a term has several senses, these senses can be related or unrelated. If the senses are related, the term is called **polysem**. If however the senses are unrelated, the word is called **homonym**. Example of polysemy is the term *foot* which can mean the foot of a human or the foot of a mountain. Because both of these senses refer to the bottom part of something, the senses are considered related. Example of homonymy is the term *rock* where the senses *stone* and *music* are completely unrelated.

3.1.3 Named Entity Evolution

We consider a **named entity** to be the name of a person, organization or geographic location. A name is what differs one entity from other entities with similar characteristics (Grishman and Sundheim, 1996).

We use contexts to describe a named entity. The **context** c_w of a term w are all terms related to w at time t_i. Similar to Berberich et al. (2009) we consider the context to be all co-occurring terms within a window of size k, however, other contexts can be used.

Co-references are expressions that refer to the same entity. In the sentence "The president said he had discussed the issue" the words *the president* and *he* refer to the same person. In this thesis, we consider **temporal co-references** to be different lexical representations that have been used to reference the same entity at different periods in time.

We have two types of temporal co-references, direct and indirect. **Direct temporal co-references** are temporal co-references that are variations of each other with some lexical overlap. **Indirect temporal co-references** are temporal co-references that lack lexical overlap on token level. For example, *Hillary Clinton* and *Hillary Rodham* are direct temporal co-references while *Pope Benedict XVI* and *Joseph Ratzinger* are indirect temporal co-references.

A **temporal co-reference class** contains all direct temporal co-references for a given named entity and is denoted

$$\text{coref}_r\{w_1, w_2, \ldots\} \ .$$

Each temporal co-reference class is represented by a class representative r that is also a member of the class. E.g., *Joseph Ratzinger* is the representative of the co-reference class containing the terms {*Joseph Ratzinger, Cardinal Ratzinger, Cardinal Joseph Ratzinger, ...*}.

We consider a **change period** p_i to be a period in time in which one term evolves into another. Change periods can have different granularity, e.g., days, weeks, months and years, and in this thesis we choose change periods corresponding to yearly periods. We consider two consecutive periods as different periods and allow gaps, e.g., $p_1 = 1.1.1998 - 31.12.1998$, $p_2 = 1.1.1999 - 31.12.1999$ and $p_3 = 1.1.2005 - 31.12.2005$.

We denote term to term evolutions with $w_i \rightarrow w_j$ where w_i and w_j are temporal co-references. The denotation can be extended to cover also change periods $w_i \underset{t_i}{\longrightarrow} w_j$ or validity periods $w_i \underset{t_i,t_j}{\longrightarrow} w_j$ where t_i and t_j correspond to change periods. The period (t_i, t_j) corresponds to a validity period for the term w_i.

3.1.4 Measuring Quality

We use precision and recall, where possible, to measure the quality of our results. Here we will give a general definition of these measures, more detailed and adapted definitions are given in the chapters where needed.

We consider A to be a set of elements resulting from an algorithm and set B to be the ground truth corresponding to all existing correct elements. **Precision** is measured as the proportion of elements found in A that are correct, i.e., also present in B. **Recall** is measured as the proportion of correct elements from B captured by A. More formally we define

$$\texttt{precision} = \frac{|A \cap B|}{|A|} \tag{3.4a}$$

$$\texttt{recall} = \frac{|A \cap B|}{|B|} \tag{3.4b}$$

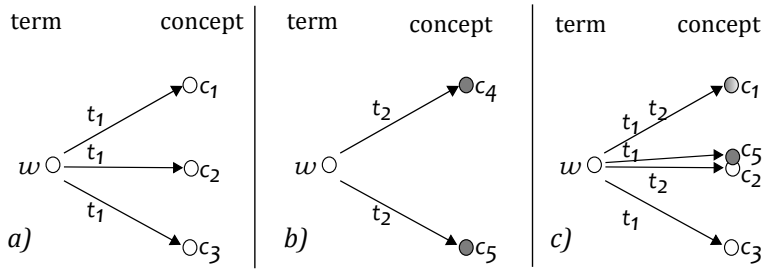


Figure 3.3: Merging two TCG's from time t_1 (a) and t_2 (b). The merged graph is shown in (c). Concepts c_1 and c_4 are equal and represented by c_1 in (c). Concepts c_2 and c_5 are similar but not the same.

3.2 Term Concept Graphs

In this section we consider word senses to be described using concepts, a more detailed description is provided in Chapter 6.1. Named entities are described using contexts. In order to model the relation between terms and their concepts or contexts we introduce **term concept graphs** (TCG). A TCG consists of term-concept or term-context pairs (edges in the graph) found in a collection. The graph carries a sense of time by annotating each edge with the time stamp inherited from the collection. Polysemy and homonymy can be represented by annotating two pairs with the same time period. Let W be the complete universe of terms, T the universe of time stamps and C the universe of all concepts and contexts. Then ϕ is the function that creates a term concept graph and maps a term to all its concepts or contexts. Formally we define ϕ as:

$$\phi : W \times T \rightarrow (W \times \mathcal{P}(\mathcal{C} \times \mathcal{P}(T))) \tag{3.5}$$

$$(w, t) \mapsto (w, \{(c_1, \{t\}), \ldots, (c_n, \{t\})\})$$

where $w \in W$, $t \in T$ and for all $i = 1 \ldots n$: $c_i \in C$. Here $\mathcal{P}$ denotes a power set, i.e. the set of all subsets. Although ϕ generates only one time stamp for each edge, we introduce the power set already here to simplify the next steps. We call the set of all TCGs derived using a collection from a certain time period a *terminology snapshot*. Once several terminology snapshots have been constructed, these will need to be compared in order to reconstruct the full history of a term. The comparison is made by merging two TCGs using a merging function and outputting a new TCG. The newly constructed TCG carries the gathered information of the merged TCGs by allowing for several time annotations on each edge. To shorten the notation we define τ as a set of time stamps t_i, i.e. $\tau \in \mathcal{P}(T)$ and a term concept relation can be written as a pair (c_i, τ_i).

The main difficulty in the merging step comes from decisions concerning two similar concepts (analoge with contexts), see Figure 3.3. When two concepts are the same as in the case with c_1 and c_4 the merging is trivial. In the merged TCG, Figure 3.3c), the term w is associated with one representative shown as c_1 with both time annotations. When a concept is completely different like c_3 it is present in the merged TCG with the same time annotations as in the original TCG. The difficulty arises when two concepts are similar but not equal, like with c_2 and c_5. In this case a decision must be made whether c_2 and c_5 should be merged, and if merged, how they should be represented. The function representing the merging step can more formally be described by the following:

$$\psi : (W \times \mathcal{P}(\mathcal{C} \times \tau)) \times (W \times \mathcal{P}(\mathcal{C} \times \tau)) \rightarrow (W \times \mathcal{P}(\mathcal{C} \times \tau)) \tag{3.6}$$

$$((w, \{(c_1, \{t\}), \ldots, (c_i, \{t\})\}), (w, \{(c_j, \tau_j), \ldots, (c_m, \tau_m)\})) \\ \mapsto (w, \{(c'_1, \tau'_1), \ldots, (c'_n, \tau'_n)\})$$

where $c_i, c'_j \in \mathcal{C}, t \in T$ and $\tau_i, \tau'_j \in \tau$ for all i, j. It should be clear that the set of concepts c' in the resulting graph of ψ does not necessarily have to be a subset of the concepts $\{c_1, \ldots c_m\}$ from the input graphs. For example, in Figure 3.3, the concepts c_2 and c_5 could be merged and considered as a new concept. ψ can be iteratively applied to a TCG from time t_N and the TCG containing all knowledge about a term up to time t_{N-1}.

3.3 Language Evolution Classification

Language evolution is a broad concept which can be divided into several sub-classes as shown in Figure 3.4. We consider four main classes, namely **spelling variations**, **word sense evolution**, **term to term evolution** and **general language evolution**. Furthermore we consider **named entity evolution** as a special case of term to term evolution. Because terms can experience different types of evolution we consider the classification as a **soft classification** meaning that a term can be placed in several classes. So far, spelling variations is the class that has received most attention by researchers and is the class that is best understood from a computational point of view. The work has focused on developing automatic/semi-automatic methods using rules as well as machine learning for creating dictionaries and mappings of outdated spelling. These resources are then used for search in historical archives to avoid missing out on important information.

The remaining three classes have received much less attention by the computer science and computational linguistic communities. To some extent this is because, up to now, there have been few available digital resources that span longer periods of time. While some types of evolution can develop during short time periods, other types take longer to develop and thus require long spanning datasets in order to research the problem. Furthermore, the remaining classes are more difficult to detect and have less direct uses in fields like information retrieval. Still there has been some work in named entity evolution focusing on finding *query reformulations* (Berberich et al., 2009) or *time based synonyms* (Kanhabua and Nørvåg, 2010) as well as studying word sense evolution (Sagi et al., 2009; Bamman and Crane, 2011).

The classification presented in this thesis focuses on how each class of evolution can be described and modeled. Depending on which class the problem falls under there are also different methods for finding contexts or concepts and different mapping and merging functions.

Spelling Variations The simplest class of evolution considers **spelling variations** of terms which refer to different spellings for the same term without any changes in meaning (Ernst-Gerlach and Fuhr, 2007; Hauser et al., 2007; Pilz et al., 2006). Spelling variations can be used during the same time period, e.g., *z* used instead of *s*, but can also be used differently over different periods in time. For example, *infynyt, infinit, infinyte, infynit, infineit* are all historical spelling variations used at different times for the word *infinite* (Oxford University Press, 2000). Discovering spelling variations, in particular for historical texts, can significantly improve retrieval results as spelling was historically less uniform than it is today (Gotscharek et al., 2009b).

Word Sense Evolution The second class of language evolution is **word sense evolution** which consists of all terms that have changed their *meaning* over time. Examples for this class are terms that have added or lost senses over time: *Rock* added the sense *music* to its previous

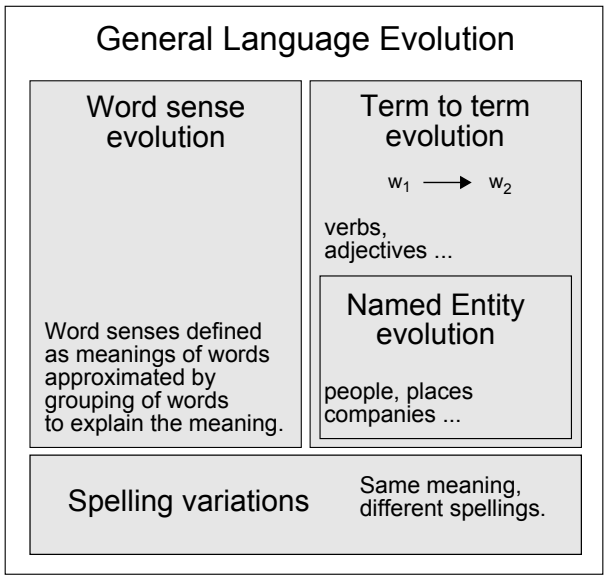


Figure 3.4: Language evolution classification.

sense *stone*. However, it is also possible for a term to completely change its dominant sense: *nice* meant *foolish* in the 14th century but is used to express a *friendly* sentiment today. In this class concepts are used to represent word senses.

Term to Term Evolution The third class of evolution is **term to term evolution** which contains terms that meant the same thing at different points in time, e.g., one sense of the term *cool* was previously expressed with *collected*. This class can contain terms w of any part-of-speech and the shift between terms typically occurs over a long period of time.

A special case of term to term evolution is **named entity evolution** which considers a given entity and different lexical names for the same entity. Here, the entity is fixed while the name changes over time. Typically, there are no slow shifts from the old term to the new term, instead name changes occur over a short period of time and are often caused by events like marriage, role change or company merger.

Unlike the terms that have evolved only with spelling variations, named entity changes do not need to have any lexical or phonetic overlap between two names. For example *Joseph Ratzinger* was the birth name of *Pope Benedict XVI* and *Hillary Rodham* was *Hillary Clinton*'s maiden name. The latter is an easier case of evolution because of the overlapping surname and can be targeted using entity consolidation or linking techniques like those presented by Shen et al. (2012) or Ioannou et al. (2010). However, most existing techniques do not take historical changes into account and only focus on merging different concurrent representations of the same entity. In this class contexts are used to represent an entity and concepts for all other terms. Because of changes that lack lexical similarities like *Joseph Ratzinger* and *Pope Benedict* we consider term to term evolution a more complex case of language evolution than spelling variations.

General Language Evolution The fourth and final class of evolution is **general language evolution** which considers all of the above classes. When a term w_i at time t_i slowly shifts to w_j at time t_j, all words used to explain w_i and w_j have also changed. Thus the contexts or concepts of the terms cannot be used directly for comparison. It should be noted that most term to term evolutions are to some extent accompanied by a concept shift, e.g., a new position, a company merger or being elected pope, which all mean that the context of a term or

entity changes. However, only when the term to term evolution is accompanied by a significant concept shift it is classified as general language evolution. Because no part of the term stays stable, this class is considered the most complex class of language evolution. The shift between *Walkman* and the modern version of it, the *iPod*, belongs to this class. The overall concept of *portable music player* stays the same but most terms used to describe the two differ significantly (see Chapter 7).

Orthogonal Classes In addition to the classes presented above, language evolution can be further classified into **preservative** and **destructive**. The first class concerns terms that have changed but can still be used in text. Examples for preservative evolution contain names of people or places where the previous names can be found in current texts to describe a time before the name change took place. An example is the name *Stalingrad* which is used in the context of WWII instead of the modern counterpart *Volgograd*. The latter class concerns types of evolution that are unlikely to return after having changed. Example of this is different spellings of terms like *Teutschland* for *Deutschland*.

3.3.1 Application Areas

The classes represented in our language evolution classification have different applications with respect to finding and interpreting in long-term archives. Though none of the classes can be said to belong exclusively to one of these objectives, some classes are more important to one or the other. Spelling variations as well as named entity evolution work primarily towards finding content. Knowing the different spellings of a term or the different names of an entity can help find content that would otherwise be lost to the user that lacks explicit knowledge of all variations. For example, finding different names for the city of *St. Petersburg* will allow the user to find documents about the city during the periods when the city was named *Petrograd* and *Leningrad*. Of course, these classes also contribute towards interpreting. For example, not knowing why documents containing the term *Leningrad* are presented as result to the query *St. Petersburg* will prevent the user from accepting these results as correct.

Term to term evolution as well as word sense evolution work primarily towards interpreting and understanding of documents that are found in long-term archives. Word sense evolution allows the user to find the meanings of terms when the terms have a different meaning than what the user would expect. This class also helps users to find the meaning of a term that is outdated at the time of reading and therefore cannot be found in a common dictionary. Term to term evolution e.g., for adjectives, help the user to map terms to their modern representatives. The word *nice* from the 13th century can be mapped to *foolish* to help the user correctly understand the meaning of the word in a historical document.

Word sense evolution and term to term evolution can work together to help interpretation. If a term w is found in document d_i from time t_i and the term has a different sense at time t_i compared to t_{now} then we can choose one of the following options. (1) Present the user with sense s_w corresponding to word w at time t_i or if possible; and (2) present the user with the corresponding term to term evolution w_{now} which shares the sense s_w and is a modern representative of w.

General language evolution works towards both finding and interpreting. Often the user wants to find a historical representative of the concept that they have in mind. This covers modern things like music players (e.g., *Ipod*) or mobile phones (e.g., *Iphone*) that have a historical counterpart from which the thing stems. The query does not seek different versions of the *Ipod* or *Iphone* but is a way to express the concept without having the correct words to express it with. General language evolution is also needed for interpreting documents. If the user is presented with a document about a *phonograph*, he or she is unlikely to accept the document as relevant without knowing of the connection between the *phonograph* and the *Ipod*, both are music playing devices for different periods in time.

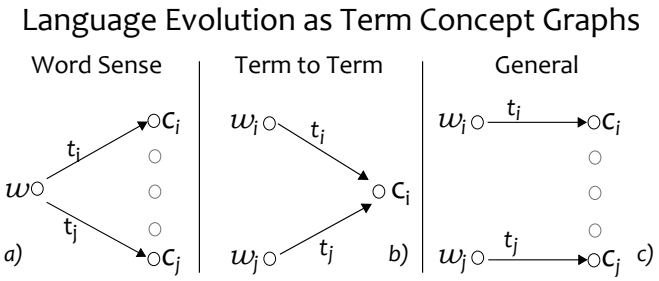


Figure 3.5: Term to Term, Word Sense and General Language Evolution modeled as Term Concept Graphs.

3.4 Modeling Evolution using Term Concept Graphs

All classes of evolution presented in this thesis can be described using term concept graphs. The graphs are of three general types as can be seen in Figure 3.5. Word sense evolution (Figure 3.5a) considers one term and all its concepts over time. The concepts can be related or completely separated. One example is the term **rock** that gained its *music* sense in mid 20th century, a sense that has since evolved. The *stone* sense however has been more stable and experienced little or no evolution. Term concept graphs for one point in time have concepts that represent individual word senses. Term concept graphs for several time points have concepts that represent several, grouped word senses.

Term to term evolution (Fig. 3.5b) can be described using one concept or context c_i with several terms pointing to it. When $t_i \neq t_j$ we consider w_i and w_j to be term to term evolutions ($w_i \rightarrow w_j$). When $t_i = t_j$ we consider w_i and w_j to be synonyms. In the case of preservative evolution w_i evolves into w_j but can be used as a "reminder" with a much lower frequency also after $w_i \rightarrow w_j$. Examples for this type of preservative evolution is the city name *Stalingrad*, when referring to the World War II era the old name is used instead of the current name *Volgograd*. In these cases we allow t_i and t_j to be overlapping but do not consider w_1 and w_j to be synonyms.

Spelling variations can be described with the same TCG that is used for term to term evolution (Fig. 3.5b) with additional constraints. Spelling variations can be used simultaneously so we allow $t_j \geq t_i$ but require the distance $dist(t_i, t_j)$ between the terms to be limited. Some variant of the Levensthein distance (Levenshtein, 1966) or a phonetic distance algorithm like Phonix (Gadd, 1990) can be used for measuring the distance between the terms.

Finally, general language evolution (Fig. 3.5c) can be described as a combination of the term concept graphs used to describe word sense as well as term to term evolution. Two terms point to two different concepts or contexts which can be seen as evolutions of each other. This class typically contains terms that describe abstract concepts like *portable music player* or *means of transportation* where terms and their descriptions have changed over time.

Each class of evolution has its own time span typically needed for evolution. The time needed for term to term evolution of other part-of-speech than nouns is longer than that needed for named entity evolution. In addition, word sense evolution typically requires much longer time than named entity evolution. Though adding a sense to a word can be of shorter time frame, the removal or evolution of a given sense would require more time. These time spans are dependent not only on the class but also on the domain. For example, the technical development is moving fast enough that general language evolution in this domain can occur in a fairly short time span. The introduction and explosion of user generated text makes dissemination of language variations and evolution a simpler and less time consuming task. This leads to a more dynamic language and increases the chances for language evolution in certain domains (Tahmasebi et al., 2012c).

Chapter 4

State-of-the-Art

Out of the four main classes presented in Chapter 3.3, **spelling variations** is the most researched class with respect to automatic methods for its detection. However, as spelling variations are not the focus of this thesis, we will not explicitly review this topic. We instead refer to papers from Ernst-Gerlach and Fuhr (2007), Gotscharek et al. (2009a) and Hauser et al. (2007) for some insight on what problems can be caused by spelling variations with respect to searching in long-term archives. In general, very little work can be found on the topic of automatic language evolution detection for the other three classes in the fields of computer science and computational linguistics. In this chapter we will review existing work directly in the field of word sense evolution and named entity evolution. In addition, we will review relevant technologies for detecting word sense evolution.

We note a difference in terminology between the works that originate in computer science versus computational linguistics. In linguistics, **language evolution** is tied to the origin of language and speech rather than variation within one single language. In this context science from fields like anthropology, biology, neuroscience and cognitive sciences come together to answer questions about what made humans develop language and speech. For an overview of this field we refer to the book by Christiansen and Kirby (2003). In computer science the term *evolution* is used to describe changes in language, while in computational linguistics as well as linguistics, the term evolution is replaced with *change* or *variation*. Continuing this chapter we will use all three terms interchangeably depending on context.

First in this chapter we will review the work that has been done on word sense evolution and continue onwards with named entity evolution. As a part of word sense evolution we will discuss word sense discrimination, that are algorithms for automatically discovering word senses from text, as well as the evaluation of automatically derived word senses. We will also review alternative methods, like the use of topic modeling, for sense discrimination. We will continue this chapter by describing works from the field of computational linguistics on word sense evolution, that is, novel sense detection as well as detection of change in word senses over time. Finally we will review a related field that considers cluster evolution.

4.1 Word Sense Evolution

To automatically detect word sense evolution we must first be able to detect word senses automatically in a given collection of text. For this reason we will start by describing different methods for automatically finding and modeling word senses. Then we will provide an overview of the existing work that has addressed word sense evolution so far.

4.1.1 Word Sense Disambiguation

Methods for automatic detection of word senses are called word sense discrimination and should be differed from word sense disambiguation. The task of word sense disambiguation is, given an occurrence of an ambiguous word and its context (usually sentence or surrounding words in a window), to determine which sense is referred to. Usually the senses used in word sense disambiguation come from explicit knowledge banks such as dictionaries or ontologies like WordNet. Word sense disambiguation is among other things a helpful component in machine translation and query expansion. Here it is the belief that the immediate context of a word helps determine which sense is referred to. In the sentence *I am listening to rock* a human can clearly interpret that *rock* is referring to *music* while in the sentence *I am sitting on a rock* the *stone* sense is referred. When translating, the correct sense of a word can be used to obtain a correct translation. E.g., the translation of the two sentences into Swedish would be *Jag lyssnar på Rock musik* versus *Jag sitter på en sten* where underlined terms correspond to different translations of the word *rock*.

Already in the late 1940s, word sense disambiguation was considered a major part of machine translation and has since been treated as a difficult problem. In the late 1980s, word sense disambiguation was described as a AI complete problem, i.e., a problem whose difficulty is at least as hard as the most difficult problems in Artificial Intelligence (Navigli, 2009). The problem is considered difficult based among others on the representation of word senses and the use of external knowledge as sense inventories. For a formal description of the problem of word sense disambiguation and a well structured survey of different approaches for solving the problem we refer to the survey of Navigli.

4.1.2 Word Sense Discrimination

Word sense discrimination is the process of identifying the meaning of words in a computational manner. If the senses are not a priori known and given in e.g., a thesaurus or sense inventory, word sense discrimination can be applied to find the senses used in word sense disambiguation.

Using word sense discrimination instead of a thesaurus has its advantages. The method can be applied to domain specific corpora where few or no knowledge banks can be found. Such corpora can correspond to detailed technical data such as biology or chemistry and on the other end of the spectra, user generated texts like Blogs where much slang or gadget names are used. For our purposes we will apply the method on historical data where such few resources are available or are not available in machine readable format.

Word sense discrimination techniques can be divided into two major groups, supervised and unsupervised. Due to the vast amounts of data that are considered in this thesis, focus will be given to unsupervised techniques. This group can also be divided into three major areas (Ferret, 2004) the first of which is represented by one of the most prominent works within the field of automatic detection of word senses by Schütze (1998). The word sense discrimination algorithms in this group make use of *feature vectors* to represent words. An unsupervised clustering algorithm is applied to these feature vectors and each cluster is considered as a sense of the target word. The second major area is represented by Pantel and Lin (2002) that use *dependency triples*. The third area is represented by Dorow et al. (2005) where a *graph view* is taken in order to cluster terms which represent word senses.

4.1.3 Word Sense Discrimination using Feature Vectors

The basic idea of context group discrimination presented by Schütze (1998) is to induce senses from contextual similarities. Each occurrence of an ambiguous word in a training set is mapped to a point in word space. This means that for each word w in a corpus, a word vector (co-occurrence vector) is created. Each position in the vector corresponds to one term v_i and the value of that position corresponds to the number of times that v_i is found in the neighborhood of w in the entire corpus. The neighborhood can be defined as a limited number of words before and after w or the sentence/paragraph/document in which w participates. A context vector is then considered as the centroid (or sum) of all word vectors corresponding to the words found in one context around word w. Because the context vector is a sum of all word vectors it has the same dimensions as each word vector but the co-occurrence values differ.

Similarity between two vectors (word vectors as well as context vectors) can be measured by means of cosine similarity. This would correspond to measuring the angle (measured by means of term overlap) between the two vectors. A small angle (i.e., a large overlap in the co-occurring terms) corresponds to similar vectors and indicates that the terms are semantically related. A large angle between the vectors corresponds to the inverse, (see Eq. 3.3).

To create word sense clusters, the set of context vectors are clustered into a number of coherent clusters using the Buckshot algorithm (Cutting et al., 1992), which is a combination of the EM-algorithm and agglomerative clustering. The idea is to create clusters of context vectors such that the context vectors in one cluster are more similar to each other than to context vectors from other clusters. The centroid of a cluster is considered as the representation of one sense and has the same dimensions as each word vector. The method is completely unsupervised since no hand tagging or manual effort is required. The disadvantages of this method are that the clustering is a hard clustering and that the number of clusters has to be predetermined. A hard clustering is a partitioning of all terms while a soft clustering allows for each term to participate in multiple clusters. Because terms can have multiple senses, they belong in several word sense clusters. Therefore, soft clustering approaches are more appropriate for word sense discrimination.

Marneffe and Dupont (2004) continued the previous work and concluded that the method is based on a simplified model of the actual Gaussian distribution for each cluster. The paper studies the impact of estimating the simplified model rather than estimating the real Gaussian model. The authors conclude that the vector model proposed by Schütze (1998) can be significantly improved when a real Gaussian model is estimated instead of its hard clustering approximation. The performance gain is obtained with additional costs for computation.

A systematical comparison of unsupervised word sense discrimination techniques for clustering instances of words using both vector and space similarity is conducted by Purandare and Pedersen (2004). In this paper a comparison is made of the Schütze method described above and a method proposed by Pedersen and Bruce (1997, 1998). In the latter a similarity based discrimination approach is used that computes similarity/dissimilarity among each instance of the target words computed from first order context vectors. As dataset 24 out of the 73 words in SENSEVAL-2 (Edmonds and Cotton, 2001) are used as well as the Line, Hard and Serve sense tagged corpora (Leacock et al., 1998, 1993). Senseval is a 20 Million word corpus, part-of-speech tagged data that is manually sense tagged. The Hard, Line and Serve corpora are over 4000 instances of the nouns *hard*, *line* and *serve* tagged with part-of-speech where each instance of the words is tagged with one out of 6, 3 and 4 WordNet senses respectively. In the evaluation standard hard clustering is assumed. The results of the comparison suggest that second order context vectors have an advantage over first order vectors for small training data while for larger amounts of homogeneous data such as the Line, Hard and Serve data, first order context vector representation is the most effective at word sense discrimination.

4.1.4 Word Sense Discrimination using Dependency Triples

The second group of algorithms build upon dependency triples and a similarity measure proposed by Lin (1997, 1998a). A dependency triple (w, r, w') consists of two words w and w' and a grammatical relationship between them in a given sentence. Similarity of terms is measured by means of their synsets and relation between synsets in WordNet.

The similarity between w and w' is considered to be the maximum similarity between their synsets and the similarity between two synsets s_1 and s_2 is defined in Equation 4.1.

$$sim(s_1, s_2) = \frac{2 \cdot log(s)}{log(s_1) + log(s_2)} \tag{4.1}$$

where s is the most specific synset that subsumes both s_1 and s_2 and $log(s)$ is the logarithm of the probability that a randomly selected noun refers to a synset s or any synset below it in the WordNet hierarchy.

The similarity measure was used to automatically create a thesaurus. This thesaurus was shown to be closer to WordNet in similarity than Roget Thesaurus and hence a suitable similarity measure for comparing words and word senses.

In 2002, the Clustering By Committee (CBC) algorithm was presented by Pantel and Lin. The algorithm makes use of dependency triples and the similarity measure above and consists of three phases. In the first phase each element's top-k similar elements are computed. Each element is represented by a feature vector (based on relations in dependency triples) and the similarities are calculated according to the similarity measure above. In the second phase a collection of tight clusters are constructed where the elements of the clusters form a committee by averaging the feature vectors. The aim of the algorithm is to form as many committees as possible under the condition that the newly formed committee is not similar to an existing committee. In the final phase each element is assigned to its most similar committee and the committees are considered as word senses.

The key to discovering word sense is that once an element e has been assigned to a cluster c, the intersecting features between e and c are removed from e. This helps CBC to discover the less frequent senses of a word, avoids duplicate detection and results in soft clustering.

The authors propose a method for evaluating clusters (i.e., candidate word senses) against the senses available in WordNet. A cluster c is said to correspond to a correct sense of a word (i.e., a synset s from WordNet) if the similarity between the cluster and the synset is above a certain threshold. In addition, the similarity between a synset s and a word w is the maximum similarity between s and any synset of w (Equation 4.1). The algorithm for comparing a cluster and a synset can be described as follows. Let c_k be the top-k words of a cluster c, where these are the k most similar words to the committee of c. The similarity between s and c is the average similarity between s and the words in c_k. The cluster c correctly corresponds to a sense s if the similarity is above a threshold α.

The method is based on the similarity measure in Equation 4.1 and depends on the probability that a randomly selected noun refers to a synset s or any synset below it in the WordNet hierarchy. These probabilities are estimated using the frequency counts of the SemCor data that is semantically annotated with WordNet senses.[1] Due to a rather small dataset (200K) the probabilities are smoothened assuming that all siblings are equally probable given the parent. The method has since been widely used in (Dorow, 2007; Deschacht et al., 2007; Roa et al., 2008; Ferret, 2004) and implemented by Ted Pedersen et. al. in the WordNet::Similarity Package.[2]

1 The SemCor corpus, created by the Princeton University, is a subset of the English Brown corpus containing almost 700,000 running words. In SemCor all the words are tagged by PoS, and more than 200,000 content words are also lemmatized and sense-tagged according to Princeton WordNet 1.6.

2 A description of this package can be found on `http://www.d.umn.edu/~tpederse/similarity.html`

The precision of CBC is measured as the percentage of output clusters that actually correspond to an existing synset of a given target word. This means that only clusters that have terms from WordNet can be evaluated. The recall is defined as the ratio between the correct clusters to which a word w is assigned and the actual number of senses in which w was used in the corpus. The target set of senses for w is found by pooling the results of the implemented algorithms. The test set is constructed by intersecting words in WordNet with words from the corpus. The authors implemented several other algorithms, among others Buckshot, K-means and Average Link, and show that CBC outperforms all algorithms implemented in both recall and precision. The algorithm is quadratic in its running time, however, the authors claim that it can be reduced dramatically by taking advantage of the fact that the feature vectors are sparse for each word.

4.1.5 Graph Algorithms for Word Sense Discrimination

Graph algorithms represent the third category of unsupervised word sense discrimination techniques. Dorow et al. (2005) present two complementary approaches for categorizing words. Both methods use a graph theoretical representation of words and their relationships. Ambiguity is especially addressed in this paper. A word graph G is build using nouns and noun phrases extracted from the British National Corpus (BNC, 2007). Each noun or noun phrase becomes a node in the graph and are linked if the words can be found in coordination in the text. More precisely, there is an edge between nodes if the corresponding nouns and noun phrases are found in the text separated by *and*, *or* and commas. The order in which two terms co-occurs is ignored and the graph is undirected. The curvature value of a node w is defined as the following.

$$curv(w) = \frac{\#\text{triangles that } w \text{ participates in}}{\#\text{triangles that } w \text{ could participate in}} \tag{4.2}$$

This is also referred to as the *clustering coefficient* (Watts and Strogatz, 1998) of a node. Curvature is a way of measuring semantic cohesiveness of the neighbors of a word. If a word has stable meaning, the curvature value will be high. If, on the other hand, a word is ambiguous, the curvature value will be low because the word is linked to members from different communities which do not link to each other. The authors show that curvature values have a higher correlation to the number of WordNet senses for a word, than word frequency. The clustering algorithm proposed is called *curvature clustering* and consists of the following steps:

1. Compute the curvature value of each node in the graph
2. Remove all nodes whose curvature value falls below a certain threshold (0.5 in the paper)
3. Resulting connected components constitute clusters of semantically similar words.

The connected components constitute hard clusters. To allow the ambiguous words to participate in all related clusters, the connected components are enriched with the direct neighbors of the members in the cluster. This procedure makes the clustering a soft clustering. The algorithm automatically determines the number of clusters and hence does not suffer from the downside of having to predetermine the number of clusters.

The second approach presented by Dorow et al. (2005) is inspired by Schütze's method where a context graph or link graph is created. As context, pairs of words, which are linked in the graph described previously, are considered. Then (*rock*, *singer*) can be distinguished from (*rock*, *granite*) because *rock* in the context of *singer* refers to another context than *rock* in the context of *granite*. By clustering word contexts as opposed to clustering words themselves, a words different meaning can be distributed across different clusters which can then be interpreted as word senses. The link graph denoted G' is constructed according to the following.

1. Introduce a node n_l in G' for each edge l in the original graph G
2. Connect any two nodes n_{l_1} and n_{l_2} in graph G' if l_1 and l_2 co-occurred in a triangle in G.

The link graph is clustered using a Markov Clustering algorithm presented by van Dongen (2000). The authors reported good results for both methods and while curvature was shown to be particularly suited for measuring the degree of ambiguity of words, link clustering showed to be well suited for splitting ambiguous words into their different meanings.

A more thorough investigation of the curvature measure as well as the curvature clustering algorithm was presented by Dorow (2007). An evaluation of the curvature algorithm is made on the BNC corpus and the evaluation method presented in (Pantel and Lin, 2002) is employed (using $\alpha = 0.25$). Dorow reports a higher precision for the curvature clustering algorithm than the one reported for the CBC algorithm. The recall however is slightly lower than for CBC though it should be noted that the target senses used for computing recall differ. Evaluation methods for word sense discrimination algorithms are discussed in Section 4.1.7. Because of the higher precision, we choose to make use of the curvature clustering algorithm for finding word senses in this thesis.

The curvature measure used by Dorow et al. (2005) considers a binary graph without edge weights. This means that Equation 4.2 counts every triangle once without taking into consideration the number of times each triangle occurred. A natural extension would be to also consider the positive edge weights for example following the work presented by Kalna and Higham (2007).

A method similar to the curvature clustering is employed by Ferret (2004). Using a sliding window of size 20 the author creates a co-occurrence matrix of 30,000 words and 4.8 million co-occurrences. Unlike the graph presented in Dorow et al. (2005) (where two nodes are linked if they co-occur in coordination in the text), the graph presented in this work links two nodes if the words representing these nodes are similar according to one of two similarity measures. The Shared Nearest Neighbors algorithm (Jarvis and Patrick, 1973) is used to find clusters representing word senses. It should be noted that the algorithm yields in a significant percentage of terms without any sense.

4.1.6 Using Topics for Word Sense Discrimination

Levin et al. (2006) present an evaluation on Latent Semantic Analysis (LSA) for unsupervised word sense discrimination. Traditionally LSA is used in natural language processing for finding complex and hidden relations of meaning among words and the context in which they were found. It has been used for language modeling, word and document clustering, call routing and semantic interface control. A co-occurrence matrix is created where the columns represent different contexts and the rows represent different word tokens.

The main step of LSA is to perform singular value decomposition on the normalized matrix. If the singular values are sorted in decreasing order, the matrix can be reduced to a much lower rank. The hypothesis is that this reduction not only lowers the computational cost of clustering but also provides better results compared to clustering the full matrix. The context vectors used in word sense discrimination are LSA based representations of documents in which the ambiguous word appears, meaning the entire document is used as context. The clustering is then evaluated based on tightness and purity, where tightness indicates that the vectors in the cluster are close to each other and close to the centroid of the cluster. Purity indicates that the vectors that belong to a cluster correspond to words with the same sense. Two measures were used, *sense discrimination accuracy* and *average silhouette value*. The average silhouette measure considers, for each point, how similar it is to points in its own clusters compared to points in other clusters. The results indicate that the utility of using dimensionality reduction using LSA lies not in the improved sense discrimination but making the subsequent computations more efficient.

Continuing the work of applying LSA in a real world application, Pino and Eskenazi (2009) try to match the meaning of a word in a document to the meaning of a word in a fill-in-the-

blank question. The conclusion is that all though LSA helps to overcome the sparsity of short contexts such as a question and gives improvements over a baseline using Lesk algorithm (Lesk, 1986) with exact matching, it is difficult to apply LSA to large amounts of data due to the computationally heavy SVD operation.

Another potential method for word sense discrimination are probabilistic topic models. Boyd-Graber et al. (2007) consider word sense disambiguation as the primary goal but a kind of word sense discrimination called LDAWN, short for Latent Dirichlet Allocation with WordNet, is performed as a subtask. The approach is an unsupervised, probabilistic topic model that includes word senses as a hidden variable. Latent Dirichlet allocation (LDA) assumes that there are K *topics*, represented as multinomial distributions over all words which describe a collection. Each document exhibits multiple topics and each word in each document is associated to one such topic. Because topic models capture the polysemous use of words, but do not carry an explicit notion of 'sense', the multinomial topic distribution is replaced with a WordNet walk which results in LDAWN.

A WordNet walk uses the hyponym relation of WordNet and lets an agent start in the synset *Entity* which subsumes all nouns in WordNet, and then chooses the next node from the hyponyms of its current position. The agent repeats this process until it reaches a leaf node which corresponds to a single word. This procedure results in a distribution over words that can be used in LDA. The synset that produce each word is assumed to be a hidden variable and posterior inference is used to predict which synset produced a word. LDAWN is used for word sense disambiguation where word sense discrimination is a subtask, therefore, the found word senses are evaluated indirectly by means of disambiguation.

The authors report that high frequency terms with many senses often are wrongly disambiguated and hurt the accuracy of the system and conclude that errors associated with less frequent terms reveal that the structure of WordNet cannot easily be transformed into a probabilistic graph. They report that LDAWN is substantially less effective in disambiguation compared to state-of-the-art. One reason for the poor performance can be that concepts created by LDA correspond to different domains rather than to word senses.

4.1.7 Evaluation of Word Sense Discrimination

There are two main methods for evaluating word sense discrimination algorithms. The first method evaluates extracted senses by comparing the senses to a sense repository, e.g., WordNet (Pantel and Lin, 2002; Dorow, 2007) while the second method performs an indirect evaluation by means of word sense disambiguation (Pedersen and Bruce, 1998; Lau et al., 2012). For the latter there are standard datasets available like SENSEVAL-2 (Edmonds and Cotton, 2001) on which new word sense discrimination algorithms can be applied and evaluated. In this thesis we are not suggesting a new method for word sense discrimination. Instead we want to apply existing methods to historical texts with a long time span in order to find word sense evolution. For such datasets there exist no sense information or disambiguation datasets, and we are therefore left with evaluation methods that make use of existing sense repositories.

The key components of an evaluation method that relies on a sense repository are firstly, the senses stored in the repository and secondly, a similarity measure that captures how similar an extracted sense is to a sense stored in the repository. Though there exist different sense repositories like Webster's 7th Collegiate, the Collins English Dictionary and the Oxford Advanced Learner's dictionary of Current English, all used by Lesk (1986), most current methods make use of WordNet (Miller, 1995). The main advantage of WordNet is that the senses are stored in a hierarchy of *is-a* relations as well as other relations like *has-part, is-made-of, is-attribute-of.*

A good overview of different similarity measures can be found in Pedersen et al. (2004) that also describe a Perl implementation of mentioned similarity measures. These similarity measures can be roughly divided into two groups. The first group considers path lengths between concepts. For example, Leacock and Chodorow (1998) consider the similarity between two concepts to be based on the shortest path between the two concepts in a sense repository considering only *is-a* relations.

The second group considers information content where two concepts are more similar the more information they have in common. The information is approximated using sense tagged corpora. Resnik (1995) considers the information content of a concept c as the negative log likelihood of the probability of encountering the concept: $-log\ p(c)$. The similarity between two concepts c and d is defined as

$$sim(c,d) = max_{s \in S(c,d)}[-log\ p(s)] \tag{4.3}$$

where $S(c,d)$ is the set of concepts that subsume both c and d in the WordNet hierarchy. The probabilities for each concept are estimated using the Brown Corpus (Francis, 1964). The details of the Lin measure (Lin, 1998a) which also fall in this group of similarity measures is described by Equation 4.1. The probabilities for encountering a sense are found in the same way as described above.

Pantel and Lin (2002) describe an evaluation method for evaluating the output of a word sense clustering algorithm to WordNet by means of information content. The method has widely been used and implemented in the WordNet::Similarity package Pedersen and Bruce (1997). Due to a wide acceptance of the method and the fact that the same evaluation method has been employed when evaluating the output of the curvature clustering method (Dorow et al., 2005; Dorow, 2007), we base our method of evaluation on this work.

We have chosen to use the curvature clustering algorithm for automatically detecting word senses from a collection of documents. In Section 4.3 we provide a motivation for our choice.

4.1.8 Word Sense Variation

Automatic detection of changes and variations in word senses over time is a topic that has gained interest recently. During the past years researchers have evaluated and researched different parts of the problem mainly in the field of computational linguistics.

Sagi et al. (2009) presented work on finding the senses of words by means of context vectors and found *narrowing* and *broadening* of senses over time by applying semantic density analysis. Each word occurrence of a target word is mapped to its context vector which follows the Schütze (1998) definition. A context around a word is considered to be 15 words before and after each target word. The 40,000 most frequent words excluding stopwords constitute the vocabulary W and the 50th to 2049th most frequent words from the vocabulary are considered to be content bearing terms C.[3] A matrix $W \times C$ is created with co-occurrence counts where each element $< w, c >$ in the matrix marks the number of times a word c occurred in the context of a word w in the corpus. Singular value decomposition is used to reduce the dimensionality of the matrix from 40,000 $\times$ 2,000 to 40,000 $\times$ 100 by finding the most important content bearing terms C'. Using these terms, a context vector around each target word can be created by finding the number of times a token $c \in C$' is found in the context the target word.

For a specific target word w each occurrence of the word in the corpus can be mapped to a context vector. The semantic density of the word in a specific corpus can be seen as the average cosine similarity of the vectors. A high similarity can be seen as a dense set of vectors and correspond to words with a single, highly restrictive meaning. A low similarity is seen as

3 To capture as much variation as possible, the stopwords list is kept to a minimum and instead the 49 most frequent terms are disregarded as content bearing terms. These can be seen as stopwords in the specific vocabulary.

a sparse set of vectors and corresponds to a word that is highly polysemous and appears in many contexts. To reduce the computations a Monte Carlo analysis is conducted to randomly choose n vectors for pair-wise computation. To measure change in word senses over time, context vectors are created for a target word in different corpora (from different time points) and the semantic density is measured for each corpus. If the density of a word is increased over time then it is concluded that the meanings of the word has become less restricted due to a broadening of the sense or an added sense. Decreased density over time corresponds to a narrowing of the sense or lost senses.

Unlike in the work by Schütze (1998), the context vectors are not clustered to give more insight into the different senses. Instead, a random set of context vectors are selected at one point in time to represent the overall behavior of a word. This means that even though there can be indication of semantic change there are no clues as to what has changed. The problem here is reduced to the same as in our case, to map different senses to find out *what* has changed.

Similar to the work described above, the work presented by Gulordava and Baroni (2011) builds on context vectors to identify semantic change over in time. The definition of context is however slightly different from the previous presented papers as the authors use Google books Ngram data as their corpus. More specifically 2-grams (pairs of words) are chosen which means that the context of a word w is the other word in the 2-gram.

Two separate sub-collections are chosen, the first one corresponding to the years 1960-1964 (the 60's) and the second one corresponding to 1995-1999 (the 90's). The content bearing words are chosen to be the same for both collections and thus the word space for each context vector is the same for the 60's as for the 90's collections. That means, the context vector for a word w has the same dimensions (the same words) in the 60's as in the 90's but with different co-occurrence counts. The two context vectors corresponding to the word are compared by means of local mutual information similarity scores.

Among the 10,000 randomly chosen mid-frequency words 48.4% had very high similarity scores, 50% had mid-range similarity scores (between $0.8-0.2$) and only 1.6% had a lower similarity score than 0.2. The assumption of the authors was that words with low similarity scores are likely to have had a semantic change, an assumption that was tested by manually evaluating a sample of 100 words over all similarities. The words were also differentiated based on the increase or decrease of frequency between the 60's and the 90's. The results indicate that some level of semantic change can be found using this methodology. However, as with the previous paper, it is not clear what happens to the word and there is no differentiation between word senses. All senses are considered at once and there is no alignment between the different senses over time.

The work presented by Lau et al. (2012) aims to detect word senses that are novel in a later corpus compared to an earlier one. Topics are used to represent word senses and thus word senses discrimination is made by means of LDA. In particular, a non-parametric topic model called Hierarchical Dirichlet Process (Teh et al., 2004) is shown to provide the best results in the word sense discrimination task. The process for detecting word senses is the following; firstly, topics are detected for two corpora, a reference corpus and a newer corpus. Secondly, each instance of a target word w in the corpora is assigned a topic which has been chosen among all detected topics regardless of which corpus it originated from. Finally, if a topic is assigned to word instances in the latter corpus but not in the former, then it is considered novel. A novelty score is proposed which considers the difference in probability for topic assignments normalized by a maximum likelihood estimate.

The reference corpus is chosen to be the written parts of the BNC and the second corpus is a sample of the 2007 ukWaC Web corpus (Ferraresi et al., 2008). Ten words are chosen for deeper examination, half of which have been manually assessed to have experienced change while the other half has remained stable. Though it is not suggested by the authors, the method can be used to find the inverse as well; if a topic is assigned to instances in the reference corpus but

not in the second corpus, then the sense can be considered as outdated. Overall, the method shows promising results for detecting novel word senses by means of topic modeling. However, an alignment of word senses over time or the detection of relations between the senses is not covered in this work.

The work conducted by Bamman and Crane (2011) aims to track the rise and fall of Latin word senses over 2000 years. By using a bilingual sense inventory for training a word sense disambiguation classifier, the relative frequency of all word senses related to a target word over time are tracked.

The use of two aligned corpora in different languages allows for translation of words into another language to help approximate the senses of the word. If a word in language A is ambiguous, it is likely translated into different words in language B, thus enabling to detect that the word is indeed ambiguous. The number of different translations in language B will provide a probable guess on how many different senses are valid for the word in language A. The translation mechanism also helps to determine the frequency with which the instances of the target word are assigned to the different senses. The more often the target word is translated to word i in language B, the more often the sense i is assigned to the target word in language A.

The bilingual corpus consists of 129 manually chosen Latin-English book pairs evenly distributed in time as a training set. Using the training set a 6-gram language model classifier for word sense disambiguation was trained. Using this classifier, a total of 7055 books containing 389 million words were classified. Five Latin words with known word sense shifts were chosen for deeper analysis of the sense variation over time and 105 instances of each word were manually labeled. The variation of the senses was measured as the proportion of a sense assignment to all sense assignments for a word. The results clearly show that sense variations can be measured over time and point to a change in the predominant sense over time for the 5 chosen terms.

The method is far more beneficial in studying words and their meanings over time than just performing studies based on word frequency. Similar to Lau et al. (2012) new and outdated senses could be detected though it was not a part of the work described in the paper. However, the method is restricted as it requires a translated corpus to train the word sense disambiguation classifier. It also does not allow the senses to be aligned over time to follow the evolution of senses and their relations.

To the best of our knowledge there is only one other work that directly targets automatic tracking of word senses over time presented by Wijaya and Yeniterzi (2011). The objective of this work is to track changes that occur to an entity in terms of changes in the words that co-occur with that entity. Similar to the work by Lau et al. (2012) topics are used to approximate word senses. The experiments are conducted on Google Ngram data where 5-grams are chosen in such a way that the target term w is the 3rd word, i.e., (w_1, w_2, w , w_4, w_5). A document D^i_w is created for each year i consisting of all 5-grams where w is the third word. Then these documents are clustered using two different clustering methods. For each method the documents D^i_w are represented with a word vector where the values differ depending on clustering technique. The first experiment makes use of the $K-means$ clustering algorithm and each word in the word vector corresponding to a document is represented by the tf-idf value of the word. The second experiment makes use of a Topic-Over-Time algorithm (Wang and McCallum, 2006) which is a LDA like topic model. The word vector is represented by occurrence frequencies of each word in the corresponding document.

In each experiment topics are considered as changing if two consecutive years are assigned to different clusters. To reduce noise, only clusters that have a consecutive run for more than 3 years are chosen. For the *K-means* algorithm this means the following: assume that cluster 1 contains documents $\{D^1_w, D^2_w, D^3_w, D^4_w\}$ and cluster 2 contains documents $\{D^5_w, D^6_w, D^7_w, D^8_w\}$. Then year $4-5$ is the change period and the top words for year 4 and year 5 that represent

cluster 1 and cluster 2 are used to represent the different meanings. For the Topic-Over-Time clustering, topics are clustered instead of documents but the rest is analogue to the $K-means$ clustering and topics that are active at the time of change are used to represent the different word meanings.

A few different words are analyzed and there is indication that the method works and can find periods when words change their primary meaning as well as detect which new meaning is used to replace the previous. By manually analyzing the topics and the co-occurrence graphs the authors can determine what happened to the word. For example, by studying the word *gay* the authors can find the shift from *happy* to *homosexual* by the change in clusters and topic terms. It can also be seen in the co-occurrence graph as an addition of a new sub-graph corresponding to the latter meaning and eventually the removal of the sub-graph that corresponds to the former meaning.

Adjectives seem not to be well suited for the method as the topics cannot well capture the meaning of an adjective. This might be because topic modeling is not optimal for capturing word senses (Boyd-Graber et al., 2007). In general, the work in this paper is preliminary but it is the first paper to provide an automatic method for tracking senses to find out *what* happened rather than *when*. There is no proper comparison between the different clustering algorithms to indicate which method performs better or to quantify the results. Nevertheless, the overall methodology to use clustering to associate different topics or documents with each other can be a promising direction. In our case this would mean clustering (or by other means partitioning) word sense clusters to relate word senses over time.

4.1.9 Tracking of Word Sense Clusters

Related to tracking word senses over time is the field of cluster (or community) tracking from which we can gain insights related to our task. Therefore, in this section we review representatives of existing methods for the tracking of communities.

Analysis of communities and their temporal evolution in dynamic networks has been a well studied field in recent years. A community can be modeled as a graph where each node represents an individual and each edge represent interaction among individuals. When it comes to detecting evolution, the more traditional approach has been to first detect community structure for each time slice and then compare these to determine correspondence. These methods can be argued to introduce dramatic evolution in short periods of time and can hence be less appropriate to noisy data (Lin et al., 2008).

Representing the traditional approach a framework called Monic is proposed by Spiliopoulou et al. (2006) for modeling and tracking cluster transitions. In this framework internal as well as external cluster transitions are monitored. The disadvantages of the method are that the algorithm assumes a hard clustering and that each cluster is considered as a set of elements without regarding the links between the elements of the cluster. In a network of lexical co-occurrences this can be valuable since the connections between terms give useful information to the sense being presented. Palla et al. (2007) propose a method for detecting cluster evolution that also takes into account the edge structure among cluster members. Clusters are created first for two different time points t_i and t_j separately and then for the two time points jointly $t_{i\cup j}$. If two clusters from t_i and t_j end up in the same cluster in $t_{i\cup j}$, they can be considered related. In order for the method to work, specific properties of the clustering algorithm are assumed. The clustering algorithm used in the paper does not guarantee that the found clusters will correspond to word senses and is therefore not appropriate for word sense discrimination. However, the properties of the curvature clustering algorithm does not guarantee that two related clusters from t_i and t_j end up in the same cluster in $t_{i\cup j}$. Therefore the method cannot be considered when using the curvature clustering algorithm for word sense discrimination.

In contrast to the traditional approach where clusters are detected independently for individual time points and then compared to find relations, Lin et al. (2008) propose a framework where the clustering and the relations are found at the same time. The FacetNet framework discovers community structure at a given time step t which is determined both by the observed data at t and by the historical community pattern $t-1$. The problem is stated as an optimization problem with the objective to minimize the difference between the community structures at time t and $t-1$ while maximizing the fit to the data at time t. Because historical structures are taken into consideration, FacetNet is unlikely to discover community structures that introduce dramatic evolution in a very short time period and is therefore well suited for noisy data. However, as with the method above there is no guarantee that clusters correspond to word senses.

A method for describing and tracking evolution can be found in the related field of Temporal Text Mining. In the work by Mei and Zhai (2005) themes are found and tracked over time. A theme evolution graph is defined that seems particularly suitable for describing word sense evolution and is similar to what the erm concept graphs presented in Chapter 3.2. Themes are represented using topics and different themes are compared to determine if one theme evolved into another. The key to the tracking is the similarity measure used to compare two themes. Because a theme is expressed as a probability distribution over all words in a corpus, Mei and Zhai (2005) use Kullback-Leibler divergence (Kullback and Leibler, 1951) to compare themes and create theme evolution threads. The Kullback-Leibler divergence measures the difference between two probability distributions and shows promising results in tracking themes. However, since we represent our clusters as a set of terms without any weights, and not probability distributions, the similarity measure cannot be used for our purposes.

Changes in Semantic Orientation of words

Knowing the semantic orientation of a word can help when interpreting documents in long-term archives. Knowing the orientation can help to determine the difference between *awesome* and *awesome* from two different periods in time. The method does not differentiate between different senses, i.e., the term *awesome* today has both a positive and a negative sense; however, the predominant value of a word can be determined. Though semantic orientation is not enough to determine the full meaning of a word, it can still contribute positively to the interpretation of information from the past. It is however not straight forward to determine the orientation of a word over time as this is subject to semantic change similar to the change exhibited by the word sense.

Cook and Stevenson (2010) aim at detecting words that have changed their semantic orientation over time. In particular, they target words that have become more positive or negative. The work bases its method of determining semantic orientation of a target word on the work by Turney and Littman (2003) and a point wise mutual information method. The semantic orientation of a target word is determined by its association with a set of known positive and negative words. In this work, the set of known words are taken from the General Inquirer to be a larger set of known positive and negative words (1621 and 1989 respectively). The set of known words are ranked and the top 25% are chosen.

Three different datasets are used, all distributed from 1640 to the late 20th century. A set of 8 test terms are chosen where six which have become more positive and two that have become more negative over time. The methods is able to provide a positive value for words that have a positive change and a negative value for words that have a negative change. Though the method is not completely accurate the direction seems promising.

4.2 Named Entity Evolution

Previous work on automatic detection of term to term evolution has been very limited and mainly focused on named entity evolution. The interest has mainly been from an information retrieval perspective as named entity evolution makes finding relevant documents more challenging.

Berberich et al. (2009) proposed a solution to the problem of detecting named entity evolution by reformulating a query into terms prevalent in the past. The procedure is as follows, the user specifies a time point of interest, t_{int} and a query v for which query reformulations are sought. The time point t_{now} is specified as the current time point and a mapping is made between $t_{now} = T$ and $t_{int} = R$. For two terms $v@T$ and $u@R$ context terms are captured. Context terms are defined as all terms that co-occur to the target term within a fixed window of 10 terms on either side. Similarity between the two terms is then a probability based on the term overlap of the context terms (See Equation 4.4).

$$P(u@R|v@T) = \sum_{w \in V} P(u@R|w@R) \cdot P(w@T|v@T) \tag{4.4}$$

$P(u@R|w@R)$ and $P(w@T|v@T)$ are estimated based on co-occurrence statistics according to Equation 4.5 and 4.6 and V denotes the vocabulary of all terms in the collection.

$$P(u@R|w@R) = \frac{cooc(w@R, u@R)}{\sum_{z \in V} cooc(w@R, z@R)} \tag{4.5}$$

$$P(w@T|v@T) = \frac{cooc(v@T, w@T)}{\sum_{z \in V} cooc(v@T, z@T)} \tag{4.6}$$

The key to the similarity measure in Equation 4.4 is the overlap between the context terms of $v@T$ and $u@R$. If there are no overlapping terms w between the contexts of $v@T$ and $u@R$, then the similarity equals 0.

In addition to a similarity measure between the two terms, *coherence* and *popularity* measures are considered. The first measure takes into account coherence of different terms in the query reformulations: If the original query term is *Leningrad Cowboys* then it does not make sense to translate this query into *St. Petersburg Cowboys* because this term is not a valid term in the vocabulary. However, the query reformulation *Leningrad Museum* → *St. Petersburg Museum* is valid. The second measure considers the popularity of the query reformulation for the time point of interest and filters out reformulations that are unlikely. The approach is computationally expensive as it requires a recurrent computation each time a query is submitted because of the target time t_{int} for the query reformulations. It also requires checking all terms $u@R$ as reformulations for $v@T$ which reduces efficiency and scalability. The results presented in this paper are "anecdotal" (to use the words of the authors) and thus do not provide a basis for direct comparison. However, because of the promising results we use the same method for defining a context.

In the work by Kanhabua and Nørvåg (2010) the named entity evolution problem is approached using *time-based synonyms* as terms that are semantically related to a named entity at a particular time period. Their approach is different from that of Berberich et al. in that it relies on link information in Wikipedia rather than on the collection alone.[4] A target named entity is chosen as the title of a page P that describes an entity, e.g. *St. Petersburg*. Then candidate time-based synonyms are extracted from all internal links from any Wikipeida article to P, redirect pages that send the reader to P as well as disambiguation pages and categories that link to P. The anchor text of each link is used as a candidate time-based

4 `http://Wikipedia.org`

synonym. This methodology will capture terms like *Leningrad* that is redirected to the page of *St. Petersburg*.

Kanhabua and Nørvåg (2010) extract time-based synonyms using the full history of Wikipedia. The paper also proposes a method for detecting validity periods of each synonym. Due to the limited time span of Wikipedia, they extend the discovered time of synonyms using the New York Times Annotated Corpus. The authors evaluate the quality of the time-based synonyms by measuring increased precision and recall in search results rather than directly evaluating the quality of the found synonyms.

Though the quality of the results seems to be rather high, the method requires link information, such as anchor texts which limits the method to hypertext collections. When looking to the past and using newspaper collections (see Chapter 2.3) link information is not available and therefore the method cannot be employed in this thesis but should rather be considered when using e.g., Web data. There are also disadvantages of detecting named entity evolution on one corpus and using this for search in another corpus. There is a high likelihood of a gap between the collection used for finding named entity evolution and the collection used for search. As an example, the high quality name *Barack Hussein Obama II* is found as one time-based synonym in Wikipedia, however it cannot be found in the New York Times Annotated Corpus. The authors have gathered a set of time-based synonyms upon which we build our testset of named entity evolution.

Kaluarachchi et al. (2010) propose to discover semantically identical concepts, where concepts are named entities, used in different time periods. Their proposed method makes use of association rule mining to associate distinct entities to events. Sentences containing a subject, a verb, objects, and nouns are targeted and the verb is interpreted as an event while the noun is interpreted as an entity. Two entities are considered semantically related if their associated event is the same and the event occurs multiple times in both document collections. The temporally related named entity is used for query translation (or reformulation) and results are retrieved appropriately with respect to the specified time criteria. The authors present precision and recall for very few queries and evaluate only indirectly on the basis of retrieved documents. One disadvantage of the method lies in the dependency on the associated event (verb). Over long periods of time, also the verb will experience evolution and hence the task of associating entities over time is replaced with associating verbs over time. In fact, there is evidence to suggest that verbs are more likely to change over time than nouns (Sagi, 2010) and hence the problem of automatically detecting named entity evolution cannot be considered as fully solved by this method.

4.3 Summary

In order to find *word sense evolution* we need to find methods for automatically discovering word senses from large corpora. In particular, because of the size and nature of our intended collections and their time span (The Times Archive, 1785-1985 as well as New York Times, 1987-2007) we need methods that (1) do not rely on external resources like machine readable dictionaries or knowledge banks; and (2) do not require human input. We have investigated the main methods for finding word senses, namely word sense discrimination as well as topic modeling.

Topic modeling has been used to approximate word senses in among others the work of Boyd-Graber et al. (2007) and Lau et al. (2012), however, there is no strong evidence that topics can be used as a good representation of word senses. There seems to be a mapping between found topics and word senses (Lau et al., 2012) but no direct relation. Therefore, in this work we will use word sense discrimination as the main method for automatically detecting word senses.

We investigated three different methods for word sense discrimination. The first method, represented by Schütze (1998) uses hard clustering and requires some knowledge about the number of clusters expected from a corpus. Hard clustering is less suitable for word senses as each term can only appear in one cluster, i.e., one sense. Terms that are ambiguous should be allowed in as many clusters as as the term has senses and therefore, the two remaining methods seem more suited for our task. The second method, represented by Pantel and Lin (2002) provides clusters where each element has some likelihood of belonging to the cluster. This has the advantage of assigning elements to a cluster that are more significant than others. The algorithm is quadratic and might have scaling problems when the size of the corpus is large.

The third method, represented by Dorow et al. (2005) uses a graph based approach called curvature clustering and has reported higher precision than the one found by Pantel and Lin (2002) with a slightly lower recall. There are also fewer threshold values that need to be set for this method. As there is no strong evidence for either dependency triples or graph based methods, we opt to use the method with the higher precision and fewer thresholds. For evaluation of the word sense clusters we employ the evaluation method proposed by Pantel and Lin (2002) that is also used to evaluate the output of the curvature clustering. None of the presented methods for word sense discrimination have thus far been evaluated on historical data, (see Section 5.1). Therefore, as a starting point, we must verify that the output of the chosen discrimination method captures word senses also for text older than 50-60 years. Having chosen to rely on automatic word sense discrimination to find word senses it follows that we must limit our work on word sense evolution to nouns and noun phrases.

Thus far little work has been done to automatically detect word sense evolution and the work that has been done has focused on detecting that a word sense has changed. Sagi et al. (2009) found broadening and narrowing of word senses over time, by means of context vectors, without differentiating between or relating different senses. By applying clustering proposed by Schütze (1998) it would be possible to cluster context vectors into word senses. It still remains an open problem to find out which senses that are related over time. Lau et al. (2012) detected the appearance of novel word senses by making use of topics. By extension, their method could also find disappearing word senses. However, word senses are not related and the approach results in two main difficulties: (1) topics are not optimal for describing word senses and a mapping between topics and word senses must be provided to fully utilize the method; and (2) topics must be compared to find how they are related over time. Wijaya and Yeniterzi (2011) are, to the best of our knowledge, the first to work to automatically relate word senses over time by clustering on top of the extracted word senses. Though the authors make use of topics to detect word senses, the overall methodology seems promising but has thus far not been fully evaluated.

There has been little research in the field of *named entity evolution*, however, the field has recently begun to attract more interest. From the existing research, we found that the work of Berberich et al. (2009) is closest to our work as it does not rely on external resources or hyperlinks. Therefore, we will follow a similar approach to the one above and extend the methodology to avoid having to compare word contexts from arbitrary time points. How we go beyond the work of Berberich et al. is covered in detail in Chapter 7.1.

Chapter 5

Finding and Evaluating Word Senses

To help users interpret content in long-term archives we need methods to automatically find word sense evolution. The first, major component in such a pipeline are word senses. In this chapter, we explore an automatic method to find senses that does not rely on human input or resources like machine readable dictionaries and that can be applied to historical as well as modern texts. We therefore make use of **word sense discrimination** as a method to automatically find word senses. Applied on digitized collections, word sense discrimination algorithms have the potential of capturing old, as well as new meanings for a term, and hence aid in discovering word sense evolution covered in Chapter 6.

Word sense discrimination algorithms are used in many applications such as information retrieval, automatic machine translation and question answering systems. The aim of word sense discrimination is to extract, from term collections, coherent groups of terms where each group represents one word sense or meaning.

Word sense discrimination is a process of three major steps: (1) natural language processing (NLP) for the extraction of relevant terms and information; (2) co-occurrence graph creation or feature extraction; and (3) clustering. For all steps there exist different approaches. These approaches (and their combination) have mainly been evaluated on digitally born or manually corrected datasets like the British National Corpus or the New York Times.[1] When applying this process to old document collections, the quality of the resulting term clusters depends on the data quality of the collection as well as the sensitivity of the NLP tools to language changes.

The quality of a digitized collection depends on the digitization process used on the original text. The digitization process is called optical character recognition (OCR) and needs to deal with varying issues such as different paper qualities, dirty pages, and different kinds of fonts or manual annotations that cause errors. Errors need to be dealt with to improve quality and readability of the archive. Unfortunately, correction of OCR errors is often omitted for various reasons; automatic correction is not fully reliable while manual correction is expensive and time consuming.

Co-occurrence analysis and NLP tools are influenced by changes in language because of their dependency on language syntax and semantics. As an example, co-occurrence analysis based on a sliding window is less dependent on the time when the documents were created because it does not use any language specific information. Co-occurrence analysis based on grammatical relations however is more dependent on the collection and whether or not the used

1 `http://www.natcorp.ox.ac.uk/corpus/creating.xml`, Retrieved 2013-04-08

relations are present in the collection. NLP tools have similar dependencies. Trained on modern collections tasks like lemmatization or part-of-speech tagging can be influenced by language evolution.

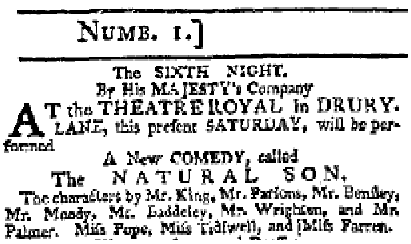
In this chapter we present an in-depth evaluation of the quality of word sense discrimination on historical documents when using NLP tools and evaluation technologies for today's language (Tahmasebi et al., 2013). If the results of word sense discrimination correctly correspond to word senses, also for historical data, we can use the word senses derived using word sense discrimination as a basis for word sense evolution.

5.1 Applicability to Historical Data

Because we want to apply our model to historical texts, we must first verify that the algorithms available for extracting word senses are applicable to text older than 50 years (more details on The Times Archive in Section 5.3.3). Part-of-speech taggers, dictionaries and word sense discrimination algorithms should be able to cope with documents that contain language for which they were not tailor made. In particular, word sense discrimination algorithms have been evaluated on more recent datasets; Schütze (1998) used an extract of the New York Times News Service from 1989 to 1990 for evaluations. Pantel and Lin (2002) as well as Dorow et al. (2005) used the British National Corpus, mainly consisting of documents from 1975 onwards. An important property of these datasets is their quality. They were either created digitally or have been manually corrected to make them error free; in some cases they are even manually sense tagged. Therefore, they are ideal candidates for research. However, the real world looks different and the corpus used in our work, The Times Archive, is a prominent example for this. In 2001, OCR technology was applied to process the images and resulted in an archive containing everything published in the newspaper during a 201-year period, from 1785 to 1985. One example shown in Figure 5.1 is a quotation from The Times Archive from January 1785 (The Times, 1785).

As we can see, there are several apparent difficulties. Firstly, the OCR processor had difficulties with fonts, in particular the letter *s* is interpreted as a *f* for certain time periods, e.g., *prefent* should be *present*. Secondly, the term *performited* is not present in a modern dictionary. This makes it difficult to determine if it is a valid, but outdated form or *performed* or an OCR error. Thirdly and most prominently, there are OCR errors present. The quotation in its original form is shown in Figure 5.1. It becomes obvious that *A T* and *TH EATR E* are falsely segmented by the processor for various reasons and *Nev* should be *New*. More detail on the types of OCR errors present in The Times Archive can be found in (Tahmasebi et al., 2013).

Because of a lack of previous evaluations, it is important to verify if word sense discrimination, applied on OCRed documents, can cope with different levels of data quality. For finding word sense evolutions it is also crucial to verify that the output of word sense discrimination applied on older datasets does indeed provide the expected output, that is, word senses.



NUMB. I.]

The SIXTH NIGHT.
By His MAJESTY's Company
AT the THEATRE ROYAL in DRURY-LANE, this preſent SATURDAY, will be performed
A New COMEDY, called
The NATURAL SON.
The characters by Mr. King, Mr. Parſons, Mr. Bensley, Mr. Moody, Mr. Baddeley, Mr. Wrighten, and Mr. Palmer. Miſs Pope, Miſs Tidſwell, and Miſs Farren.

The SIXTki NIGHT, By Pis MAJESTY's Company A T the TH EATR E ROYAL in DRURY. ALAIIE, this prefent SATURDAY, will be per- formited A Nev COMEDY, called The NATURAL SON. The chbaraersby Mr. Ying, Mr. Pfor.s, Mr. Betey, Mr. Moody, Mr. Baddeley, Mr. Wrighten, and Mr. Paner. Mifs Pope, Mifs Tiarwell, and jhlifs Farret.

Figure 5.1: A snapshot from The Times Archive, January 1785. To the right, the same text after OCR. Reason for many of the errors can be seen from the snapshot.

5.1.1 Considered Aspects of Evaluation

When evaluating the output of a word sense discrimination algorithm, applied on older texts, we need to be aware of three uncertainties which could affect the quality.

Firstly, our methods for extraction are trained on contemporary text collections. Therefore they may have difficulties recognizing terms which are no longer in use. If terms are not recognized by the natural language processors as nouns or noun phrases, they cannot be included in our clusters.

Secondly, the method for evaluation is significant. There are several methods for evaluating clusters found by word sense discrimination algorithms (Lin, 1998b; Pantel and Lin, 2002; Pedersen and Bruce, 1997; Resnik, 1995). The measures can be divided into two main categories. The first uses an external resource such as a dictionary or ontology for evaluation while the second relies upon a collection of sense tagged data. To our knowledge there are few or no digitized, sense tagged collections available from these periods. Therefore, we must either do the tagging ourselves or use a dictionary based method for evaluation. For the dictionary based method, the dictionary of choice can be of significance. Terms that are correctly spelled (considering the time that they were written), but not covered by a modern dictionary, will not be recognized as correct terms. As an example *infynyt*, *infinit*, *infinyte*, *infynit*, *infineit* are all spelling variations of the word *infinite* (Oxford University Press, 2000) which were correct at the time they were written, but would not be recognized by most modern dictionaries. Terms with outdated spellings present in the collection will decrease the assessed quality of the output.

Thirdly, the extracted word senses are affected by the quality of the text data. With a high proportion of OCR errors, terms containing errors will be recognized by neither the natural language processor nor the dictionary used for evaluation. Taking all the above into consideration, a low quality or quantity of clusters could indicate one of the following;

- terms have not been correctly extracted by the natural language processing step because they are outdated or contain OCR errors,
- terms are not recognized by the dictionary used for evaluation, or
- the chosen word sense discrimination algorithm is not suitable for use on data containing language older than a couple of decades.

When measuring the suitability of a certain word sense discrimination algorithm, all three features are important. We intend to investigate how the results of the word sense discrimination algorithm are affected by these three uncertainties.

5.1.2 OCR Quality of The Times Archive

To better understand the output of the word sense discrimination algorithm, we need to measure the distribution of OCR errors in the collection over time. The amount of OCR errors is approximated using a *dictionary recognition rate*. The dictionary recognition rate measures the portion of the text that is covered by a modern dictionary, in our case Aspell 0.60.6 (Atkinson, 2008) and WordNet 3.0 (Miller, 1995), from now on referred to as WordNet and Aspell. Details on the dictionaries and our use of them can be found in Section 5.3.1. We consider the proportion of OCR errors in a collection to be $OCRerror \approx 1 - f(t)$ where $f(t)$ is the dictionary recognition rate for a given dictionary and a time period t (in our case a year). Because terms with outdated spellings are not recognized by the dictionary, we consider this approximation, in addition to OCR errors, to also capture outdated terms. The text is cleaned and run through the previously mentioned dictionaries, one token at the time. Cleaning refers to removing heading and trailing non-letter characters while leaving any characters in a term, e.g., “&Bi1rd#!” becomes “Bi1rd”.

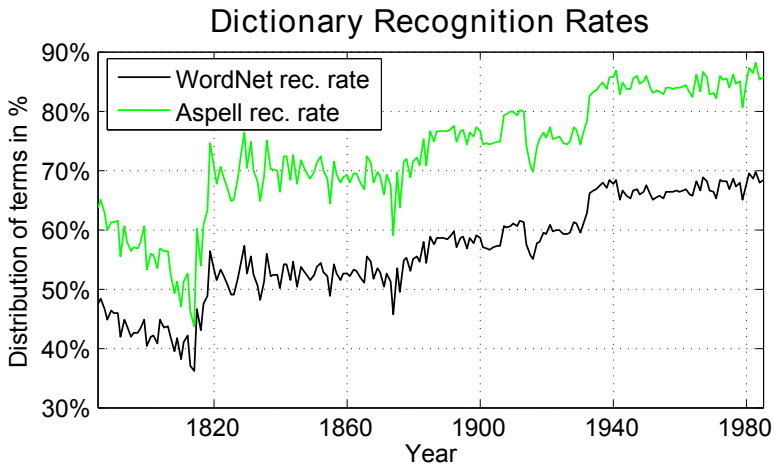


Figure 5.2: Dictionary Recognition Rates: percentages of terms covered by the dictionaries WordNet and Aspell from The Times Archive. WordNet contains no stopwords, adding stopwords to WordNet makes the recognition rate very similar to the recognition rate of Aspell.

It can be seen from Figure 5.2 that the two dictionaries differ in coverage. While Aspell covers between $44\% - 88\%$ of all terms in our collection, WordNet ranges from $36\% - 70\%$. On average, Aspell covers 73% whereas WordNet only covers 56% of all terms. When adding stopwords, WordNet displays almost the same mean and variation as Aspell giving us some indication of the amount of stopwords present in the collection over the years. On average 3 out of 10 terms are not recognized by any dictionary.

Because the collection covers the period from 1785 to 1985, and contains modern English after the normalization (i.e., consolidation of spelling and grammar to reduce variants), it is less likely to contain many outdated spelling variations of terms. Therefore we draw the conclusion that most of the terms which are not recognized by the dictionaries are in fact caused by OCR errors.

5.2 Word Sense Discrimination

Word sense discrimination is the task of automatically finding the sense classes of words present in a collection. The output of word sense discrimination are sets of terms that are discovered in the collection and describe word senses, for a more in-depth discussion on word senses see Section 6.1. This grouping of terms is derived from clustering and we therefore refer to such an automatically discovered sense as a **cluster**. Throughout this thesis we will use the terms **cluster**, **word sense**, and **sense** interchangeably.

In hard clustering an element can only appear in one cluster, while soft clustering allows each element to appear in several. Because each word can have several different meanings, soft clustering is more appropriate for word sense discrimination. The techniques can be further divided into two major groups, supervised and unsupervised. Because of the vast amount of data found in The Times Archive, we are using an unsupervised technique proposed by Dorow et al. (2005), called **curvature clustering**. For a motivation, please see Chapter 4.3. The curvature clustering is the core of the processing pipeline described next. Implementation details are given in Section 5.2.2.

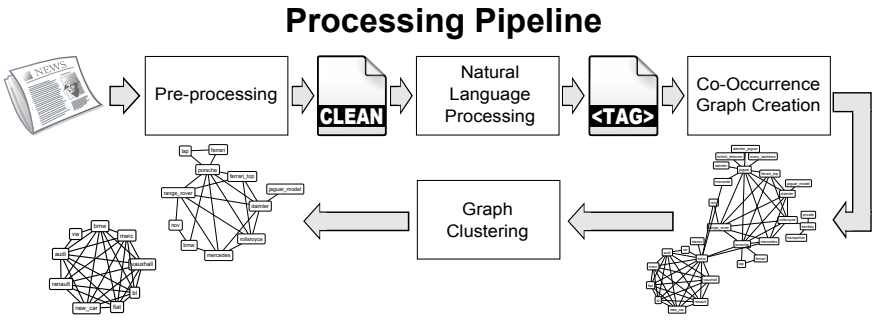


Figure 5.3: Overview of the word sense discrimination processing pipeline showing all four steps involved beginning with pre-processing and text cleaning.

5.2.1 Processing Pipeline for Word Sense Discrimination

The processing pipeline depicted in Figure 5.3 consists of four steps; pre-processing, natural language processing, creation of co-occurrence graph and clustering. These constitute the three major steps involved in word sense discrimination with the addition of pre-processing. Each step is performed for a separate subset of the collection. Each subset represents a time interval and the granularity can be chosen freely.

Text Cleaning / Pre-Processing

The first step towards finding word senses is to prepare the documents in the archive for the subsequent processing. This step includes among others extracting content from documents and performing an initial cleaning of the data. For collections that contain OCR errors we apply OCR error correction in addition to the cleaning step. Details on the OCR error correction algorithm can be found in Tahmasebi et al. (2013).

Natural Language Processing

The next step is to extract nouns and noun phrases from the cleaned text. To this end, it is first passed to a linguistic processor that uses a part-of-speech tagger to identify nouns. In addition, terms are lemmatized if a lemma can be derived. A lemma is a canonical form of a word, e.g., for nouns it is the singular form (*mice* has lemma *mouse*) and for verbs the infinitive form (*going* has the lemma *go*). Lemmas of identified nouns are added to a term list which is considered to be the **dictionary** corresponding to that particular subset. The lemmatized text is then given as input to a second linguistic processor to extract noun phrases. The noun phrases, as well as the remaining nouns for which the first part-of-speech tagger was not able to find lemmas, are placed in the dictionary.

Co-occurrence graph creation

After the natural language processing step, a **co-occurrence graph** is created. Typically the sliding window method is used for creating the graph but our initial experiments indicated that using the sliding window method in conjunction with the curvature clustering algorithm provide clusters corresponding to events rather than to word senses. Therefore we use following grammatical approach proposed by Dorow et al. (2005) instead.

Using the dictionary corresponding to a particular subset, the documents in the subset are searched for lists of nouns and noun phrases. Terms from the dictionary, that are found in

the text separated by an *and*, an *or* or a comma, are considered to be co-occurring. For example in the sequence *... instruments like cello, guitar and violin ...* the terms *cello*, *guitar* and *violin* are all co-occurring in the graph. Once the entire subset is processed, all co-occurrences are filtered. Only co-occurrences with a frequency above a certain threshold are kept. This procedure ensures that the level of noise is reduced and most spurious connections are removed.

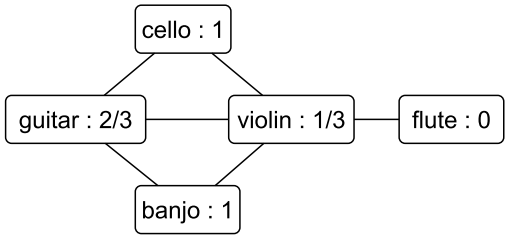


Figure 5.4: Graph to illustrate curvature value. *Nodes* are labeled with *name : curvature value*.

Graph clustering

The clustering step is the core step of word sense discrimination and takes place once the co-occurrence graph is created. The curvature clustering algorithm by Dorow et al. (2005) is used to cluster the graph. The algorithm calculates the clustering coefficient (Watts and Strogatz, 1998) of each node, also called **curvature value**, by counting the number of triangles that the node is involved in. The triangles, representing the interconnectedness of the node's neighbors, are normalized by the total number of possible triangles. Depicted in Figure 5.4 is a graph which illustrates the calculations of curvature values using different triangles. Node *cello* has a curvature value of 1 as it is involved in its only possible triangle $(guitar, cello, violin)$. The node *guitar* has a curvature value of $\frac{2}{3}$ because it is involved in two triangles $(guitar, violin, cello)$ and $(guitar, violin, banjo)$ out of its three possible triangles $(guitar, violin, cello)$, $(guitar, violin, banjo)$ and $(guitar, cello, banjo)$. The node $flute$ is not involved in any triangle and therefore its curvature value is 0.

After computing curvature values for each node, the algorithm removes nodes with a curvature value below a certain threshold. The low curvature nodes represent ambiguous nodes that are likely to connect parts of the graph that would otherwise not be connected (shown as red nodes in Figure 5.5 (a)). Once these nodes are removed, the remaining graph falls apart into connected components (shown as black nodes in Figure 5.5 (b)). The connected components, from now on referred to as clusters, are considered to be candidate word senses. In the final step each cluster is enriched with the nearest neighbors of its members. This way the clusters capture also the ambiguous terms and the algorithm is shown to handle both ambiguity as well as polysemy.

5.2.2 Implementation Details and Thresholds

When implementing the modules described above we rely in part on well established, freely available modules. Many suitable modules are available as Perl modules and if not otherwise mentioned, Perl is used for our pipeline.

For the natural language processing step we use two separate processors, namely TreeTagger (Schmid, 1994) and Lingua::EN::Tagger (Coburn, 2008). TreeTagger is used as the first processor to find lemmas. The second processor, Lingua::EN::Tagger is used to recognize noun

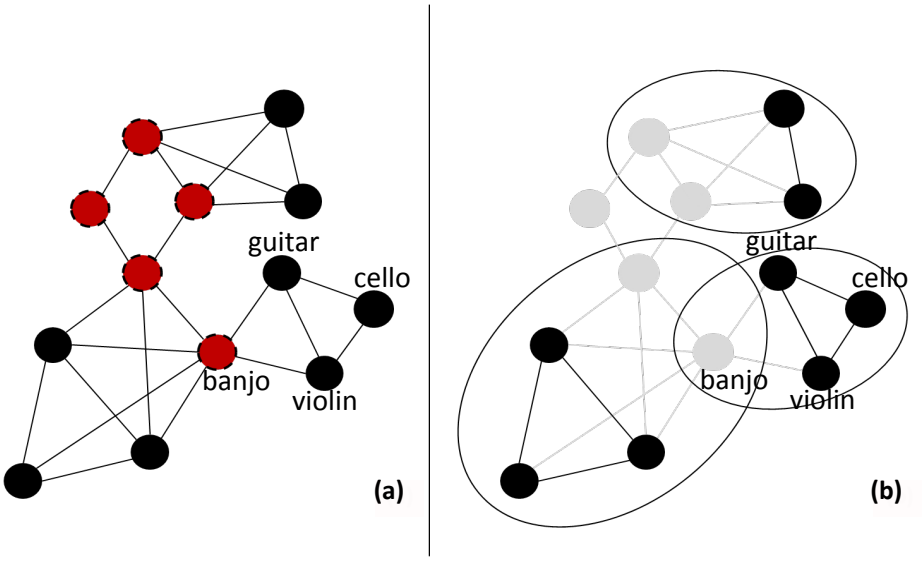


Figure 5.5: Illustrating the steps involved in the curvature clustering algorithm. *Nodes* in *red* (*dotted line*) (a) have a low curvature value. Once removed (b), the graph falls apart into connected components which constitute the core of the clusters.

phrases. We restrict nouns and noun phrases to length two in order to capture proper nouns like *New York* and noun phrases like *repressive environment* or *international call* but avoid noun phrases like *hiring of train*. Both processors were trained on statistics from the Penn Treebank. The Lingua Tagger applies a bigram Hidden Markov Model to assign the appropriate part-of-speech (POS) tag for a word based on the known POS tag for the given word and the POS tag assigned to its predecessor. TreeTagger uses a binary decision tree for assigning POS tags.

The co-occurrence graphs are created using a Java module. Once a full co-occurrence graph corresponding to an entire subset is created, it is filtered using a filtering threshold of 2, that means all co-occurrences with a frequency lower or equal to 2 are removed. Experiments have shown that this threshold provides good results for the majority of graphs obtained. The threshold used ensures that most of the noise is filtered out and that the resulting graphs are reasonable in size. In this thesis the same threshold is applied to all graphs. However, a further improvement is to establish the threshold based on the size of each graph. A larger graph should result in a higher filtering threshold.

For the clustering we chose the curvature threshold of 0.3. In Dorow et al. (2005) the threshold 0.5 was used while Dorow (2007) used the threshold 0.35. Since we aim to find word senses which evolve over time, we choose a slightly lower coefficient, expecting to get less strict word senses which are more likely to evolve over time. The lower coefficient should also provide us with more clusters as well as more terms in each cluster, that is, clusters that cover a larger portion of the collection.

5.3 Evaluation of Word Sense Discrimination

The aim of the evaluation is to analyze the applicability of NLP tools and word sense discrimination on long-term archives. To answer this question a number of measurements need to be taken like dictionary recognition rates, unique term rates and the quality of clusters. These measurements and the evaluation method are described in detail in Section 5.3.1.

For comparison purposes, in Section 5.3.2 we analyze the New York Times Annotated Corpus (NYTimes) Sandhaus (2008) as a reference corpus. Because of the method of creation, we consider the NYTimes corpus as a ground truth because it is considered to be error free and hence a good corpus for comparison. It provides a lower bound for how well the tools and algorithms should perform for a newspaper corpus.

To determine the effect of OCR errors, we first analyze The Times Archive in its original form in Chapter 5.3.3 and compare it with the results of the OCR corrected corpus in Chapter 5.3.4. Finally in Chapter 5.5 the results of the three analyses are used to compare and discuss the applicability of the NLP tools and word sense discrimination on long-term archives.

5.3.1 Evaluation method

In digitized collections, OCR errors are an obvious reason for having a large number of unique terms. Therefore, as a first measure of text quality we consider the *unique term rate* in a collection. As a second measure, we use the dictionary recognition rate, defined as the proportion of terms which can be recognized by a modern dictionary (see Section 5.1.2). Furthermore, we analyze the relation between OCR errors and the number of unique terms and investigate the implications of OCR errors on the output of the word sense discrimination algorithm.

To measure the *dictionary recognition rate* we use two dictionaries, namely WordNet and Aspell. WordNet contains about $147k$ unique single as well as compound terms but no stopwords. Aspell contains roughly $138k$ unique terms without any compound terms. Since we run each term separately through the dictionaries, we disregard compound terms in WordNet which leads to a reduced size of roughly $83k$. Because WordNet only contains lemmas we have to lemmatize each token before it can be passed to the dictionary. Therefore we use the stemmer from the MIT Java WordNet interface JWI Finlayson (Released under Creative Commons Attribution-NonCommerical Version 3.0 Unported License) which follows the WordNet stemmer implementation with one additional rule for terms ending on "-ful". Additionally, we add the WordNet "exception entries" to the dictionary. These entries contain mappings from irregular words to their corresponding lemmas which the stemmer cannot compute. Including these entries the WordNet dictionary contains about 89k terms. The Aspell dictionary contains lemmas as well as morphologies and names and therefore no lemmatization or "exception entries" are necessary.

To evaluate the *quality of the clusters*, that is, the correspondence between clusters and word senses, we use a method proposed by Pantel and Lin (2002) which relies on WordNet as a reference for word senses. The method compares the top k members of each cluster to WordNet senses. A cluster is said to correctly correspond to a WordNet sense S if the similarity between the top k members of the cluster and the sense S is above a given threshold. Following Pantel and Lin we chose similarity threshold 0.25. The clustering algorithm proposed by Pantel and Lin assigns to each cluster member a probability of belonging to the cluster, thus providing an intuitive way of choosing "top" members. The curvature clustering algorithm does not provide such probabilities and therefore we chose our k members randomly among the WordNet terms. Though it was not mentioned by Tahmasebi et al. (2010a), only clusters with at least two WordNet terms are evaluated resulting in $k \geq 2$. We found no statistically significant difference at a 95% level using $k = 4$ as in Pantel and Lin or $k \geq 2$ for any of the evaluated datasets in this thesis.

It should be noted that our precision differs from that used by Pantel and Lin, as well as Dorow, where precision is measured as the amount of correct clusters assigned to a word. Our aim is different than that of the mentioned works as we do not wish to automatically reconstruct a dictionary; instead we wish to check the quality of the clusters we have been able to find. Therefore, we do not consider cluster - term assignments but measure precision as the amount of clusters with at least two WordNet terms, which correctly correspond to a WordNet sense. Therefore our results are not directly comparable to that of Pantel and Lin or Dorow.

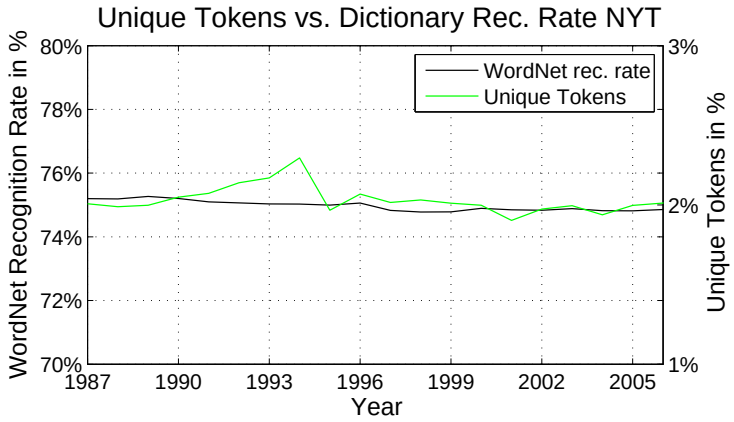


Figure 5.6: WordNet recognition rate versus unique terms in the NYTimes corpus. Both are stable over the entire period and we see no strong correlation between them.

The above measures are taken to analyze the impact of OCR errors on the quality of the found word senses. After correction, we expect that the dictionary recognition rates increases while the unique term rate should drop. As a consequence the graph sizes are expected to increase which should lead to a larger number of clusters.

5.3.2 New York Times as a Reference Corpus

The NYTimes corpus contains over 1.8 million articles written and published by the New York Times between January 1, 1987 and June 19, 2007. In our experiments we use the first 20 years and consider each as a separate dataset. Each year contains an average of 90'000 documents. The number of white space separated tokens range from 42.3 million in 1994 to 55.4 million in 2000. In total we found 1 billion tokens. When considering the length of an article, we count the number of terms in the article. The average length of an article is 539 terms with a steady increase from 490 tokens in 1987 to 591 in 2006.

We start by looking at the dictionary recognition rate for WordNet displayed in Figure 5.6. As we can see, the behavior is very steady over the entire dataset as can be expected when there is a large sample of text without OCR errors present. The Aspell recognition rate is not displayed. However, it follows roughly the same distribution as the WordNet recognition rate. For Aspell the mean value of the dictionary recognition rate is 96.5% with a standard variation of 0.1% and for WordNet the corresponding values are $75.0\% \pm 0.2\%$. Adding stopwords to the WordNet dictionary would increase the WordNet recognition rate to $94.6\% \pm 0.2\%$. This indicates that on average, 19.6% of the terms in the NYTimes corpus are stopwords. In Figure 5.6 we see also the proportion of unique terms in the NYTimes corpus. On average $2.0\% \pm 0.1\%$ of all terms are unique. There is no obvious reason for the slightly higher values for 1994 and this is also not reflected in the dictionary recognition rates. For this corpus we cannot see any relation between the two variables which is also indicated by a correlation value of 0.04.

We cluster the graphs using the curvature clustering algorithm with a clustering coefficient of 0.3. Each year we find an average of 1327 clusters and half of those contain at least two WordNet terms and thus participate in the evaluation. This translates to an average of 655 clusters per year. The precision for these clusters is high with an average of $0.85\% \pm 0.02\%$. To give some examples of clusters that pass the evaluation: the cluster "*car, minivan, truck, pickup*" is labeled with truck#n#1 which represents the first noun sense of *truck* in WordNet and the cluster "*son, brother, cousin, aunt*" is labeled with relative#n#1. Among clusters that

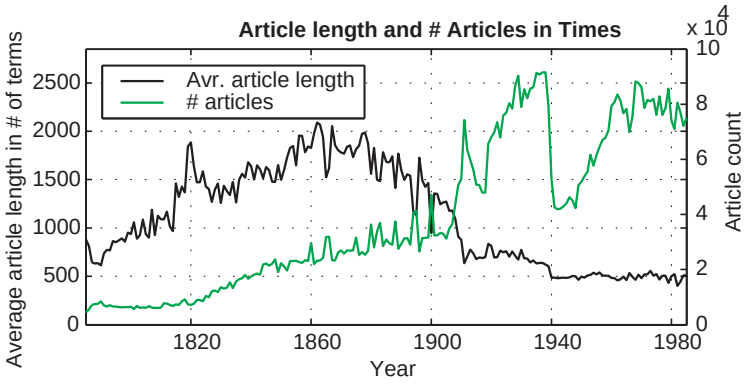


Figure 5.7: Number of articles and average length of articles in The Times Archive from 1785 to 1985. There are fewer articles with more terms before 1900 and more but shorter articles after.

could not be evaluated we find clusters like "*suleyman demirel, prime minister, bulent ecevit*" representing Turkish politicians, "*mickey rourke, bob hoskins, alan bate*" representing actors and "*eddie hunter, running, dennis bligen*" represent american football players who played as running backs.

5.3.3 The uncorrected Times Archive

We use The Times Archive as a sample of real world modern English. The corpus contains newspaper articles spanning from the year 1785 to 1985. The digitization process was started in the year 2001 when the collection was digitized from microfilm and OCR technology was applied to process the images. The resulting 201 years of data consist of between $4,363$ and $91,583$ articles. The number of space separated tokens range from 4 million tokens in 1785 to 68 million tokens in 1928. In total we found 7.1 billion tokens that translate into an average number of 35 million tokens per year.

The number of articles increases steadily during the first 100 years as shown in Figure 5.7. In the early 20th century the increase becomes more rapid and in 1911 we have almost double the number of articles as in 1905. The higher number of articles is affected during World War I (WWI) and World War II (WWII). In fact, in both periods the number of articles decreases heavily. The maximum number of articles is found in year 1938 when almost $92,000$ articles are published. We find that the average length of articles increase from 1785 until 1862 when a maximum of almost $2,100$ terms per article is measured. There follows a period of decrease which continues until 1940, then the average length of articles converges at roughly 500 terms per article.

Dictionary Recognition Rates

The recognition rates are on average 73% for Aspell and 56% for WordNet (see Figure 5.2 and Chapter 5.1.2). The large difference between 1814 and 1815 found for both dictionaries, is caused by the introduction of the steam press in end of 1814 (The Times, 1814). The decrease in quality from 1785 until 1815 is likely to have been caused by the logographic printing blocks used for printing during the period. They wore out quickly and had to be replaced frequently. As an anecdote, there is an editorial in The Times addressing this issue with an apology and a promise to attend to the problem.

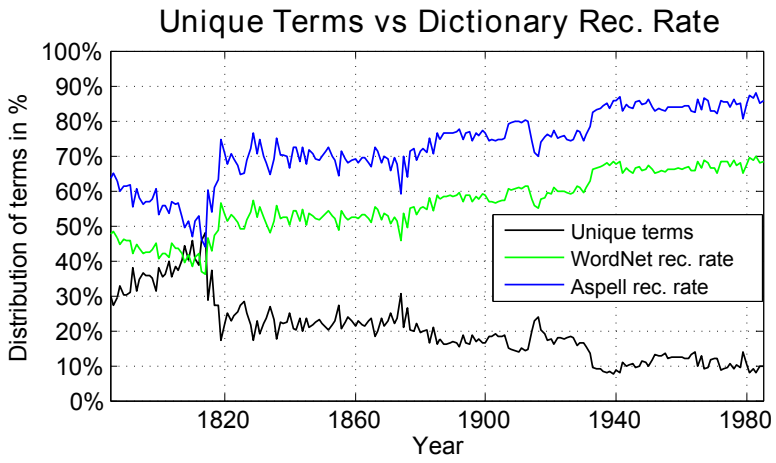


Figure 5.8: Percentage of unique terms in the collection compared to the dictionary recognition rates. Drops and peaks correspond well to each other. An increase in unique terms corresponds to a drop in the recognition rate.

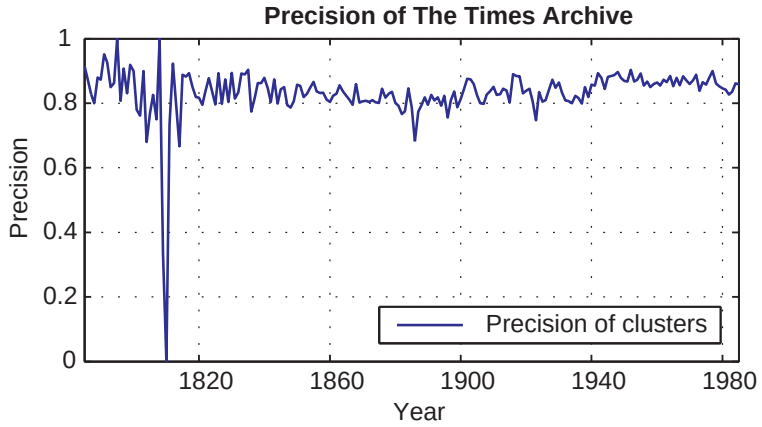


Figure 5.9: The results of the cluster evaluations for The Times Archive. On average 83% of all clusters correspond to WordNet senses. The low values around $1808 - 1810$ correspond to periods with very few (or zero) clusters due to low quality of the text.

Impact of OCR Errors

In Figure 5.8 we compare the percentage of unique terms from the collection against the WordNet and Aspell recognition rates. The result of the analysis is analog for Aspell recognition rate. We note that the graphs look like inverses of each other. In the first period, $1785 - 1814$, WordNet covers a decreasing amount of terms while during the same period the percentage of unique tokens increases. The results for period $1820 - 1880$ show a rather stable rate of unique terms as well as terms recognized by WordNet.

After concluding that the year 1874 and the period of WWI are likely to have a high percentage of OCR errors, we investigate how this affects the clusters. We find that the number of clusters dramatically decreases during these periods in comparison to the neighboring years, for example, in the years 1873 and 1875 there are 348 and 579 clusters respectively while in 1874 there are merely 91 clusters.

Cluster Analysis

After WWII we find that the number of clusters increases relative to the size of the graph. This indicates that the curvature clustering algorithm performs better with respect to the quantity of found clusters in this period. It is interesting to note that this coincides with the period of low percentage of unique terms.

Cluster Quality Evaluation

In Figure 5.9 we see the quality of clusters created using The Times Archive. On average 69% of all clusters contain more than two WordNet terms and can thus be evaluated. During the first period, up to 1840, we see much fluctuation. The extreme values of maximum or minimum precision for any one year occur before 1811 where there are very few clusters. In 1810 we have a total of two clusters of which none can be evaluated. In 1808 we have only one cluster and that cluster correctly corresponds to a WordNet sense and hence gives a precision of 1 for that year. Also in 1795 we have 17 measurable clusters out of which all pass the evaluation. The low amount of clusters corresponds to low quality text for which few terms can be extracted, leading to small graphs and extremely few clusters. We note that the period of high fluctuation in the precision corresponds well to the period of high fluctuation in the dictionary recognition rates. Considering also the extreme values for $1808 - 1810$ the average precision is 0.83 ± 0.08. However, removing these three years we have an average precision of 0.84 and a standard deviation of 0.04. The minimum precision is 0.67 and occurs for year 1814. In the period starting in 1940 and onwards we note that the average precision is higher (0.87) and that the standard deviation is lower (0.02). This period of a higher and more stable precision corresponds well to the period with a high and stable dictionary recognition rate for The Times Archive.

5.3.4 The Improved Times Archive

In our next set of experiments we applied our OCR error correction method presented in (Tahmasebi et al., 2013) on each year of The Times Archive and repeated the evaluation. We start by counting the number of tokens in each dataset. On average there was a decrease of $475,000$ tokens per year which is roughly a 1.6% decrease. This is mostly a consequence of reducing hyphenation errors, when merging terms we get fewer tokens. Hyphenation errors are caused by line and column breaks. In Figure 5.1 hyphenation errors are illustrated by the term *per-formited*.

In Figure 5.10 we see the proportion of unique tokens present in the archive after applying OCR error correction compared to before. It becomes clear that we dramatically reduce the amount of unique terms, from an average of $20.2\% \pm 8.9\%$ unique terms to $9.5\% \pm 4.6\%$ after the correction. This means we reduce the amount of unique terms to roughly half, a number that decreases in the later parts of the collection. We see that the shape of the curve is still the same even after the correction; however, the fluctuations are reduced. Increases and decreases are not as radical as before the correction. Again there is a distinction between the period up to 1940 and the period after. The average decrease up to 1940 is 12.2% while after 1940 the decrease is only 5.5%.

Dictionary Recognition Rates

If we look at the dictionary recognition rates we see a similar behavior as with the unique terms. There is an average increase of the WordNet recognition rate with $12.8\% \pm 4.0\%$ over the entire period making the total average $69\% \pm 4.4\%$. As seen in Figure 5.11 the fluctuations have decreased. The behavior of the graph is smoother with respect to large increases and decreases

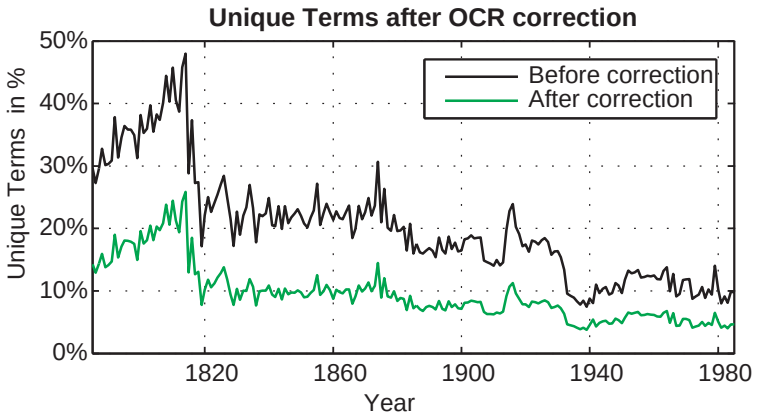


Figure 5.10: Decrease in unique tokens for The Times Archive after correcting OCR errors. On average there are over 10% fewer unique tokens per year and the variations between adjacent years are less extreme.

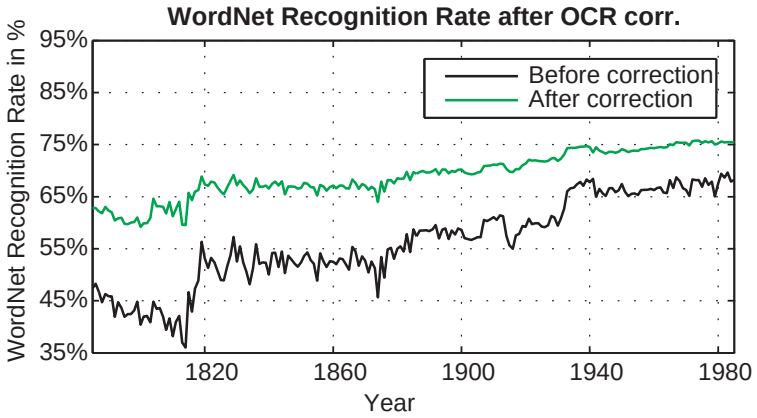


Figure 5.11: Increase in WordNet recognition rate for The Times Archive after correcting OCR errors. The average increase is 12.8% with a higher improvement for older data.

over a short period. If we measure average improvement we find that the improvement is almost double before 1940 in comparison to after 1940. The Aspell recognition rate is not displayed as a graph but follows the behavior of the WordNet recognition rate. The average increase over the entire period is $15.9\% \pm 5.2\%$ making the Aspell recognition rate $89\% \pm 4.8\%$. The OCR error correction results in a smoother graph. The improvement for the period before 1940 is almost twice as high as for the period after 1940. The Aspell recognition rate after 1940 is much closer to that of the NYTimes corpus, differing only by 3%.

Impact of OCR Errors

After examining Figure 5.10 and 5.11 we conclude that the relationship between dictionary recognition rate and unique terms in the collection is strong. To further investigate this relationship we plot the increase in WordNet recognition rate versus the inverse of the decrease of unique terms in Figure 5.12. As we can see they are strongly related. To measure their relationship we measure their correlation and find that they have a correlation value of -0.98544. The correlation between the increase in Aspell recognition rate and the unique terms is -0.98635. We use dictionary recognition rate as a measure for quality. Because the decrease in unique

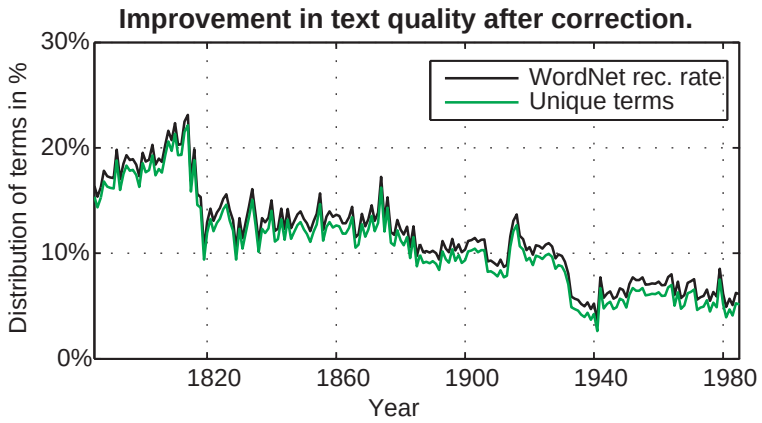


Figure 5.12: Improvements in WordNet recognition rate against unique terms after correcting OCR errors. For unique terms we use the absolute values of the decrease. The recognition rate increase corresponds to a very similar decrease in unique terms which shows that the two are very highly correlated.

terms is directly related to the improved quality of the collection, we can thus use the amount of unique terms as a quality measure for a collection. We see that during periods of high volume of errors - 1874, $1914-1918$, 1935 - the improvements in dictionary recognition rate are higher compared to neighboring years. However, the improvements are still not satisfactory as the cluster counts for these periods are not comparable to neighboring years (see Section 5.3.3).

Cluster Analysis

The average number of relations in the graph after applying OCR error correction to The Times Archive increases by 59%. The dependency between the unique relations in the graph and the number of nouns found by WordNet stays high. On average each graph contains 37% more terms than before the correction.The larger graphs result in a higher number of found clusters for the collection. On average we find 24% more clusters per year. The highest increase is during the early periods and in particular before 1815 where we increase the amount of clusters by 61%. In addition to having more clusters, the clusters cover more terms. Before 1815 we more than double the coverage and over the entire period the coverage increase by 75%.

Cluster Quality Evaluation

Considering the quality of the clusters we find a slight increase in precision from 0.83 to 0.85 ± 0.04. We note that after applying OCR error correction we can evaluate 68% of all clusters. A student t-test shows that the increase in precision is statistically significant with a 98% probability. This holds even if we ignore the extreme values for $1808-1810$ (see Figure 5.9) where we only have a handful of clusters. After correcting OCR errors, we find that the cluster quality for 1940 and onwards, with a precision of 0.86 ± 0.2, is now directly comparable with the cluster quality of the NYTimes corpus.

5.4 Word Sense Cluster Examples

To investigate if the clusters can be used for word sense tracking we manually investigate some clusters from Times as well as the NYTimes corpus. Due to repetitions, the clusters shown are

Table 5.1: Selected clusters and cluster members for the term *flight* from The Times Archive after error correction.

Year	**Cluster members**
1826	robson flight organ builder
1833	robson flight organ builder
1869	hurdle race, flight, yard, leaving
1895	hurdle race, flight, yard, steeplechase
1938	length, flight, spin, pace, capture
1957	direction, length, spin, flight, pace
1973	flight, riding, sailing, vino, free skiing
1978	flight, taverna accommodation, including yacht
1980	flight, visa, free board, week, pocket money, home
1984	flight, swimming pool, transfer, accommodation

Table 5.2: Selected clusters and cluster members for the term *computer* from the NYTimes collection.

Year	**Cluster members**
1988	word processing, graphics, datum base, data base, computer, spreadsheet
1990	telephone, printer, personal computer, facsimile machine, copier, computer
1990	game, computer, modem, fax machine, language translator
1992	television set, telephone, television, vcr, tropical plant, stereo, computer, camera
1993	multimedia, consumer, electronics, computer, appliance, communication
1995	television, video, cdrom, video, game, home shopping, qvc, personal computer
1996	software, telecommunication, personal computer, semiconductor, cellular phone
1999	printer, copier, computer, monitor, pc, scanner, disk drive, keyboard, fax machine
2000	laptop, telephone, internet, other wireless, wireless phone, cell phone, computer
2001	internet, commercial, video game, online, movie, television show, computer

sampled from all clusters mentioning each term and a limited number of terms are shown for each cluster. In both cluster sets we find that the number of terms in each cluster increases over time. It should be clear that clusters displayed here do not follow the evolution of each term as a whole, but as it was mentioned in Times or the NYTimes.

In Table 5.1 we see clusters for the term *flight* from Times. Among the displayed clusters it is clear that the senses for flight are several and mostly grouped together. Between 1819-1832 there are seven clusters that all refer to a company *Flight & Robson* which built church (finger) organs. Three decades later, between 1868-1895 there is a unit with 11 internal clusters that all refer to *hurdle races*. 1938 - 1966 the clusters refer to cricket; the terms in the clusters are referring to the ball. Starting from 1968 there are two units which correspond to the modern sense of flight as a means of travel, especially for holidays. The introduction of among others *hotel, transfer, accommodation*, differentiates the latter clusters from the earlier.

In Table 5.2 we see some selected clusters for the term *computer* from NYTimes. The first clusters reveal the computer as a tool for working with terms like *spreadsheet, database, printer, language translator*. Over time the clusters reveal the computer as an everyday tool for entertainment with terms like *game, home shopping, commercial, movie* and *communication*. We can also find terms that are now much less frequently used like *cdrom, vcr, qvc* and based on the surrounding terms infer meaning and context.

Table 5.3: Selected clusters and cluster members for the term *travel* from The Times Archive after correction.

Year	Cluster members
1803	literature, science, art, travel, voyage
1815	illustration, travels, science, travel, voyage, poetry, mile
1843	history, romance, memoir, travel, voyage, novel, biography
1867	travel, revival, colonial, foreign residence
1905	history, travel, book, mythology, biography
1906	full board, travel, best hotel
1924	town, apply, river, city, seaside, straight, travel, london
1928	sight, meal, reserved seat, superior hotel, sightseeing, travel
1966	loan, travel, good hotel, maintenance, fishing, tuition, hotel
1984	lanzarote, tenerife, sardinia, ravello, verona, malaga, travel

In Table 5.3 we see some selected clusters corresponding to the term *travel*. We can see that the concept of travel changes over time. In the 19th century it referred primarily to books and was not an everyday activity for ordinary people. Early 20th century the concept changes and travel becomes more common. With the introduction of terms like *sightseeing, full board, good hotel, fishing* including locations for travel, the concept of travel clearly becomes more concrete rather than something only available through books.

5.5 Discussion

Overall the results show that the used word sense discrimination algorithm can be applied to The Times Archive at least dating back as far as to the 19th century. We found that OCR errors have a major effect on the amount of clusters that can be extracted, however, the quality of the clusters remain high even with a high amount of OCR errors. Applying the OCR error correction method to The Times Archive significantly improves quality of the text and allows for an increased amount of extracted, high quality clusters. The corrected Times Archive, from 1940 and onwards differs very little from the NYTimes corpus with respect to dictionary recognition rates and unique terms and gives evidence for the performance of the OCR Key method used for correction.

In this section we will summarize the main results and provide a comparison between the three corpora; NYTimes corpus as a ground truth and The Times Archive before and after OCR error correction. Some key values can be found in Table 5.4.

Natural Language Processing Tools

The output of the word sense discrimination algorithm is affected by the number of nouns recognized as well as the number of those that can be lemmatized. We observe a high correlation between the number of lemmatized nouns found in a year and the size of the graph corresponding to that year. Therefore it is very important to find good natural language processing tools covering also historical texts. Currently 6 out of 10 WordNet nouns can be lemmatized. Before arriving at a definite conclusion on the performance of the lemmatizer used, it is important to verify the proportion of proper nouns among the remaining WordNet nouns as these mostly do not have lemmas. However, WordNet already contains relatively few proper nouns and hence we have reason to believe that for the lemmatizer, there is much room for improvement.

Table 5.4: Comparison between the three corpora with regards to Unique Term Rate, Dictionary Recognition Rates (with and without stopwords (SW)) and cluster precision.

Measure	**NYTimes**	**Times Archive**	**corr. Times Archive**
UTR	2.0%	20.2%	9.5%
WordNet DRR no SW	75%	56%	69%
WordNet DRR SW	95%	72%	87%
Aspell DRR	97%	73%	89%
Cluster precision	0.85	0.83	0.85

OCR errors

To measure the quality of text we use the dictionary recognition rate. A dictionary recognition rate is a quality measure for a text as well as the chosen dictionary. The more terms recognized, the better the quality of the text and suitability of the dictionary. The dictionary recognition rate is an average of 73% for Aspell and 56% for WordNet before making any corrections. We note that periods of high amounts of unique terms correspond to periods with low dictionary recognition rates. From Section 5.3.4 we learned that after correcting OCR errors, the decrease in unique terms is directly related to the increase of the dictionary recognition rate. We thus conclude that the amount of unique terms can be used as a quality measure for a collection; the fewer unique terms in a collection, the higher the dictionary recognition rate and thus the quality of the archive.

As a ground truth for the performance of the algorithms we use the NYTimes corpus. After applying the OCR error correction on The Times Archive, for the period $1940 - 1985$, we have an Aspell recognition rate that is 2.5% lower than that for the NYTimes corpus. It can also be observed that the unique term rate is 3.2% higher for the same period. We estimate the amount of remaining errors in The Times Archive for this period to be around $2.5\% - 3.2\%$ - a drop from $9\% - 14\%$ before the correction. By using the same reasoning as above we estimate for the period up to 1940 the amount of errors to be around $8.7\% - 9.4\%$ as compared to $20.6\% - 27\%$. The improvement is substantial and we can conclude that the OCR error correction is performing well.

After correcting OCR errors we find that there are slightly more WordNet nouns (3.6%) in The Times Archive (1940-1985) than in the NYTimes corpus. In general, we see the trend that WordNet performs better on the corrected Times Archive than on the ground truth. A possible explanation for this could be the OCR correction method. There is a tendency to correct unknown terms by replacing them with frequently used terms from the collection, which could lead to more terms recognized by the dictionary.

Clusters and Cluster Quality

Before starting these evaluations we worked under the assumption that the more recent the text, the more terms from the collections would be covered by the clusters. As a result of this we assumed that the quality of clusters would increase over time starting at a very low quality. Whilst the results supported the first hypothesis, we found evidence which disproved our assumption about the quality of clusters. We did, in fact, establish that clusters from the earlier period retain a high quality. On the other hand, the coverage for the same period is much lower; we have fewer clusters and fewer terms in each cluster. This indicates that, while our methods cover a much smaller portion of the data during the first 50 years, the covered portion keeps a high quality. We conclude that the quality of the clusters produced is not

significantly affected by the variation in the dictionary recognition rates while the coverage is highly affected. For the periods with larger dips in the graph (1814, 1874 and $1914 - 1918$) we have reason to assume that there are high rates of OCR errors as we have a significant decrease in the number of clusters compared with neighboring years.

Based on the similar performance of the Times Archive after OCR error correction to that of the ground truth, the NYTimes corpus, we conclude that word sense discrimination can successfully be applied to a collection of newspaper articles spanning from $1785 - 1985$ and that the resulting clusters correspond well to word senses.

5.6 Conclusions and Contributions

In this chapter we investigated whether the used word sense discrimination algorithm and NLP tools can be applied to a collection of modern English. The resulting word senses will serve as the basic elements used for detecting word sense evolution. Because many digitized collections contain OCR errors, we investigated the effects of OCR errors on the word senses found. For our evaluations we used The Times Archive (1785-1985) which contains OCR errors and compared it to the error free New York Times (1987-2007) as ground truth.

We applied an OCR error correction algorithm presented in Tahmasebi et al. (2013) and discovered that after correcting the errors present in The Times Archive, the performance of word sense discrimination is comparable to that of the ground truth. The clusters produced for content from the 19th century correspond well to word senses. However, although the clusters are of high quality, we found that the number of clusters is highly related to the amount of OCR errors; the more errors present, the fewer clusters can be found. Furthermore, we found that the natural language processing tools we used for recognizing part-of-speech and lemmatizing terms must be improved for high quality processing of historical data. However, based on the results presented, we conclude that the found word senses can be used as a basis for finding word sense evolution and thus language evolution.

The main contributions presented in this chapter are the following:

- We presented an in-depth analysis of dictionary recognition rates and unique term rates on two corpora that span a total of 221 years.[2] We showed that there is a strong correlation between the amount of unique term compared to the amount of terms that can be recognized by a modern dictionary.
- We applied word sense discrimination on The Times Archive and evaluated the resulting word sense clusters. We showed that the clusters correspond well to word senses (by means of synsets in WordNet) and that the correlation between the quality of the dataset and the number of word senses that can be found is stronger than the correlation between the quality of the dataset and the quality of the found word senses.
- We performed the same evaluation on an error free dataset, namely the New York Times Annotated Corpus, to provide a ground truth as an upper limit of the performance of the word sense discrimination algorithm.
- Finally, as an attempt to bridge the gap between the error prone content of The Times Archive and the New York Times, we applied an OCR error correction algorithm and we show that the gap between The Times Archive and the NYTimes can be bridged to some extent. We found that the lower amount of word sense clusters before the correction, are likely a consequence of the OCR errors, rather than the time or the language used in the collection and that the resulting clusters can be used for word sense tracking.

2 The timespan is 221 years because year 2007 was excluded for NYTimes since only part of the year was available.

5.6.1 Limitations and Future Work

There are several limitations to the word senses and the pipeline presented in this chapter. The first major limitations are the terms that are extracted. The number of noun and noun phrases recognized as well as the number of those that can be lemmatized are highly dependent on the age of the text and the amount of errors. We observe a high correlation between the number of lemmatized terms found in a year and the size of the graph corresponding to that year. Therefore it is very important to find better natural language processing tools covering also historical texts. The current level of lemmas found among the nouns, a maximum of 67%, is not sufficient for this purpose.

The second observation we make is that the longer the noun phrase, the larger the graph and harder and more time consuming to process. We perform noise filtering by removing all co-occurrences that have a frequency lower or equal to two. The filtering threshold is fixed and the same for all graphs, regardless of size. The same applies to the curvature clustering threshold that is set to 0.3. Depending on the size of the graph, this can sometimes lead to one or several larger clusters that represent more than one cluster and should be split. To improve the performance, future work should learn filtering threshold as well as the curvature clustering threshold for each time period. Larger clusters could also be re-clustered in a hierarchical fashion using a different threshold in order to extract the remaining clusters.

We find that even after correcting the OCR errors in Times there are relatively few clusters compared to the size of the dataset. Other word sense discrimination algorithms, perhaps in combination with other term extraction tools, could provide a larger number of clusters with a higher coverage of the terms and senses for each collection. In addition, a *localized* view on the collections can be taken to provide more fine grained senses, see Chapter 6.

To improve automatic extraction of word senses in the future, it is necessary to create tagged corpora so that natural language processing tools can be trained on historical data. In addition, we need methods for word sense discrimination that can handle a certain level of uncertainty in the form of misspelled and outdated terms, OCR errors and bad formatting of text.

Finally we believe that the type of dataset limits the number of word senses found. Newspaper archives are not optimal for finding word senses as the underlying data is highly dependent on media coverage of events. This results in that certain type of words and senses are not covered in the data simply because there were no events that required using these word senses. In future work, different types of data should be pooled to generate a more complete word sense repository and to avoid sparseness of word senses.

Chapter 6

Finding Word Sense Evolution

When interpreting the content of historical documents, knowledge of changed word senses play an important role. Without knowing that the meaning of a word has changed we might falsely place a more current meaning on the word and thus interpret the text wrongly. As an example, the phrase *an awesome concert* should be interpreted as a positive phrase today. The phrase *an awesome leader* in a text written two hundred years ago, should however be interpreted as a negative phrase only.

The interpretation of the term *awesome* depends on the time of writing and not on the context terms *concert* or *leader*. Therefore, we cannot regard this problem as a word senses disambiguation problem. Instead, we consider this problem as a manifestation of **word sense evolution** and in this chapter we investigate methods for automatically detecting such evolution.

For certain terms, like *awesome*, it can suffice to learn changes in the semantic orientation (Cook and Stevenson, 2010). The orientation would indicate that the term *awesome* went from being an exclusively negative term to including a positive meaning. This does, however, not reveal the full picture and learning what was meant with the term in two different periods of time is still an open question.

In this chapter we investigate methods for utilizing automatically extracted word senses to find word sense evolution given a collection of text. Our methods are unsupervised, that is, they require no manual input and are independent of external resources. We build on methods for automatically extracting word senses (see further Chapter 5) and, therefore, focus on nouns and noun phrases. We show the potential of our method on a set of terms that have experienced evolution in the past centuries.

6.1 Contributions and Relation to Existing Methods

Automatic word sense evolution has previously been tackled to a limited extent. Sagi et al. (2009) use context vectors and detect *broadening* and *narrowing* of a word's meaning by comparing the spread of the context vectors without discriminating between individual senses. Lau et al. (2012) used two different datasets to detect *novel word senses* as topics that are assigned to the latter dataset and not to the former. None of these track a word's individual senses and their temporal changes. Wijaya and Yeniterzi (2011) are, to the best of our knowledge, the only other work that documents automatic tracking of word senses over time. The presented method is in initial development and has not yet been fully evaluated. More details on these methods in Chapter 4.

In this thesis, we go beyond existing methods by presenting a model that has the potential of detecting novel and disappearing senses, broadening and narrowing of senses as well as splitting

and merging of senses. In addition to monitoring individual senses, we monitor the relation between senses and track the full development over time.

As a basis for our discussions we start with some definitions related to word senses and their evolution. As before, we define a **term** as a single or multi-word noun or noun phrase. The rest of the definitions are borrowed from Cooper (2005).

A **word sense** is one **meaning** of a word. Though Kilgarriff (1997) states that "*word senses exist only relative to a task*" like word sense disambiguation, in this thesis we consider word senses and meanings to be synonymous. We further borrow the definition from Cooper (2005) regarding concepts. Two meanings of a given word correspond to the same **concept** if and only if they could inspire the same new senses by association.

Hence, we consider a concept c to group one or more word senses s for a word w so that $c = \{s_1, s_2, \ldots\}$. For each word, the number of concepts is less than or equal to the number of senses because, theoretically, all senses can be unrelated and hence give rise to individual concepts.

6.1.1 Language Evolution

Cooper provides a definition of language evolution (Definition 6.1). This definition excludes term to term evolution but provides a good base for word sense evolution.

Definition 6.1 (Language Evolution)**.** Let L_D be a language defined by the set of word sense pairs $< w, s_w >$ in a dictionary D. Then the evolution of L_D is considered a stochastic process where each state is one of the following:

(a) Elimination of a word sense.
(b) Introduction of a new word.
(c) Added sense for existing word.

Cooper (2005) assumes that each newly introduced word has only one sense and thus state (b) extends L_D with one word sense pair $< w, s_w >$. Furthermore, the definition states that the elimination of a word sense s_w from a word w removes the word sense pair $< w, s_w >$ from the dictionary. We deduce that if w has more senses, the number of words in the dictionary does not change. If, however, s_w is the only sense for w then the number of words in the dictionary is decreased by one. When a new word is introduced the number of words in the dictionary is increased by one.

States (a) and (c) can both affect the number of concepts present for a word. Elimination of a word sense s_w can eliminate one concept under the condition that the concept only contains s_w. Step (c) can either add a sense to an existing concept, thus keeping the number of concepts constant, or create a new concept, thus increasing the number of concepts by one.

We can infer two things from Definition 6.1:

Inf. 1: Senses are building blocks that cannot change themselves, they can only be *active* or *passive* for a word.

Inf. 2: If the senses belonging to a concept are changed by means of (a) elimination or (c) addition then there has been concept evolution.

6.1.2 Word Sense Evolution

We make use of the above inferences to map the definitions to our work. In our definition of word sense evolution the language of L_D maps to a terminology snapshot. This snapshot contains all term concept graphs corresponding to all terms in D. Each term–concept pair is

annotated with validity period (see Section 3.2). We make use of *term concept* graphs rather than *term sense* graphs as each concept can represent single senses as well as several, grouped senses.

Furthermore, we conclude that while Definition 6.1 is memoryless, a terminology snapshot also has a temporal dimension. Using this perspective the elimination of a sense from the dictionary does not map to a change in the terminology snapshot. The corresponding term–concept pair remains in the term concept graph but is no longer updated. The introduction of a new word adds a new term concept graph to the terminology snapshot. The newly added term concept graph consists of one concept with a single sense. Formally we define word sense evolution as follows:

Definition 6.2 (Word Sense Evolution)**.** Let TS_D be a terminology snapshot defined by the set of term concept graphs over a collection of documents D. Then word sense evolution is considered a stochastic process where each state is one of the following:

(a) Introduction of a new word and thus a new term concept graph.
(b) Added sense for existing word represented by an added concept in the term concept graph.
(c) Changed concept for existing word.

States (a) and (b) in the above definition are analog to states (b) and (c) in Definition 6.1. State (c) in the above definition is considered to be an added or eliminated sense from an existing concept of a word and builds on Inf. 2. In addition, state (c) captures broadening and narrowing of senses. The main difference between Cooper's definition of language evolution and our definition of word sense evolution is the notion of time.

We defined word sense evolution as changes in the terminology snapshot. The evolution of one term w is expressed as changes in the corresponding term concept graph TCG_w. At each point in time, two term concept graphs corresponding to w are compared and merged to create one, merged term concept graph which contains information regarding both time points. In addition, we relate all senses present in the merged term concept graph to find groups of senses that represent concepts. In the next section we present the definitions needed for merging term concept graphs. We then present our methodology for detecting word sense evolution.

6.2 Definitions and Terminology

In this section we build on the definitions presented in Chapter 3.1.

In addition, we define a **unit** $u_{(t_i,t_k)} := u_i$ as an ordered sequence of clusters $\{c_1^{t_i}, c_2^{t_{i+1}}, \ldots, c_n^{t_k}\}$ such that each cluster comes from a distinct, time period t_j where $t_i \leq t_j \leq t_k$. We allow time gaps between the clusters, i.e., $t_k \geq t_j + 1$, in order to capture senses that have lost in popularity or are underrepresented for a period of time. The clusters involved in the unit are called **internal clusters**.

Each unit is represented by a set of terms that constitute the **unit representative** defined as u_r. The unit representative contains a set of terms that can be used to represent all internal clusters.

We define a **single unit** to be a unit consisting of only one cluster where the cluster terms are used as the unit representative.

We measure **similarity between units** as similarity between the unit representatives. Two units are considered similar if their similarity, for example measured by means of set similarity, is larger than a constant α also called *minUnitSim*. Similarity between clusters is measured by means of unit similarity where each cluster is represented as a single unit.

All units related to term w are considered as the set U_w. For each unit in U_w we consider three **relations**:

- $=$ If $sim(u_i, u_j) > \beta$ for $u_i, u_j \in U_w$ we consider the units to represent the **same sense** and denote the relation $=$.
- $\backsim$ If $\alpha \leq sim(u_i, u_j) \leq \beta$ for $u_i, u_j \in U_w$ we consider the units to represent the same but **evolved sense** (e.g., by means of broadening) and denote the relation $\backsim$.
- $\neq$ If $sim(u_i, u_j) < \alpha$ for $u_i, u_j \in U_w$ we consider the units to represent completely **different senses** and denote the relation $\neq$.

If two units u_i and u_j have overlapping time spans and have the relation $\backsim$ then we consider the units to represent **polysemic** senses. We allow for partial relations to capture merging and splitting of units. If two units u_i and u_j are related in an interval in the beginning of their time span and not during the entire period, we consider the units to be **splitting** units. If the inverse occurs we consider the units to be **merging**.

A **path** $path(t_i, t_k)$ is an ordered sequence of units $\{u_1^{t_i}, u_2^{t_j}, \ldots, u_n^{t_m}\}$ such that all units are pairwise related or partially related, i.e., $sim(u_i, u_j) \geq \alpha$. As within each unit, we allow time gaps between the units, i.e., $t_m \geq t_k + 1$, in order to capture senses that have lost in popularity or are underrepresented for a period of time and then re-appear. A path represents all evolution within one concept of a term. Different paths for a term represent **homonymic** senses.

To bring all definitions together, we consider a *sense* s_w at one point in time to be represented by a *cluster*. A *unit* u_w captures a sense s_w over a period of time and allows for broadening, narrowing and evolving within s_w. These operations correspond to state c) in Definition 6.2 and are captured by allowing $=$ and $\backsim$ relations between all internal clusters of u_w. A *path* corresponds to a concept by grouping all units in U_w that have $=$ or $\backsim$ relations. Within a path we allow for merging and splitting of units. A novel word sense for term w is a word sense s'_w with $\neq$ relation to all existing word senses of w corresponding to state b) in Definition 6.2.

6.3 Methodology

The key to finding word sense evolution is to find changes in the term concept graph TCG_w corresponding to a term w. Changes can only occur over time and therefore, to capture these changes, term concept graphs from different time points must be merged.

At each time point t_i we assume the existence of a term concept graph that contains all known evolution of a word until t_{i-1}. We call such a graph a **merged term concept graph**. In order to detect evolution we must compare the term concept graph corresponding to time t_i with the merged one. We then update the merged term concept graph to contain also the information about t_i.

Because we iteratively add one term concept graph to the merged graph our merging function needs only to merge two graphs at the time. The merging corresponds to the merging function ψ, discussed in Chapter 3, Eq. 3.6.

Henceforth we consider each term concept graph to consist of units. A term concept graph representing period t_i consists of single units while a merged term concept graph consists of units that represent word senses over a period of time. The process of merging two term concept graphs TCG_i and TCG_j is then the following:

1. Measure similarity between all units in TCG_i and TCG_j
2. For each unit $u_i \in TCG_i$, merge with the most similar unit in $u_j \in TCG_j$ if the relation between u_i and u_j is $=$ or $\backsim$.
3. The merged term concept graph $TCG_{[i,j]}$ contains information about both time points.

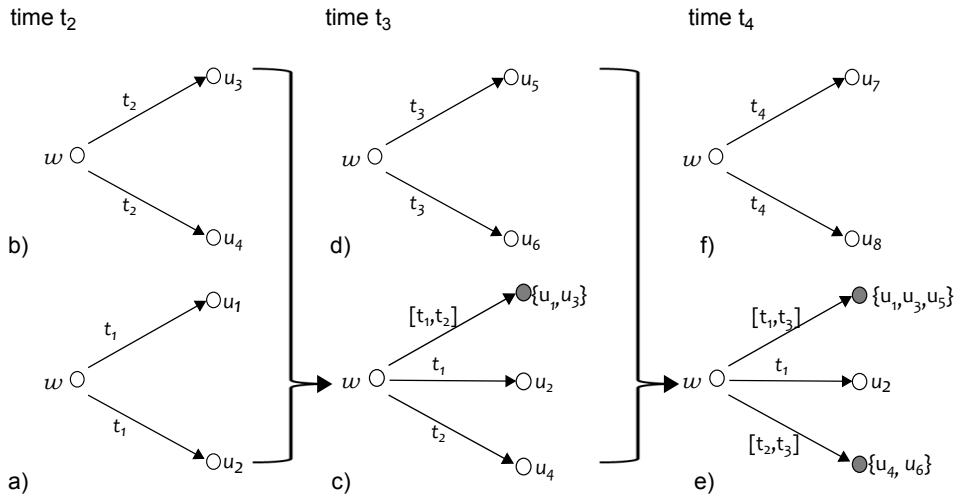


Figure 6.1: Iteration 1. TCG's from a) and b) are merged into TCG shown in c). Iteration 2. TCG in c) is merged with TCG in d) into TCG shown in e). Edges are updated with corresponding time periods. *Nodes* in *gray* show merged units, *nodes* in *white* show single units.

In Figure 6.1 we can follow the process of merging. At time point t_1 there exists only one term concept graph TCG_{t_1} and there is no need for merging. At time point t_2 we have a second term concept graph TCG_{t_2} and want to merge the two. The resulting term concept graph $TCG_{[t_1,t_2]}$ is shown in c). There is now a merged unit $\{u_1, u_3\}$ and two single units u_2 and u_4. The merged unit is represented by its unit representative. In the next iteration, a new term concept graph d) is found and merged with $TCG_{[t_1,t_2]}$. The resulting term concept graph $TCG_{[t_1,t_3]}$ is found in e) and the process continues in the same way whenever a new term concept graph is found.

As a final step in the merging we construct paths by relating senses in the last, merged term concept graph. This means relating $\{u_1, u_3, u_5\}$, u_2 and $\{u_4, u_6\}$. Broadening and narrowing of senses are captured within the merged units. The paths capture merging and splitting of senses, polysemy as well as homonymy.

Example 1

Let w represent the term *tape* and u_1 and u_3 be single units with the following terms u_1 = {*stereo, cassette, tape, radio, record*} and u_3 = {*video, cassette, record, tape*}. Because of the high number of overlapping terms u_1 and u_3 are merged into $\{u_1, u_3\}$. At time t_4 the unit u_5 = {*television, record, tape, video, book, film, magazine, video industry*} is merged with $\{u_1, u_3\}$ resulting in $\{u_1, u_3, u_5\}$. An example of broadening is found when unit u_5 is merged with $\{u_1, u_3\}$ where the single term has a broader sense than the merged unit. The clusters and units in this example are taken from from Figure 6.2, Section 6.4.3.

6.3.1 Measuring Unit Similarity

A central part of word sense tracking is the comparison of clusters and units. For merging two term concept graphs we must have a method for comparing two units u_i and u_j. Because the method used to obtain word senses produces results that are highly dependent on the underlying data, a word's senses can differ between different collections without there being any significant change in its senses. This is a result of the collection that is used to approximate word senses. Consider the following: a document collection D is repeatedly split into two random partitions D_1 and D_2. The word senses created for each part will differ from each other

and a word sense in D_1 would most likely be a variation and not a replica of the corresponding word sense in D_2. Therefore, we need a similarity measure that reduces this effect and thus reduces noise. In addition, we want to capture broadening and narrowing of word senses. This would correspond to senses being similar in certain parts, with the added constraint that one sense is significantly larger.

To get a similarity measure that respects these requirements we make use of *almost* matches. We define the similarity between two units u_i and u_j as a modified Jaccard similarity between the unit representatives u_{r_i} and u_{r_j} under two conditions.

$$\text{Condition I. } |u_{r_i} \cap u_{r_j}| \geq |u_{r_i}| - 1 \text{ or } |u_{r_j}| - 1$$

$$\text{Condition II. } |u_{r_i}| > 2 \cdot |u_{r_j}| \text{ or } |u_{r_j}| > 2 \cdot |u_{r_i}|$$

The similarity between two units is then defined as follow:

$$sim(u_i, u_j) = \begin{cases} 1 & \text{if I. holds} \\ \gamma & \text{if I. and II. holds} \\ \frac{|u_{r_i} \cap u_{r_j}|}{|u_{r_i} \cup u_{r_j}|} & \text{otherwise.} \end{cases} \tag{6.1}$$

where $\frac{1}{2} < \gamma < 1$. If one unit representative is a direct (or almost direct) subset of the other, the similarity between the units is the maximum possible similarity. If the above holds and one unit representative is double the size of the other, we scale up the similarity values under the condition that γ is smaller than the maximum similarity. We use this to capture senses that have broadened or narrowed between two time points and hence are similar in certain parts while being significantly larger or smaller in size. Finally, if none of the above conditions hold, the similarity between two units is the Jaccard similarity of their representatives as defined in Equation 3.1.1. We choose $\gamma < 1$ to mark that the broadening and narrowing relation is weaker than the (almost) direct subset relation. In addition we choose $\gamma > \frac{1}{2}$ to increase similarities values compared to Jaccard similarity. Assume the following: u_{r_i} is a direct subset of u_{r_j} and $|u_{r_j}| > 2 * |u_{r_i}|$. Because of the direct subset relation, their overlap will equal the smaller set and their union will equal the larger set, i.e., $u_{r_i} \cap u_{r_j} = u_{r_i}$ and $u_{r_i} \cup u_{r_j} = u_{r_j}$. The Jaccard similarity then equals

$$\texttt{JaccardSim}(u_{r_i}, u_{r_j}) = \frac{|u_{r_i} \cap u_{r_j}|}{|u_{r_i} \cup u_{r_j}|} = \frac{|u_{r_i}|}{|u_{r_j}|} \leq \frac{|u_{r_i}|}{|2 * u_{r_i}|} \leq \frac{1}{2}.$$

Thus, to increase similarity values between units that represent broadening and narrowing senses, we choose to set $\gamma > \frac{1}{2}$.

In order to measure term overlap between sets u_{r_i} and u_{r_j}, we need to define equality between terms $w_i \in u_{r_i}$ and $w_j \in u_{r_j}$. A straight forward way to define equality is to require an exact match between w_i and w_j. This definition is however very restrictive. Especially given that we want to compare senses that span a long period of time and that our senses contain nouns and noun phrases. Therefore, we consider equality between terms in two ways; (1) full match; and (2) partial match. Partial matches between terms are only considered if a term w_i does not have a full match w_j in set u_{r_i}. We split all terms into their parts $w_i = w_1\ w_2$ and a term w_i is only accepted as a partial match to w_j if any w_1 or w_2 is a suffix or prefix in w_j. Using partial matches we are able to capture similarity between sets of terms that contain for example, *motor car* and *motorcar* as well as *monitor* and *color monitor* but avoid matching *rave* and *gravel*.

Example 2 To illustrate the utility of the similarity measure we continue the example from above and measure similarity between $u_1 = \{$*stereo, cassette, tape, radio, record*$\}$ and $u_3 = \{$*video, cassette, record, tape*$\}$. Because three of four terms from u_3 can be found in u_1 we consider the similarity to equal 1. In comparison, the Jaccard similarity between u_1 and u_3 is $3/6 = 0.5$. The similarity between u_3 and $u_5 = \{$*television, record, tape, video, book, film, magazine, video industry*$\}$ is equal to $\gamma > \frac{1}{2}$ while the corresponding Jaccard similarity is $3/9 = 0.33$.

6.3.2 Merging Units

In the merging step, pairs of units that qualify for merging are merged into one. In each iteration of the merging we calculate the similarity between the units $u \in TCG_{[t_1,t_k]}$ and $v \in TCG_{t_{k+1}}$. For each unit v we measure similarity to all units u, normalized such that the sum of the similarities between v and all units u amounts to 1. We then choose to merge with the most similar unit assuming that the unit similarity is larger than *minUnitSim*. If the similarity of v and several unit u is larger than *minUnitSim* we know that there are polysemic concepts in $TCG_{[t_1,t_k]}$. By uniquely assigning and merging v with at most one concept u we implicitly perform disambiguation between the related senses.[1] Once a unit has been created we need to represent the clusters involved in the unit. A unit representation should highlight the important and discriminating terms from the underlying clusters and capture the essence of the meaning presented by the unit.

To find a good representation for two or more clusters we make use of *local curvature values* (*lcurv*) and *sense participation rate* (*part*). First we calculate the local curvature value of each node in one cluster. This means, from the original co-occurrence graph we find how terms are linked. We then calculate the curvature value (Section 5.2.1) using only the nodes and links present in the cluster, ignoring triangles where nodes are not members in the cluster. We measure concept participation rate as the amount of clusters where the term is present. Both *lcurv* and *part* are normalized and averaged over all internal clusters of a unit.

The local curvature value measures the centrality of a term in a cluster while the participation rate measures the consistent contribution of a term to the overall sense. To allow a term in the unit representative, the term must be highly central in its cluster, be highly consistent in the unit or have an average high rate of both. By comparing a new unit to the representative of a merged unit using the adapted Jaccard similarity, we allow merged units to capture broadening and narrowing of senses. Each time a new concept is merged, we update the unit representation, and therefore we allow a slow shift of the unit representatives capturing updated and evolved senses.

Example 3 We continue our example from previous section: In order to merge the single units we must start by finding the local curvature values for each term. For the units in our example the curvature values are $u_1 = \{$*stereo 1.0, cassette 1.0, tape 0.66, radio 0.66, record 0.5*$\}$ and $u_3 = \{$*video 1.0, cassette 1.0, record 0.66, tape 0.66*$\}$. For the merged unit $\{u_1, u_3\}$, all terms are placed in the unit representative except *radio* because it has low values for both measures: $lcurv = 0.33$ and $part = 0.5$. At time t_4 the unit $u_5 = \{$*television 1.0, record 0.83, tape 0.83, video 0.66, book 0.5, film 0.4, magazine 0.0, video_industry 0.0*$\}$ is compared to the unit representative of $\{u_1, u_3\}$.

6.3.3 Detecting Relations between Units

To capture the relation between units and to group senses into concepts we make use of paths. These paths follow ideas from Mei and Zhai (2005) where they track themes over time using

1 This one-to-one assignment is a simplification that risks merging locally optimal units, however, multiple assignments increases the difficulty of evaluating and visualizing the results and is left as future work.

theme evolutionary graphs. A theme is expressed using a probability distribution over all words in a corpus with probabilities for each word to belong to a theme. Mei and Zhai (2005) use Kullback-Leibler divergence (Kullback and Leibler, 1951) to compare themes and create theme evolution threads. These theme evolution threads can be seen analog with our paths with different methods to create and compare elements in a path.

After all term concept graphs have been merged, the final term concept graph consists of units that represent individual senses over time. In order to find concepts we must group units such that the units are related. We do this by comparing each unit u to all other units that start at the same or later time point. If the $=$ or $\backsim$ relation hold, the units are considered related. We start by looking for partial relations to capture merging and splitting of senses. We do this by comparing internal clusters and require similarity among the first two clusters for splitting and the last to clusters for merging. We allow time gaps between units to capture relations between underrepresented senses. If we find that two units should be merged or split, we compare the unit representatives to capture polysemy. If two units are similar in their first or last internal clusters as well as similar over all, then we consider the units to be polysemous.

All units that have relations $=$ or $\backsim$ are placed in a path. We construct our paths such that each path is a tree with one unit as the root. In order to avoid cycles we require, for each path, that if v is a child of u then v cannot be a grandchild of u. Paths represent concepts where all senses are fully or partially related. Different paths for a word represent unrelated senses.

6.4 Experiments

The aim of our experiments is to determine the quality and degree to which word sense evolution can be found using our proposed methodology. There exists no standard datasets or evaluation metrics for automatically found word sense evolution. In particular because the outcome is specific to the used collection and the evaluation must be made with the collection and place of publication in mind. The creation of such a dataset and evaluation metrics requires experts with linguistic and historical knowledge of the collection and lies outside of the scope for this thesis. Instead, in our experiments, we opt for a simplified evaluation. We evaluate the discovered instances of evolution for each term by comparing to the general knowledge of the term, and do not take completeness into account.

As a testset, we manually choose a set of terms which we know have experienced evolution during the past centuries. Using available resources we find the main evolution for each term and evaluate the automatically discovered instances of evolution against the manually found counterparts. To increase the amount of clusters, we create local clusters and merge with the clusters found in Chapter 5. Finally, we create and merge term concept graphs and relate the senses to see if, and to which extent, these can reflect the expected evolution.

6.4.1 Dataset and Localized Clusters

There exists no standard dataset for automatic word sense evolution detection and therefore we manually choose a set of 10 terms that we know have experienced evolution. Table 6.2 shows the details for each term and the type of evolution that can be found based on a manual assessment. To extend the time span we consider both Times and NYTimes consecutively. With the exception of 1986, we get a collection that covers 222 years, from 1785 to 2007. For some terms, the actual time span is shorter since the terms were not introduced, or not used in the dataset, until after 1785.

In Chapter 5 we concluded that the pipeline used for extracting word senses provided clusters which (1) correspond to word senses also for historical data; and (2) are suitable for word

Table 6.1: Description of dataset for word sense evolution detection . Definitions derived from dictionary.com, Wikipedia and Oxford English Dictionary.

Term	**Definition**	**# Clusters**
tape	A narrow woven strip of stout linen, cotton, silk, or other textile Magnetic tape for storing music	53
aeroplane	An aircraft which relies on aerodynamic lift for flight; a heavier-than-air aircraft	50
rock	Stone / Any of various styles of pop music having a heavy beat	160
travel	To go from one place to another, as on a trip; journey	290
gay	U.S. slang. (a) Of a person: homosexual; (b) (of a place, milieu, way of life, etc.) of or relating to homosexuals.	73
tank	An armored military vehicle moving on a tracked carriage and mounted with a gun, designed for use in rough terrain.	153
cool	Marked by calm self-control.	62
flight	The act of flying through the air by means of wings.	61
mouse	An animal, small rodent A hand-held device used to control the cursor movement	59
telephone	A device used to communicate over distances	491

sense evolution tracking. In this chapter we make use of the same pipeline with a modified view on the collections. The word senses found in Chapter 5 can be seen as global clusters; given one year and all documents published in that year, terms are extracted, co-occurrence graphs created and clusters derived. Comparably few terms were covered in these clusters. To overcome this sparseness of clusters, in this chapter we create clusters in a localized fashion; for each term w we extract all documents that mention w corresponding to a year. The rest follows the same pipeline as Chapter 5.

Using the localized collections to extract word senses provides us with more clusters for each term, on average only 38% of the clusters are global clusters. Definitions and the total number of clusters for each term can be found in Table 6.1. Table 6.2 summarizes the main evolution for the terms.

6.4.2 Experimental Setup

Most of the implementation details and thresholds are inherited from the pipeline for finding word senses, see Chapter 5.3.1. In addition we set the minimum unit similarity *minUnitSim*, also considered $\alpha = 0.3$.

In the merging process we assume $u \in TCG_{[t_1,t_k]}$ to be a node in the merged term concept graph and $v \in TCG_{t_{k+1}}$ to be a node in the newest term concept graph. We order all node pairs (u,v), that have relation $=$ or $\backsim$ in a list in decreasing order of similarity between u and v. At each point in time, we choose the highest ranked pair and merge the units. If a unit $u' \in TCG_{[t_1,t_k]}$ is chosen we remove all pairs (u',*) from the list and no other node v can be merged with u'. If several (u,v) pairs have the same similarity we choose to merge the v that has the lowest cluster number (all nodes v are from the same year). We use this procedure to avoid having to choose randomly among the pairs with the same similarity and thus produce non-deterministic output. Future work is to find better ways for ordering pairs or methods for choosing node pairs in a probabilistic manner.

For creating unit representatives we use the following strategy. Assume that a unit u has the following internal clusters $\{c_1, c_2, c_3, c_4 \ldots\}$. A term w from a cluster $c_i, i = 1, 2, \ldots$ is placed in

Table 6.2: Description of evolution for each term, derived dictionary.com, Wikipedia and Oxford English Dictionary. * can be found in Roslin Bennett (1895). WWI occurred during 1914-1918, WWII occurred during 1939–1945.

Term	Year	Description
tape	1960-1965	Common household use
aeroplane	1908	First modern aircraft design
	WWI	First test as weapon
	WWII	Large scale war weapon
rock	1950-1960	Birth of rock-and-roll music
gay	1985-1990	Recommended for use instead of *homosexual*
tank	1916	First tank in battle
cool	1964	Slang used for self-control
flight	WWI-WWII	First commercial flights non-war related
	after WWI	Commercial aviation grows rapidly
mouse	1965	The computer mouse was introduced
	1980-1985	Common usage with computers like Macintosh 128K
telephone	1839	First commercial use in Great Western Railway
	1893	28,000 Subscribers in Sweden with highest density in the world.*
	1914	USA has twice the phone density than any other country.

the unit representative u_r if one of the following conditions hold: (I) $lcurv \geq 0.7$; (II) $part \geq 60\%$; or (III) $lcurv \geq 0.4$ and $part \geq 50\%$. For all clusters c_i where $i = 3, 4, \ldots$, we add the requirement that w has to have a non-negative $lcurv$ to condition II. The selection of terms for the unit representative is highly restrictive using this strategy but has empirically proven to provide better and longer spanning units specially when combined with a low minimum unit similarity. To filter out noise, we remove all single units.

Because we are providing a proof-of-concept we have empirically discovered and set the thresholds and merging strategies. To find optimal strategies and thresholds that will maximize the performance of word sense evolution detection we need a proper evaluation dataset and automatic evaluation methods and is left as future work. A discussion on the effects of different thresholds and merging strategies is provided in Section 6.5.

6.4.3 Word Sense Evolution Tracking

In this section we will present the results of the automatically discovered word sense evolution for the terms in Table 6.1. We present an extract of the units and paths in this chapter as well as in Appendix A. To reduce the complexity of the evaluation, we consider the general agreement between the evolution presented in Table 6.2 and the discovered instances of evolution and do not differ between polysemy, merging or splitting senses.

Evolution for the term *Tape*

We begin by analyzing the evolution of the term **tape**. One of the most common usages of tape is as a (music) storing device, more specifically, the the central component of a cassette. The tape became a common household product in the 1960's. Before then, a common use of the tape was as *sowing tape*, for example as bias tape.[2] In our merged term concept graph, shown in Figure 6.2 we find both senses represented. The *sowing tape* sense is present with

2 `http://en.wikipedia.org/wiki/Bias_tape`, Retrieved 2013-04-08

one unit and one path in the upper part of the figure. The unit $u_{(1785,1854)}$ spans 1785-1854 and shows some variation within the unit. Over the entire time span the terms *thread*, *tape* and *silk* are highly important. During the first five years also the term *lace* was important but is rare in the later years. The terms in the unit clearly indicate the *sowing* sense. The path consists of two units $u_{(1787,1798)}$ and $u_{(1818,1915)}$. The overall sense of the path is the same as the unit $u_{(1785,1854)}$ but senses in the paths are expressed with different terms. In $u_{(1787,1798)}$, the terms *pin*, *needle* and *tape* are important. In $u_{(1818,1915)}$ also the term *linnen* is important. Between the second and the last cluster in $u_{(1818,1915)}$ there is a 95 year gap. However, the last cluster with the terms *linnen*, *tape* and *lint* clearly belong to the same sense and show that the sowing sense of tape is still active at least until 1915.

The path and $u_{(1785,1854)}$ both correspond to the *sowing* sense but are falsely considered separate by our method because of low similarity. Term similarity measures that considered the terms *cotton*, and *flannel* similar to *silk* and *satin* because they all represent fabric would have helped to connected the unit to the path.

The music storage sense is present starting in 1971. The first unit $u_{(1971,1972)}$ places *tape* in a car equipment context, more specifically a car stereo where the *tape* is almost synonymous with the *cassette*. The unit evolved into $u_{(1976,1994)}$ that is related in general to music storing devices like the *cassette* and the *record*. The second internal cluster introduces the term *video* and the third cluster broadens the meaning of the sense by introducing more terms related to *video* and *television*. The last cluster from 1994 is incorrectly placed in the unit. The only overlapping term is the term *tape* but because of partial matches, the terms *cord* and *record* are considered as well.

The unit $u_{(1976,1994)}$ is directly related to several other units that relate to music storage. Unit $u_{(1979,2009)}$ contains the terms *disc* and *cd* and eventually evolve into a unit that contains also the term *dvd*. Unit $u_{(1987,1990)}$ also has the term *lp* which in addition to *record*, *cd* and *radio* places the unit in a *music* context. Unit $u_{(1996,1999)}$ introduces new terms like *compact disc* and *cdrom*, both of which are synonymous to *cd*. The unit shows a widened use of storage devices with terms like *television shows* and *software*.

The unit $u_{(1980,1981)}$ does not make good sense as a unit. There are two internal clusters which have one part relating to *rental*, *leasing*, *camera* and *video*. Both clusters also have a separate part that are completely unrelated like *old york flagstone* and *contract hire*. The unit is related to $u_{(1976,1994)}$ because the internal cluster from 1981 has the terms *tape*, *record* and *video* that are full or partial matches to *tape*, *video* and *tape recorder* in the internal cluster from 1985 in $u_{(1976,1994)}$.

The concept of tape as a storing device has a second sub-path shown in the bottom part of Figure 6.2 which corresponds to a mix audio device separate from music and *scotch tape*.[3] The unit $u_{(1974,1976)}$ is related to *recordings*, *correspondence* and *documents*. The second unit in the sub-path $u_{(1988,1995)}$ is hard to evaluate because of the few terms. Unit $u_{(1990,1993)}$ is however wrongly placed in the path because of the small internal clusters; the cluster from 1995 is considered an almost match to the cluster from 1993 because two out of three terms overlap. The unit $u_{(1988,1995)}$ is incorrectly related to $u_{(1996,1999)}$ because of the reasons like the above. With this incorrect relation, the unit $u_{(1974,1976)}$ is related to the path.

The *sowing* sense of term is in existence over the entire dataset; however, our paths can only capture the sense until 1915. Afterwards there are no clusters that correspond to the *sowing* sense among our clusters. This can be an effect of underrepresentation of the sense in the collection or a reduced usage of the sense in general. The *music storing* appears slightly later than we expected.

3 `http://en.wikipedia.org/wiki/Scotch_Tape`, Retrieved 2013-06-18

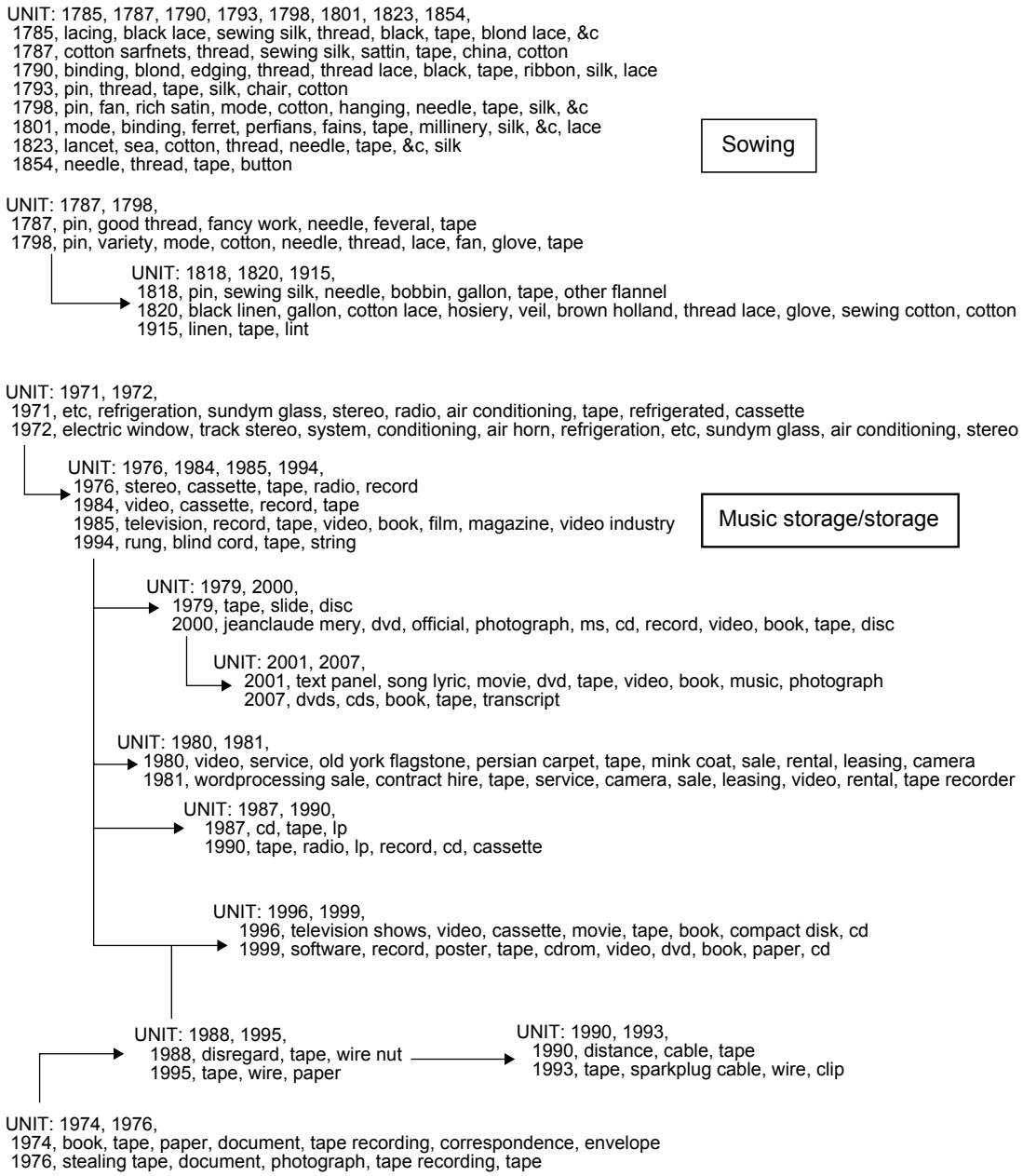


Figure 6.2: Word sense evolution tracking for the term *tape*. Some units and terms as well as all single units are omitted due to space requirements

Further Findings

The terms discussed below are presented in tables in Appendix A.

We continue with the term **aeroplane**. The first unit for the term spans from 1908 to 1916 and corresponds to a (primitive) flying machine. The unit is highly linked with ships, most likely because of overlapping terminology, e.g., air*port*. This initial sense is evolved into two distinct senses. The first corresponds to a weapon of war and is active during WWI and WWII with two polysemous units; one generally related to weapons (*infantry*, *piping*, *gun*, *artillery* and *aircraft*) and one with a more specific sense of ammunition (*bomb*, *grenade*, *rifle* and *ammunition*). The second distinct sense corresponds to a means of transportation

and starts in 1914. Terms like *motorcycle*, *car*, *lorry* and *streamlined train* discriminates the transportation sense from its weapon/war sense. All senses for *aeroplane* belong to one path and hence to one concept. The timeline fits well to the expected evolution.

For the term **rock** we have several distinct paths which can be classified as two distinct concepts, *stone* and *music*. The first starts in 1914 and has *rock* in a gardening sense with *flower garden*, *tennis lawn*, *tree* and *rock garden* as discriminating terms. There is a separate path for *rock* as a material which optimally should belong to the same concept as the above. However, because of the unit terms *stone*, *clay*, *gravel* and *sand*, there is little similarity between the different units. Therefore, they are placed in different paths. Starting in late 1980's, we find rock in its *music* sense with different types of music. The music styles are mixed with *classical*, *soul*, *rockabilly*, *folk* and *jazz* in the same unit. A *rock-and-roll lifestyle* sense appears in 1996 and is merged with the music units. *Rock*, *drugs*, *sex* and *alcohol* discriminates the lifestyle sense from the *music* sense but places both in the same concept. This grouping shows that the lifestyle is clearly derived from the *music* sense and not the *stone* sense of *rock*. Though the relation between the units in the *music* sense generally correspond to the expected evolution, they appear much later than expected. A last unit related to *rock* considers the game *Rock-Paper-Scissors* and is valid from 2000-2005 and unrelated to the other units.[4]

For the term **travel** we find many units and long paths. The first path for *travel* begins in year 1801 with two polysemous units that specify its sense as a *literature genre* with terms like *science*, *art*, *voyage*, *memoir*, *poetry* and *history*. These senses evolve into $u_{(1821,1973)}$ (not shown in the figure due to its size with 27 internal clusters) that in 1971 has an internal cluster with terms like *economics*, *design* and *politics*. The unit shows that the *literature genre* sense remains also in late 20th century with a somewhat updated meaning. The first signs of *travel* as a concrete activity appears in the early 1900's with terms like *first class hotel*, *ticket* and *sightseeing*. This sub-path can be followed until 1989. Other parts of the paths can be followed until 2007 and, in more recent times, provide more diverse clusters and more casual terminology, like *movie*, *sport*, and *wine* in addition to *travel*. The first clusters on travel as a concrete activity do not reveal the means of travel with the exception of *carriage drive* in 1904 ($u_{(1904,1929)}$). In this context the term can refer to an activity during the stay rather than a means of transportation. In 1985 ($u_{(1923,1985)}$), we find the term *airline* for the first time and in 2001 ($u_{(2001,2005)}$) we find a unit specifying different means of transportation with terms like *plane*, *automobile*, *bus* and *car*.

The evolution of *travel* seems generally reasonable but is hard to judge against history as it is subject to social, economical and regional influences. The overall meaning of the term is stable over the entire collection while the abstraction level changes over time and becomes more concrete. Because the term does not correspond to a technological invention or has a clear change in meaning we cannot formally prove the correctness of the detected evolution.

A description for the remaining terms and the word sense evolution found for these terms is provided in Appendix A. Like with the terms presented here, the main evolution of the remaining terms can be found and in a timely manner. Overall, only the terms *rock* and *flight* have time delays that are longer than 2–10 years.

6.5 Discussion

Our experiments show that we are able to find much of the existing word sense evolution for a term without depending on human input or existing resources. Instead we depend on automatically extracted word senses which are automatically grouped into *units* to capture individual senses over time. Units are then grouped into *paths* that capture concepts for a term.

4 `http://en.wikipedia.org/wiki/Rock-paper-scissors`, Retrieved 2013-04-08

We use a localized view to create clusters around a term and merge these clusters with the global clusters for the same term (found in Chapter 5). The resulting cluster set is larger and represents more fine-grained senses. Still, most dictionary entries for a term are not represented in our clusters. This is most likely a consequence of the underlying data compared to a dictionary. Newspapers do not cover all senses of a term and dictionaries do not reflect the popularity of senses. To verify the coverage of dictionary senses in our clusters it is necessary to first verify the coverage of dictionary senses in the dataset. For this, each instance of a term in the dataset must be disambiguated and assigned one dictionary sense. Disambiguation lies outside of the scope of this thesis and we leave this as future work.

Nevertheless, there is no guarantee that all main dictionary senses are reflected in the cluster set, even when disregarding fine-grained and obsolete dictionary senses. To improve the coverage of senses there are several alternatives; (1) using different or additional data; (2) other clustering thresholds or hierarchical clustering; (3) different word sense discrimination algorithm to derive clusters; or (4) other types of extracted information like topics or context vectors.

We find that, by using other similarity thresholds and strategies than those presented in Section 6.4.2 for choosing unit representatives, clusters are grouped differently into units. These differences correspond to different levels of granularity and capture more or less evolution within one unit. Smaller units with shorter time spans have the potential to capture fine-grained senses. However, fine-grained units require better matching in order to group units into concepts, because it is less likely that the units have large term overlaps. As an example, fine-grained units for the term *rock* can capture different styles of music into different units. However, if we use the same relating strategies as for the coarse grained units, we cannot place the *rock-and-roll lifestyle* units in the same path as the *music style* units.

To convey word sense evolution to users, it is important to keep the notion of concepts. For the general public, we believe that coarser granularity with better grouped senses is more important than capturing fine-grained senses. For example, to have the concrete sense of *travel* grouped with the *literature* sense of the term allows the user to understand that the literature reflects the same underlying concept, namely *moving from location A to location B*. It helps users understand that text written about *travel* in early 19th century reflect the same concept as modern travel. Using the same rationale, users can understand that books written about *flight* in early 19th century do not reflect *flight in an aircraft* because the latter is not placed in any paths with other senses of *flight* before 20th century.

To fully evaluate our results as well as find optimal thresholds and merging strategies, it is necessary to have a properly defined ground truth. The creation of such ground truth requires linguists and historians with in-depth knowledge of the specific collection. The intended application and users must be taken into consideration when constructing ground truth datasets. If users are scholars in linguistics or the general public, is of great importance. For the general public it can be valuable to know that the *rock-and-roll lifestyle* is related to *rock music* while a linguist does not necessarily agree that these senses belong to the same concept. Creating such collections is expensive and time consuming and out of scope for this thesis. We hope that increasing interest for the topic of automatic detection of word sense evolution leads to the development of standard datasets annotated with ground truth as well as standard evaluation measures.

To help users understand word sense evolution and interpret content in long-term archives it is important to present information on an appropriate level. For this reason we believe that labeling of clusters, units and paths is of utmost importance for the future usability of automatically found word sense evolution. By classifying and labeling our findings, we can better display the information to the user of an archive. Without labeling we must present units and paths much like we have in Figure 6.2, as lists of terms. Non-expert users, however, do not benefit from such a representation as it. When properly chosen, labels like those used in the TeVo browser, see Figure 6.3, can significantly help clarify the evolution, e.g., *aeroplane as*

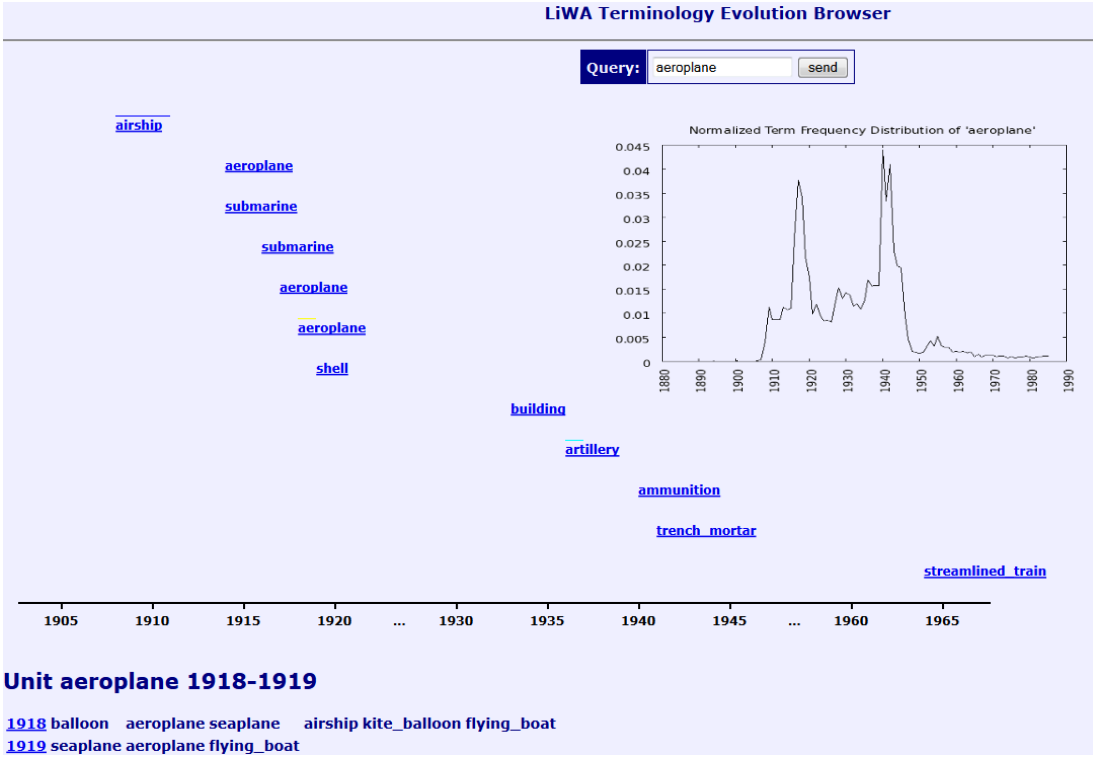


Figure 6.3: Terminology Evolution (TeVo) Browser showing global clusters and units for the term *aeroplane*.

means of travel/aeroplane as tool of war. In addition to labels, example sentences or documents can be shown to provide further context to each word sense. Also contemporary images would provide users with information to better understand a word sense. This additional context information helps users to get a first understanding of a term and its senses without requiring the user to invest significant amount of time to manually gather the same information.

In addition to aid in understanding, labeling and classification can help to detect evolution. Units that belong together, like the *sowing* units of *tape* shown in Figure 6.2, can be placed in the same path also when the units contain few overlapping terms. Algorithms to find good labels, context information and matching images are left as future work.

The clusters, units and paths that we have worked with in this chapter are independent from the original documents and reflect high level, word sense evolution of a term over time. This is not sufficient to alert users that a specific term in a historical document has changed meaning. For this, it is also necessary to perform disambiguation and assign clusters, units or paths to individual terms. Using the merged term concept graph it can then be determined if the word senses have changed over time and alert the user if necessary.

6.6 Application – The *TeVo* Browser

To present a possible scenario for utilizing word senses, we created a user interface called the *TeVo browser*, short for Terminology Evolution Browser, aimed at professionals who wish to explore long-term archives. The user interface was published by Zenz et al. (2013) and builds on word senses extracted in this thesis. A more detailed description of the system is available in the citation above and the implementation is fully credited to the first author.

The TeVo browser allows the user to view word sense clusters, units and paths over a timeline where different paths are colored using different colors. The timeline shows one selected term for each unit, assuming labeled units. Since we have no proper labeling, we choose a term, among those with the highest curvature value, as the label. Clicking on one of the terms in the timeline displays the unit with all its internal clusters. The user can go deeper by choosing a cluster and viewing the link information between the cluster terms. In addition, the user can ask for context terms for each cluster and the terms are extended with terms from the co-occurrence graph that link to the cluster terms. To help the user distinguish the cluster terms from the context terms, the former are colored. The user is provided with normalized term frequency information which is displayed as a graph on the right-hand side. This information helps the user to find the expected time span of a term and can point to interesting time periods. In Figure 6.3 we see two clear peaks around WWI and WWII for the term *aeroplane*. We also find that the term did not exist in the collection before early 1900's.

We performed a user study where users were asked to use the TeVo browser to find the meaning of a set of terms. The users were asked how well they were able to assess a term's meaning using the displayed information. The study was made using only global cluster information. With a limited number of participants we found a good to average result for the usability and utility of the TeVo browser. The main conclusion of the study was the complexity and difficulties involved in presenting word sense information and the importance of finding good labels.

6.7 Conclusions and Contributions

In this chapter we investigated methods for automatically detecting word sense evolution. We made use of automatically extracted word senses using the pipeline presented in Chapter 5. To the best of our knowledge, we are the first to automatically track individual senses over time and determine changes in the meanings of terms. We have gone beyond previous work by also providing an answer to *what* has changed and *how* it has changed in addition to *when*. Our results show that the methods we have chosen have the potential to provide us with the main evolution for a term that correspond roughly to the timeline of the actual change.

We model a term and all its senses in term concept graphs for each period in time and iteratively merge term concept graphs to create a *merged* graph that stores all evolution for the term. Individual senses, their evolution, narrowing and broadening over time are captured in units. Each unit is represented by the most important terms from the underlying clusters. Units are related and grouped into paths that represent concepts. Paths capture polysemy and homonymy.

We performed a small scale evaluation on a testset consisting of 10 terms. For most of these terms, the automatically instances of evolution corresponds well to the expected evolution. In most cases, evolutions are found at the time they occur or with a slight delay of 2-10 years, likely a consequence of using newspapers which only address issues when they are newsworthy. Over a 222 year time span, this delay can be considered acceptable but leaves room for improvement. Because of the size of the testset, we should consider these results as indicative and future work is to perform large scale testing of the method and to fine-tune thresholds and find good strategies for (1) finding important terms to represent units; and (2) merging units.

The contributions presented in this chapter are the following:

- We formally defined word sense evolution in terms of changes in term concept graphs by building on a definition of language evolution proposed by Cooper (2005).
- We proposed the use of local as well as global clusters to overcome the sparseness of clusters and low coverage of terms and were able to more than double the number of clusters.

- We proposed a merging function for term concept graphs that captures individual senses in units and groups units into concepts. We were able to capture the main evolution for a term corresponding to roughly the timeline of the actual change.
- We proposed a fully automatic method for detecting word sense evolution that only relies on a collection of text and can capture evolution present in the collection. The evolution can be used for understanding in a long-term archive and does not suffer from any discrepancies between the collection and the discovered instances of evolution.
- Our method is the first to tackle broadening and narrowing of senses, splitting and merging of senses as well as detect polysemy and homonymy.

6.7.1 Limitations and Future Work

The methods described in this thesis have potential to capture word sense evolution present in a collection of text. For each term, there are comparably few years for which there exist clusters. Because of this sparseness, we cannot make any claims about what happened between the time points where we have clusters. Instead, we must assume that nothing happened until we see new evidence. However, newspaper discuss only things that are "news worthy" and if we rely exclusively on newspaper archives, we cannot expect to have full coverage of the senses and changes for a term. Thus, the discovered instances of evolution can only be expected to reflect the collection and not actual events and changes as can be found in an encyclopedia or dictionary. The combination of multiple, diverse resources would offer better coverage of word senses and should be considered in future work.

Our methods do not allow us to make full use of our collections. Detecting word sense evolution by relying on automatically extracted word sense clusters makes the method highly dependent on the quality and coverage of the extraction method. To find better evolution, we must find more word senses, complement using other resources like books and magazines or use additional types of information like topics or context vectors.

Our methods for comparing word senses depend on string matching for comparison of terms. This limits our possibilities as we cannot make proper distinctions between units, in particular concerning polysemy. Certain terms, like *motor* (motorbike vs. bike) or *electronic* (mail vs. electronic mail) can strongly affect the meaning of a unit but are currently not considered. In contrast, terms that have relations but no string overlap are not considered similar, like *motorbike* and *vehicle*.

It is difficult to fully evaluate the methods proposed in this chapter. Because the evolution reflects the underlying collection, it is not enough to consider a dictionary for evaluation. Instead, it is necessary to cross-reference with the collection itself. For example, the slang usage of *cool* as a personality trait was found 6 years after the first usage (as stated in the Oxford English Dictionary). To consider this as an error for the method it must first be verified that the term was used in that sense at some time before but not found by the algorithm.

Future work is to improve the accuracy and coverage of automatic detection of word sense evolution by ensuring that more senses are captured in the clusters. In addition, we must find better similarity measures to compare terms, clusters and units which also take hyper/hyponym relations and labels into account to allow finding similarity between terms also when they have no lexical or phonetic overlap.

In addition to a full evaluation and more refined methodology for detecting evolution, it remains future work to find the best way to preserve and utilize discovered instances of evolution. Temporal indexing structures, information retrieval and presentation techniques as well as scalability issues are future directions for research in the field of automatically detecting word sense evolution.

Chapter 7

Finding Named Entity Evolution

High impact events, political changes and new technologies are reflected in our language and lead to constant evolution of terms, expressions and names. Not knowing about names used in the past for referring to a named entity can severely limit the ability to find content in long-term archival search. In this chapter we propose *NEER*, an unsupervised method for *Named Entity Evolution Recognition* independent of external knowledge sources. Most work in this chapter has been published in (Tahmasebi et al., 2012a).

This research field is becoming increasingly important as digital content covers longer and longer timespans. Most previous works depend on the availability of external knowledge sources or assume a static context and expect the names to be the only changing factor. These approaches are limited to historical data as reliable knowledge sources cannot be assumed and, over time, as seen in Chapter 5 and 6, also terms in the contexts change and therefore static contexts cannot be assumed. We follow a statistical approach to eliminate the dependency on external resources and use a context based method that considers only periods with a high likelihood of name change, thereby capturing evolving names with less computational effort. This independence opens the possibility to apply the method to any corpus, including historical collections or those in different languages, and to identify undocumented named entity evolution, like those that are not covered by most modern resources.

The rest of the chapter is organized as follows: We describe our approach and highlight the limitations of previous work in Chapter 7.1 and present our methodology in Chapter 7.2. We introduce our data and testsets in Chapter 7.3 and present our experimental results. We discuss our findings in Chapter 7.4 and finally present our conclusions. The terminology and definitions needed in this chapter are described in Chapter 3.1.

7.1 Contributions and Relation to Existing Methods

Previous work in the area of term to term evolution can be generalized as shown in Figure 7.1. A word w_i is mapped to its context and compared to word w_j by comparing contexts. If the corresponding contexts are similar it is concluded that w_i and w_j are temporal co-references, i.e., are evolutions of each other. These methods have severe drawbacks because they assume that the queried entity is the only evolving factor and that contexts stay stable over time. This is however not the case. Comparing the term *walkman* and *ipod* (an example found in (Berberich et al., 2009)) directly by means of contexts from the New York Times corpus ($C_{walkman}$, C_{ipod}) we find that even though some terms occur in both contexts, the majority

Table 7.1: Five terms and their contexts in the New York Times corpus.

$\mathbf{C_{walkman}}$	$\mathbf{C_{discman}}$	$\mathbf{C_{minidisc}}$	$\mathbf{C_{mp3\ player}}$	$\mathbf{C_{ipod}}$
cassette	walkman	compact	music	apple
audio	stillvideo	disc	digital	mp3
video	sony	sony	internet	roqit
tape	portable	digital	audio	player
music	cd	cassette	player	music
sony	kodac	phillips	files	geeks
digital	video	walkman	cd	jukebox
stereo	priestly	dcc	computer	portable
earphones	digital	prerecorded	mp3	macintosh
recorders	camera	video	portable	dlink

of terms have changed.[1] In Table 7.1 we see the ten most frequent terms of the contexts of terms *walkman, discman, minidisc, mp3 player* and *ipod* from all articles from New York Times corpus during the years the term was introduced.[2]

We find that the only overlap between $C_{walkman}$ and C_{ipod} is the term *music*. By comparing the intermediate contexts pairwise instead, we find that there is a much larger overlap between the contexts. For instance, $C_{walkman}$ has a 40% overlap with $C_{discman}$ which in turn has a 30% overlap with $C_{minidisc}$. The same properties hold when we compare the 20 most frequent terms and we find that the overlap between $C_{mp3player}$ and C_{ipod} increases further. From this we deduce that comparing contexts pairwise where the contexts are closer in time is more effective than comparing two contexts far apart in time.

The same observation holds for Kaluarachchi et al. (2010). They consider nouns to have evolved into each other if they point to the same event (verbs) at different points in time. Over long periods of time also the verb undergoes evolution and hence the method is limited to those terms where the corresponding event has not changed over time. There is, however, reason to believe that verbs are more likely to change over time than nouns (Sagi, 2010) and hence the problem of finding evolved nouns is mapped to a more difficult problem, namely to find evolved verbs.

In our work, we make use of the typical characteristics of named entity evolution. Unlike with other types of evolution, such as word sense evolution, named entity changes typically occur during a short time span. There are few concept shifts where the term slowly changes, instead name changes occur due to special events like being elected pope, getting married or merging/splitting a company. If the named entity is of general interest, these name changes will also be announced to the public repeatedly during the change period with sentences like "*The day after Cardinal Joseph Ratzinger became Pope Benedict XVI ...*".[3]

By first identifying candidate change periods and then creating a context around a term, we believe that we can capture both the old and the new co-reference in the same context. We thus eliminate the risk of comparing contexts that are vastly different. Figure 7.2 illustrates our method. By identifying change periods t_1, t_2, t_3 we can create contexts around a term which contain both co-references and do not have to compare largely different contexts like those of *walkman* and *ipod*, in fact, we do not compare their contexts at all.

Using change periods we reduce the complexity of the algorithm as we first identify change periods and limit the search for co-references only to those periods. If a time period between

1 We do not consider *walkman* and *ipod* to be co-references as they do not correspond to the same named entity. We use this example to illustrate the difficulties that arise and the different methods.
2 For *walkman* we chose year 1987 as its true year of introduction falls outside of the corpus' time span.
3 The New York Times, April 21, 2005.

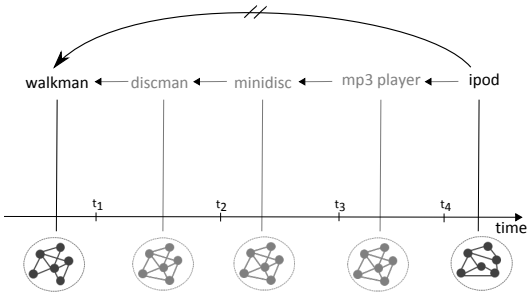


Figure 7.1: Detection of named entity evolution by comparing contexts.

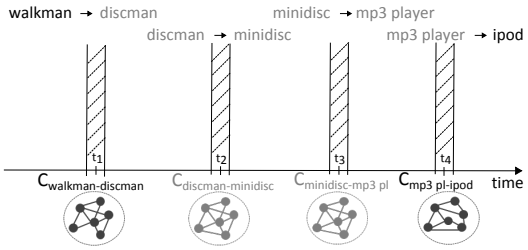


Figure 7.2: Detection of named entity evolution by creating a context in the change periods.

two change periods is required by the user, the co-reference found in the last change period is valid and can be returned to the user. The method presented by Berberich et al. (2009) requires a re-computation each time to deduce the temporal co-reference (query reformulation in the terms of the paper) at each time point required by the user.

7.2 Methodology

To find temporal co-references we use the pipeline depicted in Figure 7.3. We start by detecting change periods for a query term over the entire collection. We make use of the identified change periods to find the subsets in which we look for evolution. We extract single and multi-word nouns and find named entities mentioned in the text. We create contexts around extracted terms by applying co-occurrence analysis and use the context and the extracted terms to find direct co-references. Finally we apply frequency analysis as well as machine learning to identify direct and indirect co-references and filter out noise.

The pipeline used for detecting word senses in Chapter 5 is not applicable for capturing different named entities in clusters because names are unlikely to appear in comma separated lists. As a consequence, the pipeline for detecting evolution presented in Chapter 6 cannot be used for detecting named entity evolution. Therefore, in this chapter, we follow the context centered approach presented in Berberich et al. (2009).

7.2.1 Identifying Change Periods

Named entity changes are typically associated with significant events concerning the entities which lead to increased attention. We use this property to pinpoint change periods and detect those using a **burst detection algorithm**. We use the Kleinberg algorithm (Kleinberg, 2002) to find bursts from the entire document collection $\mathbf{D}$. The algorithm models the frequency of documents $\mathbf{D}_w$ (all documents containing term w) using a series of probability distributions. Each distribution represents an increasing degree of *burstiness*. A set of states indicates which distribution is active. By assigning a cost to state transitions, the algorithm ensures that an optimal state sequence creates bursts that end only if they are followed by a sufficiently large period of lower activity. This avoids splitting bursts for example around weekends when the number of articles drops.

We detect bursts related to an entity by retrieving all documents in the corpus containing the query term, grouping them into monthly bins and running the burst detection on the relative frequency of the documents in each bin. Each resulting burst corresponds to a significant event involving the entity. However, these bursts do not necessarily correspond to a name change.

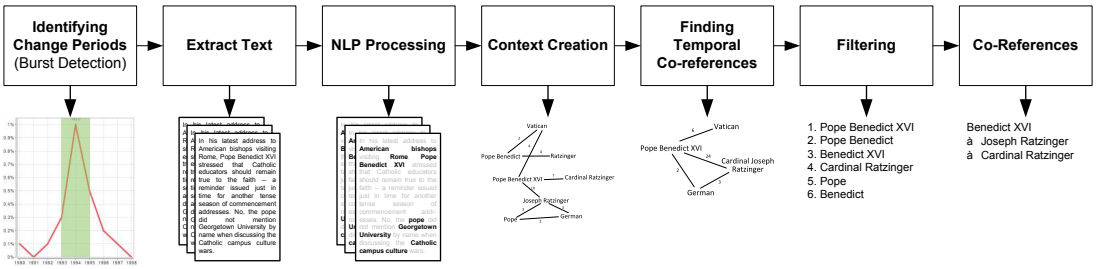


Figure 7.3: Pipeline used to detect temporal co-references.

By choosing the `topB` strongest bursts we expect to find a subset of bursts which also capture change periods. We denote each change period p_i for $i = 1, \ldots, \texttt{topB}$.

7.2.2 Creating Contexts

After identifying change periods p_i for an entity w, we create a context for each period by extracting all documents $\mathbf{D}_w$ that mention the entity or any part of it and are published in the year corresponding to p_i. We clean the text and extract nouns, noun phrases and named entities. We use noun phrases to capture more information and create richer contexts around entities. All extracted terms are added to a **dictionary** and used for creating a co-occurrence graph, much like the pipeline used to extract word senses in Chapter 5. The co-occurrence graph is an undirected weighted graph which links two dictionary terms if and only if they are present in $\mathbf{D}_w$ within k terms of each other. The weight of each link is the frequency with which the two terms co-occur in $\mathbf{D}_w$. The context of entity w is considered as all terms co-occurring with w. The context of a co-reference class is considered to be all terms co-occurring with any of the terms in the co-reference class.

7.2.3 Finding Temporal Co-references

To find *direct co-reference classes* we need to consolidate the extracted terms by recognizing all variants of each term. As an initial step each term from the dictionary with a frequency above `minFr` is placed in its own co-reference class where the term acts as the representative as well as the only co-reference, for example, $\text{coref}_{\text{Benedict}}\{\text{Benedict}\}$.

Merging: The procedure for merging terms and co-reference classes is shared between all three rules described below; each co-reference class is represented by the term with the highest frequency. A frequency is stored in the co-reference class for the representative r as the sum frequencies of all terms in the class. If two co-reference classes have the same representative, they are merged into one. Each co-reference class carries with it all co-occurrences that belong to any of the terms in the co-reference class. These are considered as the context C_r. When terms are merged, the context is updated accordingly. When two co-reference classes are merged, the representative with the highest frequency is chosen as the representative of the merged co-reference class. A special case occurs when two co-reference classes are merged where one has a single term representative. In this case, the longer term will be chosen as a representative for the merged co-reference class, see examples in prefix/suffix rule.

Next we will describe the main rules used for finding all direct co-reference classes. In the initial iteration the first rule works on the dictionary terms and populates an index with co-reference representatives. In the second and all subsequent iterations, the first rule makes use of the terms in the index. This index is passed through all the rules. The rules are iterated until there are no more terms in the index that can be merged. Only co-reference class representatives are stored in the index.

1. **Prefix/suffix rule:** This rule creates co-reference classes by merging dictionary terms that differ only by a prefix or suffix. For example, the co-reference classes of *Pope Benedict* and *Benedict* as well as *Pope* and *Pope Benedict* are merged. In both cases the resulting co-reference class has *Pope Benedict* as representative and these co-reference classes are therefore merged: $\text{coref}_{\text{Pope Benedict}}\{\textit{Pope, Pope Benedict, Benedict}\}$.

2. **Sub-term rule:** This rule merges classes which are represented by terms that can be considered sub-terms. For a term to be a sub-term of another we require the longer term to contain all terms from the shorter term in the correct order. For example, the co-reference class represented by *Cardinal Joseph Ratzinger* and *Cardinal Ratzinger* are merged.

3. **Prolong rule:** The third rule is used to create longer terms than might be found in the dictionary. It merges two representatives from the index into one longer term if the terms have an overlapping part and there exists a co-occurrence between the remaining terms. E.g., *Pope John Paul* and *John Paul II* are merged if there is a co-occurrence (*Pope John Paul , II*) or (*Pope , John Paul II*); the representative of the merged co-reference class becomes *Pope John Paul II*. The third rule also merges terms that differ due to plural of the prefixes assuming that the prefix is not considered a stopword. E.g., *Senator Barack Obama* and *Senators Barack Obama* are merged but *Mr Obama* and *Mrs Obama* are not.

Final merging When the terms in the index cannot be merged further, a final round of merging takes place. In this round we apply a *soft* sub-term rule where we drop the requirement that the terms should be in the same order but require them to be similar in frequency. This way terms like *Illinois Democrat* and *Democrat of Illinois* are merged.

Consolidation When all terms are merged we create a mapping from each term to the co-reference class representative that has the highest frequency. Using this map we consolidate all terms in the context of each class.

An example of is shown in Figure 7.4 (original context in Figure 7.4a). Using the three rules we find the following co-reference classes:

$$\begin{aligned}\text{coref}_{\textit{Cardinal Joseph Ratzinger}}&\{\textit{Cardinal Joseph Ratzinger, Joseph Ratzinger, Ratzinger}\}\\ \text{coref}_{\textit{Pope Benedict XVI}}&\{\textit{Pope Benedict XVI, Pope Benedict,Pope}\}\\ \text{coref}_{\textit{Vatican}}&\{\textit{Vatican}\}\\ \text{coref}_{\textit{German}}&\{\textit{German}\}.\end{aligned}$$

Next a mapping is created:

$$\begin{aligned}\textit{Joseph Ratzinger} &\rightarrow \textit{Cardinal Joseph Ratzinger}\\ \textit{Ratzinger} &\rightarrow \textit{Cardinal Joseph Ratzinger}\\ \textit{Pope Benedict} &\rightarrow \textit{Pope Benedict XVI}\\ \textit{Pope} &\rightarrow \textit{Pope Benedict XVI}\end{aligned}$$

Additionally, all co-reference class representatives map to themselves. Then each term in the co-reference class context is consolidated and replaced using the map. If two co-occurrences share a term they are merged into one and the frequency of the co-occurrence is updated as shown in Figure 7.4b.

Ranking The term frequencies and the merging steps offer a natural ranking of co-references. When two terms are merged, like *Pope Benedict* and *Pope Benedict XVI*, we update the

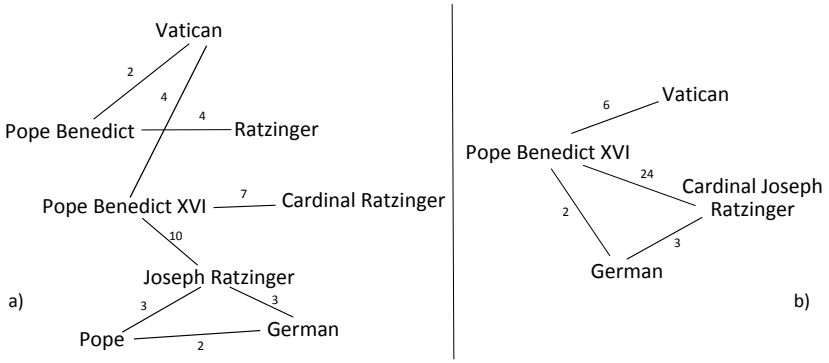


Figure 7.4: a) Example graph after creating context. b) Resulting graph after consolidating and merging of all direct co-references.

frequency of the class representative by summing up the frequencies. During merging all co-occurrences are updated with the sum of the frequencies of all participating terms. In Figure 7.4b we see that the co-occurrence frequencies of (*Vatican* , *Pope Benedict XVI*) is 6 because the frequency of (*Vatican* , *Pope Benedict*) is 2 and (*Vatican* , *Pope Benedict XVI*) is 4. The term frequencies and co-occurrence frequencies are stored in each co-reference class. The frequency of *Pope Benedict* and *Pope Benedict XVI* is much higher than that of *Benedict XVI* and *Eggs Benedict* and during consolidation the term *Benedict* is replaced with *Pope Benedict XVI* rather than *Eggs Benedict*.

Indirect Co-references Indirect co-references are found implicitly by means of the direct co-references. After consolidation, all terms in the context of a co-reference class are considered candidate indirect co-references. These are a mix between true indirect co-references, highly related co-occurrence phrases as well as noise. The quality of the indirect co-references is dependent on the named entity extraction, co-occurrence graph creation and filtering of the co-occurrence graph. The choice of including single token terms in addition to multi-token terms has a high influence on the quality of the resulting co-occurrences. In Figure 7.4b *Vatican*, *German* and *Cardinal Joseph Ratzinger* are candidate co-references for *Pope Benedict XVI*.

If our method does not find any co-references for a term, all direct co-occurrences from the co-occurrence graphs (derived from the union of change periods) are returned instead.

7.2.4 Filtering Temporal Co-references

To remove noise and identify true direct and indirect co-references, we make use of term frequencies as well as document frequencies for the filtering. We start by describing similarity measures between terms and continue with filtering techniques. All similarity measures and filtering consider found term–co-reference pairs (w,c), e.g., (*Pope Benedict XVI*, *Vatican*) or (*Pope Benedict XVI*, *Cardinal Joseph Ratzinger*), with the aim to remove the incorrect pairs and keep the correct ones.

Similarity measures To keep true co-references we need to measure the temporal relatedness of terms. Unlike previous works that take temporal features into account it is not sufficient to consider relatedness over the entire time span of a collection. Radinsky et al. (2011) use time series of terms to capture the relatedness of terms like *war* and *peace* or *stock* and *oil*. These terms are considered related because they have similar frequencies over time.

For temporal co-references, capturing the overall relatedness is not sufficient. Both direct and indirect co-references can be related only for a certain period in time and then lose their relation. To give an example: Both *Hillary Rodham* as well as *Hillary Clinton* have been used to refer to the same person at different periods in time. The latter was used after Hillary Rodham's marriage to Bill Clinton. Measuring the relatedness between *Hillary Rodham* and *Hillary Clinton* using global term frequencies (i.e. term frequency over the entire corpus) will not yield the correct results. However, global measures can help to find certain types of co-references, for example, *Barack Obama* and *Barack Hussein Obama*, that hold over the entire corpus.

Therefore, to fully capture temporal co-references we need, in addition to global relatedness measures, a relatedness measure that captures how related terms are during the time periods in which they can be related at all. To this end we allow a relatedness measure to consider periods where both terms occur. In all cases we use the normalized frequencies.

We consider four relatedness measures: (1) Pearson's correlation (*corr*) (Weisstein, 2012a); (2) Covariance (*cov*) (Weisstein, 2012b); (3) Rank correlation (*rc*); and (4) Normalized rank correlation (*nrc*).

The two first measures are standard relatedness measures where *corr* measures linear dependence between random variables while *cov* measures correlation between two random variables. The two last measures are rank correlation measures and inspired by the Kendall's tau coefficient that considers the number of pairwise disagreements between two lists. Our rank correlation coefficient counts an agreement between the frequencies of two terms for each time period where both terms experience an increase or decrease in frequency without taking into consideration the absolute values. The rank correlation is normalized by the total number of time periods. The normalized rank correlation considers the same agreements but is normalized with the total number of time periods where both terms have a non-zero term frequency.

Filtering Co-references using Pearson's Correlation The first filtering makes use of the *corr* measure to determine which co-references are related to the query term and filter out the rest. This measure is used by Radinsky et al. (2011) to measure similarity between terms and serves as a comparison for our filtering mechanisms. We keep a co-reference if its correlation to the query term exceeds the threshold corr_{min}. An increase in the filtering threshold would lead to the same or decreased recall while the precision could be affected either way. A decrease in the threshold would lead to a lower precision. Therefore a low threshold is sufficient to get an upper bound of the recall while maintaining precision.

Filtering Co-references using Document Frequency The second filtering is based on the document frequencies (df) of co-references. We filter out all co-references that differ largely in document frequency from the document frequency of the query term. The filtering depends on the document frequency of the most frequent term in the dictionary corresponding to a change period ($\text{df}_{\texttt{max}}$), the document frequency of the query term ($\text{df}_{\texttt{query}}$) and a *scaling factor* (sc). We filter out all co-references that have a document frequency $\text{df} \geq \text{df}_{\texttt{query}} \cdot \text{sc}(\text{df}_{\texttt{max}})$, i.e., which are frequently used in different contexts.

Filtering Co-references using Machine Learning Our third and final filtering method is based on machine learning. We use a random forest classifier (Breiman, 2001) consisting of a combination of decision trees where features are randomly selected to build each decision tree. In total, ten trees with five features each are constructed. We choose features from the similarity measures presented above. That means, for each found term–co-reference–change period tuple (w , c , p_i), we calculate the *corr*, *cov*, *rc* and *nrc* measures. We also use the average of all four measures as a fifth feature. We calculate these five measures globally and as well as locally around the change periods. For the change periods we choose one period

of two years before and one period of two years after each change period which results in 15 features in total for each tuple. Finally we classify the pair as either 1, for c being a correct co-reference of w, or 0 otherwise.

7.3 Experiments

The aim of our experiments was to measure how well our method, called NEER, can detect names used during different time periods to refer to the same entity. We did this by (1) investigating how well burst detection can be used to capture change periods; and (2) measuring precision and recall of the co-references found using NEER with and without filtering. Each experiment in (2) was performed using two settings: (a) The first made use of the known change periods (denoted **known periods**); and (b) the second used the detected bursts (denoted **found periods**). We used the known change periods to measure how well the method works assuming that we can find the correct change periods.

As there are no available baselines to compare our methods to, we defined our own baseline and named it **co-occurrence**. This considers all terms that co-occur with the queried named entity within a sliding window for all change periods, for (a) and (b) separately. This provided a baseline that shows what can be achieved with minimal computational effort.

We considered precision and recall as defined below for our evaluation. For a term we required all direct co-references and at least one indirect co-reference for each name change to achieve full recall. That means that for *Joseph Ratzinger* we required all direct forms {*Cardinal Joseph Ratzinger, Cardinal Ratzinger*} but only one of the indirect {*Pope Benedict XVI, Pope Benedict*} to achieve a recall of 100%.

$$precision = \frac{\#\text{ correctly captured co-references}}{\#\text{ all captured co-references}} \tag{7.1}$$

$$recall = \frac{\#\text{ captured co-references}}{\#\text{ known name changes}} \tag{7.2}$$

7.3.1 Dataset and Testset

For our experiments we used the New York Times Annotated Corpus (NYTimes). The dataset contains around 1.8 million articles published between 1987 and 2007.

We devised a testset of named entities, based on result from Kanhabua and Nørvåg (2010), with direct as well as indirect co-references and divided them into three categories: *People*, *Locations* and *Companies*. We identified all relevant name changes and the year in which they occur. Each co-reference pair was verified using three judges and kept if at least two judges agreed. If the change occurs in January one year also the previous year was added. We mirrored all the entities so that *Pope Benedict* → *Cardinal Ratzinger* and *Cardinal Ratzinger* → *Pope Benedict* both exist as separate entries. Change periods are available in the released testset.

The final testset was devised by keeping all terms that (1) exist in the NYTimes; (2) have a change period in the NYTimes time span; and (3) occur at least 5 times in at least one change period. The dataset is available in Tahmasebi et al. (2012b). We started with 75 distinct names and 294 co-references. After filtering there were 16 distinct entities corresponding to 33 names and 86 co-references (44 indirect and 42 direct).

7.3.2 Experimental Setup

We used the NYTimes API to extract documents from the NYTimes corpus. To extract terms we used Lingua English Tagger (Coburn, 2008) for finding single and multi-token nouns and

Table 7.2: Precision and recall for the baseline and different filtering techniques.

Method	**Found periods**			**Known periods**		
	Precision (%)	Recall(%)	# co-ref	Precision (%)	Recall (%)	#co-ref
Co-occurrence	8	51	120	20	59	16
NEER	8	90	128	13	89	64
NEER + Corr	20	61	107	17	74	43
NEER + DF	33	86	28	50	81	10
NEER + ML	93	83	13	90	65	4

the Stanford Named Entity Recognizer (NER, Finkel et al., 2005) to extract named entities. NERs typically consider names but not the role as a part of the name. For example *Barack Obama* is extracted but not *Senator Barack Obama*. Therefore we used the Lingua tagger which recognizes also terms like *Senator Barack Obama*. Named entities recognized by both methods are counted twice and thus receive a higher frequency. This procedure helps to choose good representatives for the co-reference classes.

In order to increase precision we filtered out infrequent terms. During graph creation we required a term to occur at least three times in the collection used for creating the graphs. If the most common terms in the dictionary occurred more than 800 times, we required at least five occurrences. For finding direct co-references we required that each term occur at least five times. However, if the most common term in the dictionary occurred more than 3000 times we increased the threshold to 10 occurrences. We also filtered out terms containing lowercase tokens. For this reason the term *Union of Myanmar* could not be found by the system.

For the relatedness calculations we used normalized term frequencies that are calculated as the fraction of term occurrences in all documents published per month divided by the total number of tokens in these documents.

To find the bursts we used the Java implementation from CIShell (Alencar, 2012) with 3 burst states, a transition probability γ of 0.8 and a density of 1.9. Using these parameters we detected on average 3.2 bursts for each term in our testset.

7.3.3 Results

Burst Detection We approximated change periods using burst detection. Considering all found bursts for an entity, we were able to capture 73% of all change periods. This indicates that burst detection works well for capturing change periods but that there is room for improvement and parameter tuning. To reduce complexity and false positives, we limited the number of bursts to `topB` = 6. Using the `topB` bursts we were able to capture 66% of all change periods.

If a name is ambiguous, bursts are less suitable for capturing correct change periods as the burst detection algorithm cannot distinguish between entities. This is the case for *George Bush* where the top bursts are 1988, 1989 and 1990 and correspond to *George Bush Senior*. *George W Bush* is not ambiguous and bursts are found for 1999 - 2001. Bursts are highly biased towards the dataset; only what is covered in the dataset will appear in the bursts. On the other hand, only what is interesting for the general public, is covered.

The results for the experiments presented next are summarized in Table 7.2.

Baseline – Co-occurrence For comparison we chose a baseline consisting of all terms that co-occur with the query term in the datasets corresponding to the known and found burst

periods. This naive baseline serves as a lower bound on recall. We used precision and recall for direct and indirect co-references found by our method and the corresponding entries from the testset. In Table 7.2 we find that the recall for the baseline is 59% using known periods and 51% using found periods. When considering the co-occurrences for the query term very few or no direct co-references were found for the terms. Instead most indirect co-references were found. The precision differs largely between known and found periods; for the latter precision is surprisingly high with 20% compared to the found bursts (8%). This shows that the known periods help to find better co-references without introducing too much noise.

To mimic other methods where the user chooses a target time, we randomly chose three years per query term (corresponding to the average number of bursts per term) and created co-occurrence graphs for these years. We repeated the experiment three times and got an average recall of 36% which is statically significantly lower than the recall for known periods, showing the power of using change periods for finding temporal co-references. As a comparison, a baseline method that chooses all terms that have lexical overlaps with the query term, can maximally achieve a recall of 49% (= 42/86) because no indirect co-references can be found.

NEER In this experiment we kept all co-references found by NEER without any filtering. This experiment provides an upper bound on the recall for the subsequent experiments. We found that for known periods as well as for found periods recall is high with 89% and 90% respectively. Only very few true co-references were missed and we found at least one correct co-reference for all but two terms. The precision is much lower for known bursts in comparison to the baseline which is a consequence of the higher average number of co-references found. However the recall is statistically significantly higher. The precision of the found bursts is comparable to the baseline and again the recall is significantly higher.

Out of 22 terms with indirect co-references our method was able to find at least one indirect co-reference for 21 terms for both known burst and found bursts. For found bursts no indirect co-reference could be found for *Airtran* because no bursts could be detected. For known bursts no indirect co-references could be found for *Andersen Consulting*.

Some sample queries and their five most frequent direct and indirect co-references for known bursts are shown in Table 7.3. As we can see the results contain co-references of high quality. For *Vladimir Putin*, NEER found four roles *President-elect, Minister, Acting President* and *President*. For *Sean Combs* all but one of his names are present in the top five co-references, missing is only *Puff Daddy* which appears on a lower rank. *Sean Combs Ruiz* is an error caused by the term extraction. The term *Ruiz* is a name of a movie character that was played by Sean Combs in 2001. For *Barack Obama* we found the term *Senator*, the term *President* is missed as it lies outside of the NYTimes timespan.

Considering names with a single token typically decreases the precision for the category *People* because it increases the number of co-occurrences with first names. However, for the categories *Companies* and *Locations* it is necessary to keep single token names, otherwise many names would be missed, e.g., *Burma*. To further improve results an extension to NEER could classify names into different categories. The extended NEER can then keep or discard single token names accordingly and allow different name patterns such as names with non-capital stopwords (e.g., *Union of Myanmar*).

NEER + Correlation filtering Using correlation as a filtering, with $\text{corr}_{\text{min}} = 0.4$, precision increased over the NEER results while recall decreased. For both known and found periods the decrease in recall corresponds to a statistically significant decrease. The recall is higher than that of the baselines but is not competitive to the NEER results and shows that global correlation measure on its own is not an appropriate similarity measure for temporal co-references.

Table 7.3: Terms and their top temporal co-references. *Terms* in *gray* are considered incorrect.

Barack Obama	**Vladimir Putin**	**Sean Combs**
Senator	President-elect Vladimir V Putin	Puffy
State Senator Barack Obama	Minister Vladimir Putin	Sean John
Senator-elect Barack Obama	Acting President Vladimir V Putin	Diddy
Senator Barack Obama	President Vladimir V Putin	Sean Combs Ruiz
Illinois Democrat	Vladimirovich	Sean John Combs

NEER + Document Frequency filtering In this experiment co-references found by NEER were removed if they occurred in more documents than the query term times a scaling factor (sc), as described in Section 7.2.4. The filtering provides a good recall for both found and known periods. The decrease in recall compared to the NEER results is not statistically significant for either found or known periods. With regards to precision both found and known periods show the most competitive performance compared to the baseline and NEER.

We used sc = 10 for $df_{max} \leq 300$, 5 for $300 < df_{max} \leq 800$ and 3 for $df_{max} > 800$. These filtering thresholds as well as the scaling factors were found empirically. Learning these could improve the results for document filtering further.

NEER + Machine Learning We showed that unsupervised filtering can perform well for filtering out erroneous co-references found by NEER. In this experiment we investigated if the results could be further improved by using a limited amount of supervision for training a classifier.[4] We used WEKA (Hall et al., 2009) and the random forest classifier. We trained our classifier on the dataset and used 10-fold stratified cross-validation to determine precision and recall. The precision presented in Table 7.2 is the one found by WEKA, while for recall, we use the predictions and follow the same procedure as for the other experiments.

We remove multiple instances corresponding to several change periods for one term–co-reference tuple. If a tuple (w , c , p_1) is classified as correct and a tuple (w , c , p_2) is classified as incorrect, we consider the correctly classified tuple. This because the filtering will keep at least one instance of co-reference c for the term w. As an example, in one fold we have the tuples tup1 = (*Sean Combs*, *Diddy*, 1988) and tup2 = (*Sean Combs*, *Diddy*, 2005). Tup1 is classified as incorrect by the classifier while tup2 is classified as correct. For the recall calculations, we remove tup1 from the testset and keep tup2, because *Diddy* is found as a correct co-reference for *Sean Combs* for at least one instance and thus counts should contribute to recall. If however both tup1 and tup2 where correctly classified, not removing one of the instances would have positively affected the recall.

For known bursts we got in total 3965 instances where 230 were correct co-references (we accepted combinations of correct names, e.g., *Sean Diddy Combs*). Using the classifier we were able to achieve a 90% precision and only 16 false co-references were classified as correct. The recall of the filter is 65%. For the found bursts there were 21371 instances with 587 correct co-references. The precision of 93% is comparable to that of the known bursts and only 34 false co-references were classified as true co-references. The recall is higher with 83%. The precision and recall values for the machine learning filtering is only for class 1, i.e., all instances that are classified as co-references to a term w. If we include class 0, the precision and recall is 97% for known bursts and 99% found bursts.

The large difference in recall for known and found bursts is most likely due to the small number of correct co-references for known bursts. A reason for the comparably low recall for both

4 Please note that these results differ from those presented in (Tahmasebi et al., 2012a), where the set of instances contained also duplicates which resulted in an artificially high recall.

Table 7.4: Top co-references found for terms from the categories *Location* and *Company* after document frequency filtering for known bursts. For *Myanmar* only *Burma* remains after filtering. *Terms* in *gray* are considered incorrect.

Accenture	**Comcast**	**Czech Republic**	**Myanmar**
Acc. Match Play Championship	Comcast Corporation	Hungary	Burma
Andersen Consulting	AT&T Comcast	Slovakia	

classes can be the acceptance of partial people names as correct. For example, for *Sean Combs* we accepted *John* to be correct because the full name is *Sean John Combs*. However, there are many *John*s and, because of the ambiguity, it is hard for the classifier to determine that *John* is a correct co-reference for *Sean Combs* based only on term frequency features.

The results show potential of the machine learning approach combined with the features chosen for the classification, in particular for the found bursts.

7.4 Discussions

Our experiments show that we are able to find temporal co-references with high recall without depending on external knowledge sources. We found that, even though not all change periods could be found using burst detection (recall 66%), we still managed to get a recall that is comparable to the high recall for the known (correct) change periods. Because every change period captures two co-references, it is possible to capture more co-references than the number of found change periods suggests.

There can be several reasons for a change period not to occur as a burst. In some cases the name change is discussed before the change takes place and, thus, there is a discrepancy between the found burst and the 'true' change period. It is also possible for a name change to correspond to a smaller increase in frequency than other events (possibly such leading up to and causing the name change). Future work is to learn thresholds to find bursts that correspond to change periods or find other methods better suited for change period detection.

We found that co-references cannot be detected in a symmetric way: Finding w_i as a co-reference for w_j does not imply that we can find w_j as a co-reference for w_i. E.g., NEER found *Slovakia* and *Czech Republic* as co-references for *Czechoslovakia*. However, for *Czech Republic*, NEER could not find *Czechoslovakia* (using found or known bursts).

Our experiments show that co-references found for terms from the category *People* have a good accuracy among the top co-references also without filtering. However, for the category *Locations* and *Companies* filtering is needed for achieving high accuracy among the top results. Some examples for companies and locations can be found in Table 7.4. For *Accenture*, *Czech Republic* and *Myanmar* we found an indirect co-reference among the top two terms after document frequency filtering.

By making use of terms from the dataset we ensure that all found temporal co-references can be used for information retrieval on the dataset. The results of Kanhabua and Nørvåg (2010), found using Wikipedia, contain co-references of high quality like *Senator Barack H. Obama Jr*, but many of these do not appear in the NYTimes and thus cannot be used to retrieve documents. By not relying on external resources we enable a robust method that can be applied on any corpus and finds ephemeral co-references like *President-elect George Bush* or *Senator-elect Barack Obama* that do not appear in e.g., Wikipedia. NEER can also be applied to heterogeneous data such as long-term archives as well as Web data and can mix content from several resources.

We approximate entities using contexts defined as co-occurrence graphs created using a sliding window. Instead, entities can be approximated using dictionary resources or other resources that are independent from the text itself. Also human input can be used to help link entities and find evolution. However, it remains an open question if humans can find all types of language evolution. In the case of named entity evolution human input can be helpful, however, in the case of general term to term evolution when the shift is slow and possibly spread over a large demographics, human input might only be of limited help.

7.4.1 Burst Detection for Finding Change Periods

There are various reasons for the burst detection to miss change periods. Firstly, the algorithm used for burst detection varies heavily for different thresholds. Future work will be to find optimal settings for detecting change periods as well as investigate other algorithms for approximating change periods.

Secondly, for an entity that figures often in media, for example because of tours, albums, movies or scandals, it can be difficult to discriminate between bursts that relate to a name change and bursts due to reasons described above. Here a deeper insight into the effects of name changes on media can help us create better models for capturing change periods. This is in particular necessary in domains where many name changes do not make it at all into any external resources.

Thirdly, it is possible that there are systematic discrepancies between true change periods and the bursts used to find change periods. Discussions preceding a change like a country or currency changing names or the delayed notice of a name change can cause change periods to differ by one or several periods. These phenomena can be further investigated and can possibly be helped by making use of external resources.

7.4.2 Relation to Term Concept Graphs

In Chapter 3.2 we defined two functions for finding term concept graphs (TCG). The first function ϕ maps terms to their concepts at one point in time and the second function ψ merges term concept graphs to contain all information about a term. For word sense evolution, the mapping and merging steps are well separated. Concepts are created using word sense discrimination in Chapter 5 and the mapping is described in Chapter 6. For named entity evolution, the procedure is slightly different because named entities are modeled differently from word senses. In addition, for named entity evolution, we are interested in the different terms that point to one concept, instead of fixing one term and following the concepts.

For named entity evolution, the concepts are represented by term contexts and are found using sliding window co-occurrence graphs and filtering as described in this chapter. This means that the mapping function ϕ directly maps to the contexts that we have defined. The function ψ corresponds to the consolidation step. During this step direct co-references and their corresponding contexts are merged interchangeably to find all co-references that point to the same entity w. After the consolidation, all direct and indirect co-references of w point to w. In addition, all direct co-references of the indirect co-references also point to w. In Figure 7.4b we see the resulting context of *Pope Benedict XVI*. All direct co-references of *Pope Benedict XVI*, namely *Pope Benedict* and *Pope* are connected to the context. In addition, the indirect co-reference found in the context is *Cardinal Joseph Ratzinger* which also points to the context (*Vatican* and *German* should be filtered out). Finally, the direct co-references of *Cardinal Joseph Ratzinger*, namely *Cardinal Ratzinger, Joseph Ratzinger* and *Ratzinger* also point to w.

Though it was not evaluated in this thesis, the change periods could be used to identify validity periods for each term-context relation. If w_1 is a co-reference of w over several change

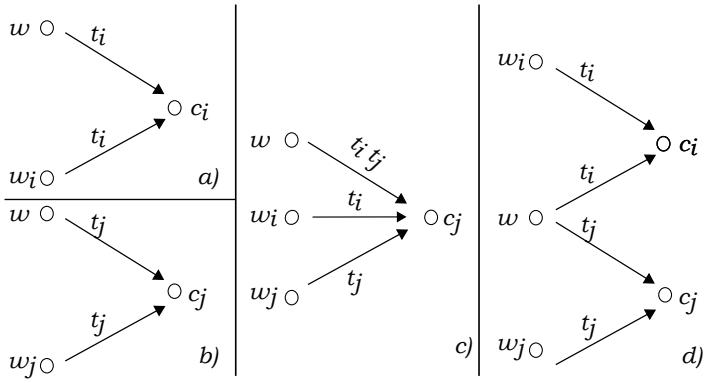


Figure 7.5: Merging of TCG's for term to term evolution. The TCG's in a) and b) from different change periods are merged and if contexts c_i is equal to c_j the results becomes the TCG in c). If $c_i \neq c_j$ the resulting TCG looks like the one in d).

periods p_i to p_k, then the term-context relation is valid at least during $[p_i, p_k]$. Currently, no disambiguation is done in NEER. This means that contexts are merged if their corresponding term can be merged. Assume that we are merging the TCG's shown in Figure 7.5 a and b). If disambiguation was done, then contexts could only be merged assuming that they were similar enough. If two concepts can be considered the same, they can be merged and the resulting TCG would look like the one shown in Figure 7.5 c). If however, the concepts cannot be merged, the resulting TCG would look like in Figure 7.5 d). This means that w has two co-references w_i and w_j but they are not co-references of each other as they have separate contexts. This would correspond to a word having synonyms for its different senses. For example, the term *Google* has the co-reference *Google Inc.* for its *search engine* sense and the co-reference *searching* for its *seek information* sense. However, *searching* and *Google Inc.* are not co-references of each other.

It is important to note that also contexts that correspond to the same entity can experience evolution. When the role of a person, company or location changes, this results in changes in the context. Being a pope means taking on a new role and getting new attributes which in turn means a changed context compared to being a cardinal. However, some parts of the context remains the same, birth dates and locations and physical descriptions like height and shape can be attributes that remain steady over time. This can help to identify when two contexts correspond to the same entity even when they have exhibit certain differences.

7.5 Application – The *fokas* search engine

To showcase the NEER algorithm we created a search engine for querying and visualizing temporal co-references. The system is called *fokas* which stands for *formerly known as* and makes use of the NYTimes corpus and the co-references found by NEER. *fokas* was published by Holzmann et al. (2012) where a more detailed description of the system is available, the implementation of the system is fully credited to the first author.

The search engine follows most modern search engines by providing a search field to enter a query and starting the search by clicking on the search button. For each query that is entered, suggested terms are presented in a roll-down menu as shown in Figure 7.6. The suggested terms are terms with the same starting characters as those already typed in the search filed as well as the known co-references. The user can choose to complete the query by continuing to type or clicking on a suggested term.

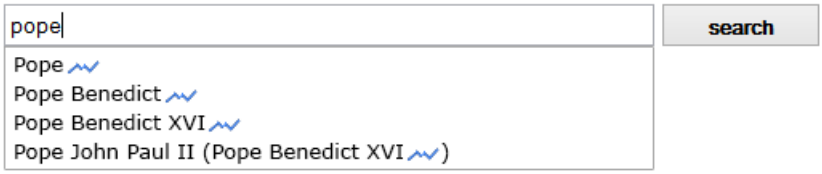


Figure 7.6: Search field in *fokas*, for each query, suggestions are provided while typing.

Once the search has been executed, the system will in part perform like a normal search engine and present the results in a list with a snippet of text to accompany each result. To the right, *fokas* presents the direct and indirect co-references found by NEER. For each co-reference it is possible to view the term frequencies over the collection to help the user to determine if the presented co-reference is correct.

fokas provides the user with the possibility to improve the search results by extending the query with co-references of the original query term. The user can select one or more co-references from the lists of direct or indirect co-references in the sidebar. This will automatically extend the search results with documents containing also the chosen co-references. All search results found through the selected co-references are marked with an icon (yellow start) to help the user identify the newly added documents. The procedure highlights the advantage of an extended search by visualizing the changes in a clear manner. The interface gives the user full control over the terms added to the query, supported by the displayed frequencies of each term. Clicking on a co-reference again in the side bar will remove the co-reference from the query and update the search results.

The *fokas* search engine is a system that takes named entity evolution into account and allows users to query a document collection and be made aware of temporal co-references. The lists of direct and indirect co-references and the frequency chart shown next to the search give users efficient tools for enriching their queries with their temporal variants. The highlighted search results give users direct feedback about the improved results gained by including co-references. However, while *fokas* provides a well selected and filtered set of co-references based on NEER, it is not able to select the best queries for augmenting the search results automatically. *fokas* still requires interaction by the users but provides deeper insight and control, as well as transparency on how *fokas* and NEER work.

There are some issues left to tackle with regards to incorporating named entity evolution in search in an effective way. If we consider the results shown for *Accenture* in Table 7.4 we find that there is one correct indirect co-reference, *Andersen Consulting*, and one incorrect direct co-reference, *Accenture Match Play Championship*. Without additional information it is difficult for a user to distinguish the correct co-reference from the incorrect one. In this case, the term frequencies do not provide enough clues for the user and the user is left to guess or find the answer using other resources. Future work is to extend the clues provided to the user, for example by adding snippets that best describe the relation for each co-reference. Sentences like "*Accenture, the consulting firm that recently changed its name from Andersen Consulting...*"[5] and "*... Accenture Match Play Championship, an $8.5 million World Golf Championships event ...*"[6] can help the user distinguish between the correct and the incorrect co-references. The question is then how to find the most informative sentence to help the user make a decision.

5 The New York Times, August 21, 2001
6 The New York Times, December 6, 2009

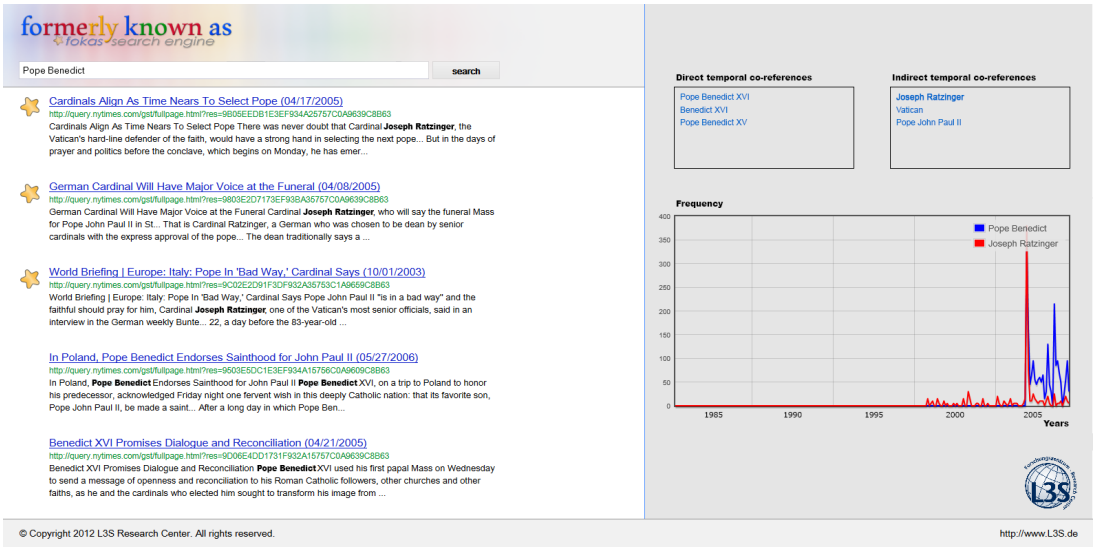


Figure 7.7: Search results as presented by *fokas*. On the right hand side all co-references found by NEER are shown.

7.6 Conclusions and Contributions

In this chapter we presented NEER, an unsupervised approach for named entity evolution recognition to help users find content in long-term archives. NEER overcomes limitations of existing approaches and does not depend on external knowledge sources. We made use of change periods to create term contexts that capture co-references in the same context, thereby avoiding to compare term contexts from vastly different periods in time. Burst detection was used to detect change periods and captured 66% of all change periods. Because term contexts created in change periods capture more than one co-reference, 89-90% of all name changes were found. We used frequency analysis to find direct and indirect co-references by filtering on document frequency as well as using machine learning to classify correct co-references. Using a random forest classifier we achieved a precision of 90% on known periods and 93% on found periods, however, only for found periods did we find acceptable recall of 83%. It remains an open issue to find better unsupervised or limitedly supervised filtering of co-references. All name changes used in our testset are released to encourage further research in this area (Tahmasebi et al., 2012b).

The main contributions presented in this chapter are the following:

- We proposed the use of change periods (i.e., periods with high likelihood of name change) to capture the evolution of one name into another in each context thus avoiding comparison of contexts completely.
- We proposed NEER, a method for named entity evolution recognition that analyzes the context of entities during time periods of change. The proposed method is independent from external knowledge sources and is able to find name changes.
- We described and compared named entity evolution filtering methods, frequency as well as machine learning based, that capture relatedness between co-references in order to increase accuracy.
- To the best of our knowledge, we are the first to provide a quantitative evolution of the quality of named entity evolution recognition. Thus far, evaluation has been made indirectly by means of information retrieval.

- We applied NEER on a standard dataset namely the New York Times Annotated Corpus, to identify named entity evolution and evaluate using precision and recall. We make our testset publicly available to encourage comparison of results.

7.6.1 Limitations and Future Work

The key to NEER are two main assumptions, (1) named entity evolutions are announced to the public repeatedly during a short period of time; and (2) named entity evolutions occur within the same context and are often mentioned in the same sentence.We believe this makes the method applicable to other languages where the same assumptions hold. However, it is not clear if the method is applicable to other types of term to term evolutions where changes occur slowly.

Because there is no disambiguation between entities, the quality of the results depends on the ambiguity of the entity. *Barack Obama* represents an easy case because the name is very unique, however, the artists *Prince* or *Madonna* are more difficult cases because the names are very ambiguous. In general, celebrities are a difficult class of entities for the NEER system; when detecting change periods using burst detection most bursts correspond to scandals. The more scandals a celebrity has been involved in, the less likely it is to find a true change period among the bursts. In the case of *Sean Combs*, the bursts primarily corresponded to a trial where he was the accused. The trial attracted much attention and the papers often mentioned all the artist's names. Therefore, the name changes could be found even if the bursts did not capture most periods. This can however not be assumed for other celebrities and the method should be seen as limited in this respect.

In certain cases, named entity evolution reflect evolution of the entities them selves. For example, after the split of *Czechoslovakia* into two new countries, *Czech Republic* and *Slovakia*, the original entity is no longer a valid entity. In our work, we have chosen a simplified view and consider all relations as symmetric regardless of the validity of the original entity. Future work is to determine, and distinguish between, different relations and find the best way to utilize these relations for finding content in long-term archives.

NEER is a statistical method and bases its foundation on a large quantity of documents rather than a few documents of high quality. For smaller collections or collections with short time spans it is unlikely that NEER would find high quality co-references. For such collections it might be better to make use of rule-based methods instead. Examples of rules that can be used to find that B and A are co-references are *A formerly known as B*, *A also known as B* and *A previously called/named B*.

A future direction for improving the results of NEER is to determining when a named entity has indirect co-references. This would help to reduce the number of false co-references that are presented to the user in cases where the term has no indirect co-references.

Chains of named entity evolutions are another future direction. NEER focused on change periods for one term w and searched for temporal co-references of w in all change periods. This limits the number of names that can be found. In future work, focus should be placed on automatically creating chains of evolution to handle terms with many changes as well as associating the validity period of co-references to each term for example by means of change periods: $\textit{Tsaritsyn} \xrightarrow[1589-1925]{} \textit{Stalingrad} \xrightarrow[1925-1961]{} \textit{Volgograd}$.

Using NEER we are able to achieve a recall of 89% and 90%. Using chains as described above can help increase the recall. To achieve full recall, NEER should be extended to further sample sentences containing the query term from years adjacent to the burst years. A certain number of sentences can be chosen with a probability decaying with the distance from the burst year. Most likely the decaying interval before a burst year should be shorter than the decaying

interval after a burst year; a name change might be discussed a short time before the change occurs but can be referred to long after the name change has occurred.

External resource can help the filtering process. By classifying entities into types, we can prevent entities of different types to be co-references. For example, the head of state is often a co-reference for a country and can be filtered out with such an approach. The applicability and extent to which external resources can be used must be further investigated in future work.

We have chosen to apply NEER only to the NYTimes corpus which is error free and spans only the past decades. It remains future work to investigate the applicability of NEER to older texts, for example, The Times Archive, as well as the effects of using diverse resources, like Blogs, newspapers, Web sites and books.

Chapter 8

Conclusions and Outlook

8.1 Achievements of the Thesis

In this thesis we studied the problem of automatically finding language evolution. We investigated the problems of language evolution with two high-level objectives in mind, namely to help users ***find*** **and** ***interpret*** **content in long-term archives**. We began by analyzing and classifying different types of evolution with respect to our objectives. We further presented a model for describing evolution, namely **term concept graphs**. We found four main classes of evolution which can be described using different term concept graphs. Each class of evolution presents different difficulties when it comes to finding and interpreting.

We continued this thesis with an in-depth analysis of two classes of language evolution and presented models and algorithms for detecting evolution in these classes. The first class of problems which we studied was **word sense evolution**, involving working towards the objective of **interpreting content**. Our method for detecting word sense evolution was twofold; (1) extract word senses for each period in time individually; and (2) compare and group word senses to find word sense evolution.

For evaluation purposes, we made use of The Times Archive with a 201 year time span. As the dataset was older than datasets from which word senses have been extracted in previous research, we began our experiments by evaluating the quality of the extracted word sense clusters. We found that when using word sense discrimination, the **number of word senses** found was highly dependent on the quality of the text and was **subject to OCR errors**. The quality of the found clusters decayed only slightly with time. Hence we concluded that the extracted clusters indeed represent word senses and use the methodology for finding senses which serve as a starting point for finding word sense evolution.

We built our definition of word sense evolution on existing models and extended those by including term concept graphs and transforming the model from a memoryless model to one with memory of all past changes. We made use of automatically extracted word senses and placed all word senses for a term in a term concept graph corresponding to one time point. We iteratively merged term concept graphs, and the word senses in the graphs, for a term. Our merged term concept graph consisted of units which represent individual senses over time and capture the broadening, narrowing and evolution of a single sense. We then grouped these senses into concepts by creating paths which captured merging and splitting of senses as well as polysemy. Different paths for the same term represented unrelated, homonym concepts.

We expanded The Times Archive with the New York Times Annotated Corpus to obtain a **collection with a time span of 222 years** and evaluated on a testset. As no previous testsets were available, we created one by choosing ten terms which have experienced evolution

in the past decades. We were successfully able to find the corresponding evolution for most of these terms and their main senses. In addition, in most cases, the evolution was detected at the time of the actual change, or with a slight delay. While most of the main senses and evolution was found for each term, many of a term's dictionary senses were not found. Indeed, future work should focus on verifying whether these senses are present in the collection and thus could be found, or if they are underrepresented in the collection.

To overcome the lack of senses, we must find additional information by (i) applying other algorithms which can find more word senses; (ii) adding more documents; or (iii) extracting other types of information to complement word senses. The third option could mean extracting topics or context vectors. Moreover, to be able to fully evaluate and validate our results, we need a proper ground truth which is expensive and time consuming to create and which also requires linguists and historians who have an in-depth knowledge of the collection. In this thesis we provided a proof-of-concept for our method, leaving the above mentioned extensions, and the full evaluation for future work.

As a second class of problems we studied **named entity evolution**, which works primarily towards the objective of **finding content**. Our method for finding named entity evolution was based on (1) finding periods in time where the name was likely to change; and (2) creating a context around the name in this time period and searching for named entity evolution in this context. Once all candidate name changes were found, we applied different strategies in order to reduce noise. We found that filtering using document frequency showed potential as it provided reasonably high quality results without requiring manual labeling. Still, there is room for improvement if a good, unsupervised filtering strategy is to be found.

The results were promising and showed that a high number of name changes could be found using this methodology. The key to the method was to **identify the periods of change**. Applying the method to arbitrary periods in time reduced the possibility of finding named entity evolution. The methodology used for finding named entity evolution finds name variants which can be of interest for named entity linking techniques where entities may not necessarily have been completely replaced but are still expressed with variation.

For both classes of evolution we used a set of example terms to evaluate our methodology. We used The Times Archive and the New York Times Annotated Corpus as sample archives and developed **example applications for search and exploration**. We concluded that defining and detecting word sense evolution is an AI complete problem while named entity evolution, on average, is a simpler problem. With this in mind, presenting named entity evolution to users poses fewer difficulties than conveying word sense evolution.

8.2 Outlook

In this thesis we took the backwards perspective, and used existing data whilst trying to find out about as much of the existing language evolution as possible. The data which we used contained editorially controlled and curated text. Current effort is focused on continuously creating new archives with content found on the Web and in the Social Web. When moving from editorially controlled textual resources to the Web, the paradigm changes. Firstly, content is published at a staggering rate and secondly, with a high rate of user generated content a different set of errors are introduced. Abbreviations, slang, grammatical errors and short contexts change the requirements. Looking for nouns and noun phrases in coordination ("*A, B or C*") or patterns ("*A formerly known as B*") will yield in a significantly lower coverage. We must therefore adapt our methods to handle the new requirements whilst also placing **focus on scalability issues**.

Furthermore, we must consider the **preservation** aspect of how to **prepare our archives for future processing** and long-term storage. Storing dictionaries, natural language processing

tools and other resources alongside each archive can help processing in the future. However, data structures, indexes and resources which carry a memory are still needed to fully take advantage of continuously updated archives without requiring re-computation. If crowd computing solutions are to be employed, the processing must take place at the time of archiving in order to avoid the crowd forgetting up-to-date changes in the language. Detected evolution must be stored alongside the archives in appropriate formats to avoid information loss and the risk that future generations will be in the exact same situation as we are in today.

Due to the nature of our problem, large datasets, long time spans and diverse domains, we have opted for **unsupervised methods** without depending on human input or external resources like dictionaries. We have thus placed ourselves in a worst case scenario where no extra help is available. This, however, is rarely the case and **our results** should be considered as **a lower bound for performance**. Future work should explore the possibility of including available resources and the possibility of making use of **crowd sourcing** to improve detection of language evolution. Studies are needed to establish where and in which format human input is most beneficial, particularly, when the input is in the form of the crowd without explicit domain expertise.

The work in this thesis has exclusively **focused on nouns and noun phrases**. The main reason for this choice has been the lack of word sense discrimination methods for other word classes. To some extent the other classes can be targeted using opinion or sentiment mining. However, sentiments are not sufficient to allow users a full understanding of a word. To extend automatic detection of language evolution beyond nouns and noun phrases we need linguistic tools for word sense extraction also for other word classes.

Current efforts to detect language evolution have focused on **tackling each class individually**. To be able to tell the full history of a word, different classes must be tackled and brought together. Term concept graphs representing term to term evolution and word sense evolution must be merged and presented in such a way that they can be efficiently used for search and interpretation. Furthermore, to improve word sense tracking, the results of term to term evolution can be used to *normalize* the word senses and allow for better tracking. This would make it possible to find word senses using *car* and *automobile* or *foolish* and *nice* because the words are considered equivalent over time.

We have excluded **explicit disambiguation** between named entities as well as word senses. This reduces the complexity of the problem and it remains future work to extend our methods with disambiguation techniques.

Finally, in order to fully utilize language evolution we must **add the cultural dimension**. The term *travel* has had the same overall meaning over time; *transporting from location A to location B.* However, this does not tell the full story of the word or the concept represented by the word. Today travel is mostly for business or a happy occasion such as holiday, without any substantial risks involved. In the past, traveling brought with it great dangers and always carried the risk death. This inherent meaning of a word should be communicated to the user to allow for a full interpretation of language and to entail all dimensions of our language and culture. One possible solution is the **use of images**. However, how **to best describe historical word senses** to users remains an open issue and it is our belief that an answer cannot be found within one discipline. Instead, the problem must be solved in a larger community involving among others linguists, historians and computer scientists. In addition, the full evaluation of the automatically detected language evolution requires the whole community.

Bibliography

Google Books. http://books.google.com/, Retrieved 2013-06-26.

Project Gutenberg. http://www.gutenberg.org/, Retrieved 2013-06-26.

The British National Corpus, version 3 (BNC XML Edition). http://www.natcorp.ox.ac.uk/, 2007. Distributed by Oxford University Computing Services on behalf of the BNC Consortium.

Aretha Alencar. CIShell Burst Detection, 2012. Available at http://wiki.cns.iu.edu/display/CISHELL/Burst+Detection.

Kevin Atkinson. GNU Aspell version 0.60.6, 2008. http://aspell.net/.

David Bamman and Gregory Crane. Measuring historical word sense variation. In *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*, JCDL '11, pages 1–10, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0744-4. doi: http://dx.doi.org/10.1145/1998076.1998078.

Klaus Berberich, Srikanta J. Bedathur, Mauro Sozio, and Gerhard Weikum. Bridging the Terminology Gap in Web Archive Search. In *Proceedings of the 12th International Workshop on the Web and Databases (WebDB'09)*, 2009.

Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. DBpedia - A crystallization point for the Web of Data. *Journal of Web Semantics*, 7(3):154–165, September 2009. ISSN 1570-8268. doi: http://dx.doi.org/10.1016/j.websem.2009.07.002.

Jordan Boyd-Graber, David Blei, and Xiaojin Zhu. A Topic Model for Word Sense Disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1024–1033, 2007. URL http://www.aclweb.org/anthology/D/D07/D07-1109.

Leo Breiman. Random Forests. In *Machine Learning*, pages 5–32, 2001.

Morten H. Christiansen and Simon Kirby. *Language evolution*. Studies in the evolution of language. Oxford University Press, 2003. ISBN 9780199244843. URL http://books.google.de/books?id=H_ROF5_z73MC.

Aaron Coburn. Lingua::EN::Tagger - Part-of-speech tagger for English natural language processing, 2008. http://search.cpan.org/~acoburn/Lingua-EN-Tagger-0.15/Tagger.pm.

Paul Cook and Suzanne Stevenson. Automatically Identifying Changes in the Semantic Orientation of Words. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may 2010. European Language Resources Association (ELRA). ISBN 2-9517408-6-7.

Martin C. Cooper. A Mathematical Model of Historical Semantics and the Grouping of Word Meanings into Concepts. *Computational Linguistics*, 32(2):227–248, 2005. doi: http://dx.doi.org/10.1162/0891201054223995.

Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/Gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '92, pages 318–329, New York, NY, USA, 1992. ACM. ISBN 0-89791-523-2. doi: http://dx.doi.org/10.1145/133160.133214.

Koen Deschacht, Marie francine Moens, and Interdisciplinary Centre For Law. Text Analysis for Automatic Image Annotation. In *Proceedings of the 45 th Annual Meeting of the Association for Computational Linguistics. East Stroudsburg: ACL*, 2007.

Beate Dorow. *A Graph Model for Words and their Meanings*. PhD thesis, University of Stuttgart, 2007.

Beate Dorow, Jean-pierre Eckmann, and Danilo Sergi. Using curvature and markov clustering in graphs for lexical acquisition and word sense discrimination. In *Proceedings of the Workshop MEANING-2005*, 2005.

Philip Edmonds and Scott Cotton. SENSEVAL-2: Overview. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5, Toulouse, France, July 2001. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S01-1001`.

Andrea Ernst-Gerlach and Norbert Fuhr. Retrieval in text collections with historic spelling using linguistic and spelling variants. In *Proceedings of the 2007 Joint International Conference on Digital Libraries, (JCDL'07), Vancouver, BC, Canada, June 18-23*, pages 333–341, 2007. ISBN 978-1-59593-644-8. doi: http://dx.doi.org/10.1145/1255175.1255242.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4)*, 2008.

Olivier Ferret. Discovering word senses from a network of lexical cooccurrences. In *Proceedings of the 20th international conference on Computational Linguistics (COLING'04)*, 1326, Geneva, Switzerland, 2004. doi: http://dx.doi.org/10.3115/1220355.1220549.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1219840.1219885.

Mark A. Finlayson. MIT Java Wordnet Interface version 2.1.5, Released under Creative Commons Attribution-NonCommerical Version 3.0 Unported License. `http://projects.csail.mit.edu/jwi/`.

Withrop N Francis. *A standard corpus of present-day edited American English, for use with digital computers.* Brown University Press, Providence, 1964. URL `\url{http://khnt.aksis.uib.no/icame/manuals/brown/}`. Report to the U.S. Office of Education on Cooperative Research Project No. E-007.

T. N. Gadd. PHONIX: The algorithm. *Program: electronic library and information systems*, 24(4):363–366, 1990. doi: http://dx.doi.org/10.1108/eb047069.

Annette Gotscharek, Andreas Neumann, Ulrich Reffle, Christoph Ringlstetter, and Klaus U. Schulz. Enabling information retrieval on historical document collections: the role of matching procedures and special lexica. In *Proceedings of The Third Workshop on Analytics for*

Noisy Unstructured Text Data (AND'09), pages 69–76, 2009a. ISBN 978-1-60558-496-6. doi: http://doi.acm.org/10.1145/1568296.1568309.

Annette Gotscharek, Andreas Neumann, Ulrich Reffle, Christoph Ringlstetter, and Klaus U. Schulz. Enabling information retrieval on historical document collections: the role of matching procedures and special lexica. In *Proceedings of Workshop on Analytics for Noisy Unstructured Text Data*, AND '09, pages 69–76, 2009b. ISBN 978-1-60558-496-6. doi: http://doi.acm.org/10.1145/1568296.1568309.

Ralph Grishman and Beth Sundheim. Message Understanding Conference-6: a brief history. In *Proceedings of the 16th conference on Computational linguistics - Volume 1*, COLING '96, pages 466–471, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/992628.992709.

Kristina Gulordava and Marco Baroni. A distributional similarity approach to the detection of semantic change in the Google Books Ngram corpus. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '11, pages 67–71, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-16-9. URL `http://dl.acm.org/citation.cfm?id=2140490.2140498`.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009. doi: http://doi.acm.org/10.1145/1656274.1656278.

Andreas Hauser, Markus Heller, Elisabeth Leiss, Klaus U. Schulz, and Christiane Wanzeck. Information Access to Historical Documents from the Early New High German Period. In *Digital Historical Corpora – Architecture, Annotation, and Retrieval*, number 06491 in Dagstuhl Seminar Proceedings, 2007. URL `http://drops.dagstuhl.de/opus/volltexte/2007/1057`.

Marti A. Hearst. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, pages 539–545, 1992.

Helge Holzmann, Gerhard Gossen, and Nina Tahmasebi. *fokas*: Formerly Known As – A Search Engine Incorporating Named Entity Evolution. In *Proceedings of COLING 2012: Demonstration Papers*, pages 215–222, Mumbai, India, December 2012. The COLING 2012 Organizing Committee. URL `http://www.aclweb.org/anthology/C12-3027`.

Jeff Howe. The Rise of Crowdsourcing. *Wired Magazine*, 14(6), 06 2006. URL `http://www.wired.com/wired/archive/14.06/crowds.html`.

Ekaterini Ioannou, Wolfgang Nejdl, Claudia Niederée, and Yannis Velegrakis. On-the-Fly Entity-Aware Query Processing in the Presence of Linkage. *PVLDB*, 3(1):429–438, 2010.

R.A. Jarvis and E.A. Patrick. Clustering Using a Similarity Measure Based on Shared Near Neighbors. *IEEE Transactions on Computers*, C-22(11):1025 – 1034, nov. 1973.

Gabriela Kalna and Desmond J. Higham. A clustering coefficient for weighted networks, with application to gene expression data. *AI Commun.*, 20(4):263–271, December 2007. ISSN 0921-7126. URL `http://dl.acm.org/citation.cfm?id=1365534.1365536`.

Amal Chaminda Kaluarachchi, Aparna S. Varde, Srikanta J. Bedathur, Gerhard Weikum, Jing Peng, and Anna Feldman. Incorporating terminology evolution for query translation in text retrieval with association rules. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, (CIKM'10), Toronto, Ontario, Canada, October 26-30*, pages 1789–1792, 2010. doi: http://doi.acm.org/10.1145/1871437.1871730.

Nattiya Kanhabua and Kjetil Nørvåg. Exploiting time-based synonyms in searching document archives. In *Proceedings of the 10th ACM/IEEE Joint Conference on Digital Libraries (JCDL'10)*, pages 79–88, Gold Coast, Queensland, Australia, 2010. doi: http://doi.acm.org/10.1145/1816123.1816135.

Adam Kilgarriff. I Don't Believe in Word Senses. *Computers and the Humanities*, 31(2): 91–113, 1997. doi: http://dx.doi.org/10.1023/A:1000583911091.

Jon M. Kleinberg. Bursty and hierarchical structure in streams. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, pages 91–101, 2002. doi: http://doi.acm.org/10.1145/775047.775061.

S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:49–86, 1951.

Christopher Kunz, Nina Tahmasebi, Thomas Risse, and Matthew Smith. Detecting Credential Abuse in the Grid Using Bayesian Networks. In *Proceedings of the 12th IEEE/ACM International Conference on Grid Computing (GRID), 2011*, pages 114 –120, 2011. doi: http://dx.doi.org/10.1109/Grid.2011.23.

Jey Han Lau, Paul Cook, Diana McCarthy, David Newman, and Timothy Baldwin. Word Sense Induction for Novel Sense Detection. In Walter Daelemans, Mirella Lapata, and Lluís Màrquez, editors, *EACL*, pages 591–601. The Association for Computer Linguistics, 2012. ISBN 978-1-937284-19-0.

Claudia Leacock and Martin Chodorow. Combining local context and WordNet similarity for word sense identification. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*, pages 265–283. The MIT Press, 1998.

Claudia Leacock, Geoffrey Towell, and Ellen Voorhees. Corpus-Based Statistical Sense Resolution. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 260–265, March 1993.

Claudia Leacock, Martin Chodorow, and George A. Miller. Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics*, 24(1):147–165, 1998.

Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM. ISBN 0-89791-224-1. doi: http://dx.doi.org/10.1145/318723.318728.

Vladimir I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviets Physics Doklady*, 10(8):707–710, 1966.

Esther Levin, Mehrbod Sharifi, and Jerry T. Ball. Evaluation of Utility of LSA for Word Sense Discrimination. In Robert C. Moore, Jeff A. Bilmes, Jennifer Chu-Carroll, and Mark Sanderson, editors, *HLT-NAACL*. The Association for Computational Linguistics, 2006.

Dekang Lin. Using Syntactic Dependency as Local Context to Resolve Word Sense Ambiguity. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 64–71, Madrid, Spain, July 1997. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/976909.979626. URL `http://www.aclweb.org/anthology/P97-1009`.

Dekang Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 2*, ACL '98, pages 768–774, Stroudsburg, PA, USA, 1998a. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/980691.980696.

Dekang Lin. Automatic Retrieval and Clustering of Similar Words. In *Proceedings of the 17th international conference on Computational Linguistics*, pages 768–774, Montreal, Quebec, Canada, 1998b.

Yu-Ru Lin, Yun Chi, Shenghuo Zhu, Hari Sundaram, and Belle L. Tseng. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In *Proceeding of the 17th international conference on World Wide Web (WWW'08)*, pages 685–694, New York, NY, USA, 2008. ACM. ISBN 9781605580852. doi: http://dx.doi.org/10.1145/1367497.1367590.

Marie-Catherine De Marneffe and Pierre Dupont. Comparative study of statistical word sense discrimination techniques. In *Proceedings of JADT 2004: 7th International Conference on the Statistical Analysis of Textual Data.*, 2004.

Qiaozhu Mei and ChengXiang Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining (KDD'05)*, pages 198–207, New York, NY, USA, 2005. ACM. ISBN 1-59593-135-X. doi: http://doi.acm.org/10.1145/1081870.1081895.

George A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38:39–41, 1995.

Roberto Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, February 2009. ISSN 0360-0300. doi: http://doi.acm.org/10.1145/1459352.1459355.

Oxford University Press. The Oxford English Dictionary 2nd ed. 1989. OED Online. Oxford University Press. 4 Apr. 2000, 2000. `http://dictionary.oed.com`.

Gergely Palla, Albert-László Barabasi, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446:664–667, 2007.

Patrick Pantel and Dekang Lin. Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'02)*, pages 613–619, Edmonton, Alberta, Canada, 2002. ACM. ISBN 1-58113-567-X. doi: http://dx.doi.org/10.1145/775047.775138.

Ted Pedersen and Rebecca Bruce. Distinguishing Word Senses in Untagged Text. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 197–207, Providence, RI, 1997. URL `http://arxiv.org/abs/cmp-lg/9706008`.

Ted Pedersen and Rebecca Bruce. Knowledge lean word-sense disambiguation. In *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, AAAI '98/IAAI '98, pages 800–805, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence. ISBN 0-262-51098-7. URL `http://dl.acm.org/citation.cfm?id=295240.295807`.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. WordNet:: Similarity - Measuring the Relatedness of Concepts. In Deborah L. McGuinness and George Ferguson, editors, *AAAI*, pages 1024–1025. AAAI Press / The MIT Press, 2004. ISBN 0-262-51183-5.

Thomas Pilz, Wolfram Luther, Norbert Fuhr, and Ulrich Ammon. Rule-based Search in Text Databases with Nonstandard Orthography. *Literary and Linguistic Computing*, 21(2):179–186, 2006. doi: http://dx.doi.org/10.1093/llc/fql020. URL `http://llc.oxfordjournals.org/content/21/2/179.abstract`.

Juan Pino and Maxine Eskenazi. An application of latent semantic analysis to word sense discrimination for words with related and unrelated meanings. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, EdAppsNLP '09, pages 43–46, Morristown, NJ, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-37-4. URL `http://portal.acm.org/citation.cfm?id=1609843.1609849`.

Bogdan Pogorelc, Artur Lugmayr, Björn Stockleben, Radu-Daniel Vatavu, Nina Tahmasebi, Estefanía Serral, Emilija Stojmenova, Bojan Imperl, Thomas Risse, Gideon Zenz, and Matjaž Gams. Ambient bloom: new business, content, design and models to increase the semantic

ambient media experience. *Multimedia Tools and Applications*, pages 1–26, 2012. ISSN 1380-7501. doi: http://dx.doi.org/10.1007/s11042-012-1228-4.

Amruta Purandare and Ted Pedersen. Word Sense Discrimination by Clustering Contexts in Vector and Similarity Spaces. In *Proceedings of Conference on Computational Natural Language Learning (CoNLL)*, pages 41–48, 2004.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th International Conference on World Wide Web, (WWW'11), Hyderabad, India, March 28 - April 1*, pages 337–346, 2011. ISBN 978-1-4503-0632-4. doi: http://doi.acm.org/10.1145/1963405.1963455.

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95*, pages 448–453. Morgan Kaufmann, 1995.

Thomas Risse, Stefan Dietze, Diana Maynard, Nina Tahmasebi, and Wim Peters. Using Events for Content Appraisal and Selection in Web Archives. In *Proceedings of Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011), in conjunction with the 10th International Semantic Web Conference 2011 (ISWC 2011)*, 2011.

Sergio Roa, Valia Kordoni, and Yi Zhang. Mapping between Compositional Semantic Representations and Lexical Semantic Resources: Towards Accurate Deep Semantic Parsing. In *Proceedings of ACL-08: HLT, Short Papers*, pages 189–192, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P08/P08-2048`.

Alfred Roslin Bennett. *The Telephone Systems of the Continent of Europe*. Longmans, Green and CO, 1895. URL `http://archive.org/stream/telephonesystems00bennrich\#page/332/`.

Eyal Sagi. Nouns are more stable than Verbs: Patterns of semantic change in 19th century English. *The 32nd Annual Conference of the Cognitive Science Society*, 2010.

Eyal Sagi, Stefan Kaufmann, and Brady Clark. Semantic density analysis: comparing word meaning across time and phonetic space. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, GEMS '09, pages 104–111, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1705415.1705429`.

Evan Sandhaus. The New York Times Annotated Corpus. Linguistic Data Consortium, Philadelphia, 2008.

Helmut Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the International Conference on New Methods in Language Processing*, 1994.

Heinrich Schütze. Automatic Word Sense Discrimination. *Computational Linguistics*, 24(1): 97–123, 1998.

Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. LINDEN: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st World Wide Web Conference 2012, (WWW'12), Lyon, France, April 16-20*, pages 449–458, 2012. ISBN 978-1-4503-1229-5. doi: http://doi.acm.org/10.1145/2187836.2187898.

Myra Spiliopoulou, Irene Ntoutsi, Yannis Theodoridis, and Rene Schult. MONIC: modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'06)*, pages 706–711, New York, NY, USA, 2006. ACM. ISBN 1-59593-339-5. doi: http://doi.acm.org/10.1145/1150402.1150491.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-654-7. doi: http://doi.acm.org/10.1145/1242572.1242667.

Nina Tahmasebi. Automatic Detection of Terminology Evolution. In *Proceedings of On the Move to Meaningful Internet Systems: OTM 2009 Workshops, Vilamoura, Portugal*, volume 5872 of *Lecture Notes in Computer Science*, pages 769–778. Springer, 2009. doi: http://dx.doi.org/10.1007/978-3-642-05290-3_93.

Nina Tahmasebi, Tereza Iofciu, Thomas Risse, Claudia Niederée, and Wolf Siberski. Terminology Evolution in Web Archiving: Open Issues. In *Proceedings of 8th International Web Archiving Workshop (IWAW'08), Aarhus, Denmark, 18th & 19th Sep. 2008*, 2008. URL `http://iwaw.net/08/IWAW2008-Tahmasebi.pdf`.

Nina Tahmasebi, Sukriti Ramesh, and Thomas Risse. First Results on Detecting Term Evolutions. In *Proceedings of 9th International Web Archiving Workshop (IWAW'09) in conjunction with ECDL 2009*, 2009.

Nina Tahmasebi, Kai Niklas, Thomas Theuerkauf, and Thomas Risse. Using word sense discrimination on historic document collections. In *Proceedings of the 2010 Joint International Conference on Digital Libraries, (JCDL'10), Gold Coast, Queensland, Australia*, pages 89–98, 2010a. doi: http://doi.acm.org/10.1145/1816123.1816137.

Nina Tahmasebi, Gideon Zenz, Tereza Iofciu, and Thomas Risse. Terminology Evolution Module for Web Archives in the LiWA Context. In *Proceedings of 10th International Web Archiving Workshop (IWAW'10) in conjunction with iPRES in Vienna, Austria*, 2010b.

Nina Tahmasebi, Gerhard Gossen, Nattiya Kanhabua, Helge Holzmann, and Thomas Risse. NEER: An Unsupervised Method for Named Entity Evolution Recognition. In *Proceedings of COLING 2012*, pages 2553–2568, Mumbai, India, December 2012a. The COLING 2012 Organizing Committee. URL `http://www.aclweb.org/anthology/C12-1156`.

Nina Tahmasebi, Gerhard Gossen, Nattiya Kanhabua, and Thomas Risse. Named Entity Evolution Dataset. Available online at `http://www.l3s.de/neer-dataset/`, 2012b.

Nina Tahmasebi, Gerhard Gossen, and Thomas Risse. Which Words Do You Remember? Temporal Properties of Language Use in Digital Archives. In *Proceedings of the Second International Conference on Theory and Practice of Digital Libraries - TPDL 2012, Paphos, Cyprus*, volume 7489, pages 32–37. Springer, 2012c. ISBN 978-3-642-33289-0. doi: http://dx.doi.org/10.1007/978-3-642-33290-6_4.

Nina Tahmasebi, Kai Niklas, Gideon Zenz, and Thomas Risse. On the applicability of word sense discrimination on 201 years of modern english. *International Journal on Digital Libraries*, 13(3-4):135–153, 2013. ISSN 1432-5012. doi: 10.1007/s00799-013-0105-8. URL `http://dx.doi.org/10.1007/s00799-013-0105-8`.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 2004.

The Times. The SIXTH NIGHT. By His MAJESTY's Company.. In *London, England, Saturday, January 01, 1785; pg. 1; Issue 1.* Gale Document Number: CS52116209, 1785.

The Times. Sestini's benefit last night at the Opera-House was overflowing with the fashionable and gay. In *London, England, Friday, April 27, 1787; pg. 3; Issue 736.* Gale Document Number: CS50726043, 1787.

The Times. We are at length able to meet... In *London, England, Saturday, December 03, 1814; pg. 2; Issue 9382.*, 1814.

DIPLOMATIC CORRESPONDENT The Times. Menace To The Volga. In *London, England, Friday, July 17, 1942; pg. 3; Issue 49290.* Gale Document Number: CS52116209, 1942.

Peter D. Turney and Michael L. Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.*, 21(4):315–346, October 2003. ISSN 1046-8188. doi: http://dx.doi.org/10.1145/944012.944013.

Stijn van Dongen. *Graph Clustering by Flow Simulation.* PhD thesis, University of Utrecht, 2000.

Xuerui Wang and Andrew McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 424–433, New York, NY, USA, 2006. ACM. ISBN 1-59593-339-5. doi: http://dx.doi.org/10.1145/1150402.1150450.

Duncan J. Watts and Steven Strogatz. Collective dynamics of "small-world" networks. *Nature*, 393:440–442, 1998.

Eric W. Weisstein. Correlation Coefficient, 2012a. `http://mathworld.wolfram.com/CorrelationCoefficient.html`.

Eric W. Weisstein. Covariance, 2012b. `http://mathworld.wolfram.com/Covariance.html`.

Derry Tanti Wijaya and Reyyan Yeniterzi. Understanding semantic change of words over centuries. In *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, DETECT '11, pages 35–40, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0962-2. doi: http://dx.doi.org/10.1145/2064448.2064475.

Gideon Zenz, Nina Tahmasebi, and Thomas Risse. Language Evolution On The Go. In *Proceedings of the 3rd International Workshop on Semantic Ambient Media Experience (NAMU Series) (SAME 2010) in conjunction with AmI-10 in Malaga, Spain*, 2010.

Gideon Zenz, Nina Tahmasebi, and Thomas Risse. Towards mobile language evolution exploitation. *Multimedia Tools and Applications*, 66(1):147–159, 2013. ISSN 1380-7501. doi: http://dx.doi.org/10.1007/s11042-011-0973-0.

List of Figures

List of Tables

Appendix A

Word Sense Evolution Examples

In this Appendix we will provide an extract of the automatically discovered word sense evolution for the testset presented in Chapter 6, Table 6.1. The term *tape* was presented in full in Chapter 6. The terms *aeroplane*, *rock* and *travel* where discussed in the same chapter and therefore we only present the corresponding figures. Finally, the remaining six terms are briefly discussed in this appendix and an extract of the units, the internal clusters and their terms is presented here. We explicitly leave out relations between the units because of the small number of units.

Gay

For the term **gay**, Table A.1, we find a path starting in 1990 where terms like *lesbian*, *homosexual* and *bisexual* are grouped in the first unit. Near the end of 1990's the terms *transgender* and *human rights* come into the unit. This unit is merged with a unit where *gay* is a minority group, with terms like *hispanic*, *black*, *white* and *asian*. Mid 1990's a unit centered around family and friends appear and in 2004 we find a unit where *gay* shows signs of a clear acceptance as a status by being placed in a unit with terms like *married*, *single*, *female*, *male* and *straight*. There are no paths relating to the *happy* sense of term, but there exists some single units, among others one with the terms *gay entertainment*, *dancing* and *cheerful music*. It should be clear that the term in its *happy* sense is not a noun or noun phrase and hence not explicitly targeted by our algorithms. The concept corresponds in time to the expected with a slight delay.

Tank

For the term **tank**, Table A.2, we see clear indications for its *liquid container* sense with terms like *cistern* and *water closet* already starting 1829. Around WWI we have a distinct path for the *armored car* sense. Terms like *artillery*, *infantry* and *gun* discriminate this concept from its other concepts. The unit from 1917-2001 is not shown in full in the table due to its large size. However, while the units for the *armored car* sense are well grouped into one path and appear correctly in the timeline, the *liquid container* senses are distributed among several paths.

Cool

For the term **cool**, Table A.3, we find three main paths, the first related to its *weather* sense, with *cloudy*, *shower* and *rain*. This path ends in a unit from 1989-1995 which has a baking context, with terms like *peel*, *cut*, *oven* and *bake*. The second path corresponds to cool materials with terms like *silk*, *cloth* and *light weight*. In 1971 we find the term *cool* in a unit with *calm*, *collected* and *efficient* symbolizing a way of being. The introduction of this sense corresponds to the first usages of the *collected* sense with a slight delay. Because this sense is deemed as slang, it seems reasonable that it appears some years later in a newspaper.

Flight

For the term **flight**, Table A.4 we were able to detect units corresponding to the senses that we manually found in Chapter 5.4 and are therefore not shown here. In addition we found a unit where flight is related to fabrics and sowing with terms like *necktie*, *knitting thread*, *sowing* and *quality*. We cannot find any paths and the *commercial flight* sense appears decades later than expected.

Mouse

The term **mouse**, Table A.5, has two distinct paths corresponding to its *animal* sense as well as its *computer* sense. The first path groups the term with mostly smaller animals like *cats*, *rats* and *rabbits*. Its second sense appears in the second path with a unit in 1985, with *icon*, *window* and *mouse*, and in 1988, also with *joystick*. From 1995 to 2007 the mouse establishes its position as an accessory to a computer by being involved a unit where most clusters contain the terms *mouse*, *monitor* and *keyboard*. The first cluster in this unit contains also the term *cat* and is therefore wrongly placed also in the path with *mouse* as an animal. The introduction of the *computer mouse* sense corresponds well to the expected evolution.

Telephone

The term **telephone**, Table A.6, appears in the 1880's and is closely linked to the *telegraph*. In the 1920's the same unit evolves to include terms like *railway* and *post office*. *Telephone* corresponds to many units and paths which in general, during the 19th and early 20th century, correspond to utilities like *lift* and *electric lightning* in houses. In the early-mid 20th century *telephone* is something that moves in to the house and is used to describe the indoor living, illustrated by terms like *hot water*, *running water*, *gas* and *bathroom*. End of 20th century it becomes more related to household products like *television*, *freezer*, *radio* and *washing machine*, showing its evolution from a utility that belongs to the house and the neighborhood, to a common, everyday product. An interesting development occurs around 1990's. The *telephone* is placed together with terms like *radio*, *television* and *newspaper* indicating its use for spreading news. Around the turn of the century, we find *telephone* also with other electronic communication like *internet*, *email*, *fax* and *web pages*. The early units fit the expected general evolution of the term, however, like with the term *travel* it is hard to determine the correctness of the evolution found for *telephone*.

Table A.1: Extract of units for the term *gay*.

Year		Cluster terms
UNIT:	1990, 1991, 1993, 1996, 1997, 1998, 2000, 2001, 2003, 2006	
	1990	gay, homosexual, intravenous drug, bisexual man
	1991	heterosexual, lesbian student, gay, lesbian, homosexual, bisexual
	1993	square, married, lesbian, homosexual, bisexual, white, gay, heterosexual
	1996	gay, heterosexual, lesbian, homosexual, bisexual
	1997	gay, transgender, lesbian, bisexual
	1998	lesbian political organization, gay, human right campaign, nation
	2000	gay, transgender, lesbian, bisexual
	2001	gay man, white, straight, gay, lesbian, black
	2003	latino, transgender, bisexual man, transgendered people, transgender student, hispanic
	2006	transgender people, transgendered alumnus network, transgender
UNIT:	1992, 1993, 1994, 1995, 1996, 1997, 2002, 2004, 2005	
	1992	hispanic, woman, gay, black, white
	1993	white, black, straight, gay, female, male
	1994	gay, white, asian, african, black, native american, latino, hispanic, africanamerican
	1995	and percent, african, white man, arabamerican, gay, jew, other minority, jewish
	1996	hispanic, straight, gay, american, asian, white, black, male
	1997	jew, gay, hasidim, woman, asian, white
	2002	white, straight, gay, female, black, latino, male
	2004	male, white, female, straight, gay, black, woman, latino, gay man
	2005	white, straight, black, gay, female, male, female brain
UNIT:	2004, 2007	
	2004	true, white, black, married, single, male, gay, female, ms, straight, male force
	2007	female, male, straight, gay

Table A.2: Extract of units for the term *tank*. Second unit only displays some of the internal clusters.

Year		Cluster terms
UNIT:	1829, 1842, 1951	
	1829	loin, hip, pipe, pantiling, cistern, tank
	1842	water closet, pot, pan, cistern, tank
	1951	cylinder, pipe, tank
UNIT:	1917, 1938, 1939, 1940, 1941, 1942, 1943, 1945, 1989, 1992, 1993, 2001	
	1917	infantry, piping, shafting, gun, aeroplane, tank
	1938	infantry, artillery, tank
	1939	antitank gun, gun, tank
	1940	gun, tank, plane, aeroplane, ship, cargo, explosive, engine

Table A.3: Extract of units for the term *cool*.

Year		Cluster terms
UNIT:	1879, 1909, 1911, 1913, 1928, 1955	
	1879	cloudy, fair, cool, weather fine, cold
	1909	rain, changeable, showery weather, cold, moderate, showery, shower, cool, cloudy
	1911	rain, fair, fair interval, fog, local mist, moderate, overcast weather, cold, cool
	1913	cloudy, changeable, gusty, occasional rain, cool, showery, fine interval
	1928	bright, rainy, particularly, bright interval, showery, shower, local, cool, cloudy, dull
	1955	rain, occasional rain, shower, cloudy, cool
UNIT:	1989, 1990, 1995	
	1989	peel, cool, oven, bake
	1990	peel, dice, cool
	1995	drain, cool, spoon, peel, cut, scoop
UNIT:	1897, 1915, 1979	
	1897	elastio, air, silk, elastic, light as air, cool
	1915	handsomely, airy, cloth, whilst light, light, cool
	1979	lightweight, cool, fits
UNIT:	1971, 1972, 1974, 1980	
	1971	calm, cool, collected
	1972	calm, cool, competent secretary
	1974	competent, calm, cool
	1980	calm, efficient, cool, collected

Table A.4: Extract of units for the term *flight*.

Year		Cluster terms
UNIT:	1913, 1914	
	1913	durability, twist, knitting thread, flight, necktie, sewing, quality, shrink
	1914	durability, flight, quality
UNIT:	1968, 1981, 2007	
	1968	car hire, scheduled flight, maid, flight, luxury hotel
	1981	court fee, flight, luxury hotel
	2007	car, hotel, flight, fight

Table A.5: Extract of units for the term *mouse*.

Year		Cluster terms
UNIT:	1890, 1927, 1967, 1968, 1972, 1982, 1985	
	1890	cated cat, tina educated cat, rat, mouse, cat
	1927	cat, rat, mouse
	1967	cat, dog, rat, mouse
	1968	mighty hercule, cat, vole, dog, rat, mouse, guinea pig, horse
	1972	cat, rabbit, dog, rat, mouse, monkey
	1982	rabbit, mouse, cat, dog, won ton ton, canary
	1985	eland, lion, giraffe, ape, horse, cattle, mouse, pig, frog, white rhino
UNIT:	1988, 1995, 1996, 2002, 2005	
	1988	joystick, mouse, keyboard
	1995	monitor, keyboard, mouse, cat
	1996	mouse, keyboard, monitor, printer
	2002	mouse, monitor, keyboard
	2005	mouse, monitor, keyboard

Table A.6: Extract of units for the term *telephone*.

Year	**Cluster terms**
UNIT:	1882, 1887, 1919, 1921, 1922, 1923, 1926, 1927, 1929, 1932, 1934, 1936, 1939
1882	hydraulic lift, electric light, telephone, lift
1887	cold, gas, floor, electrio light, landing, bath, passenger lift, sea plunge, english billiard
1919	telephone, secondary staircase, central heating, gas, passenger lift, lift, dinner lift
1922	floor, central, breakfast, light, seven bed, telephone, three reception, ten bed
1923	telephone, water, bath, garage, electric light, nonbasement, detached, freehold, modern
1926	table tennis, drawing room, plate, dance, music, electric light, telephone, bath, gas fire
1927	taxis, running water, constant bot, independent boiler, rate, beating, lighting
1929	mixture device, floor, constant bot, rdens, independent boiler, machinery, power
1932	india, floor, australia, luxurious accommodation, ceylon, burma, lodging clothes
1934	cover rent, lighting, fitted basin, phone, telephone, gas fire, electric light, basin
1936	tar, gas, modern drainage, garage, main drainage, watch, and water, fire, power
1939	tiled bath, basin, reception, floor, tastefully, good drainage, parquet
UNIT:	1932, 1937, 1938, 1940, 1942, 1943, 1947, 1949, 1950, 1957
1932	telephone, hot water, bed room, bed light, light, water, gas fire, boxspring bed, gas tire
1937	running water, hot water, gas, bath room, and telephone, and bath, electric light, telegraph
1938	bed, electric, ripe, bath room, and water, co, and power, telephone, main water, electricity
1940	lift, electricity, bed room, telephone, kitchen, bath room, cooker, water, gas, electric light
1942	telephone, central heating, running water, gas, vispring mattress, bath room, electric fire
1943	telephone, central heating, running water, gas, electricity, telephonc, electric fire, gas fire
1947	electric, bath room, ample water, company water, main electric light, telephone, electricity
1949	drainage, bath room, hot water, main, heating, power, telephone, main electricity
1950	battery, heating, telephone, bath room, wireless, lighting
1957	telephone, private bathroom, breakfast, single bedroom, double bedroom
UNIT:	1993, 1994, 1996, 1997, 1999, 2001, 2003, 2005
1993	fax, radio, television, telephone, computer, telephone line
1994	publishing, entertainment, radio, television, cable company, computer, telephone
1996	telephone, television, radio
1997	telephone, television, radio, newspaper
1999	trash compactor, highspeed internet service, car, jet, shampoo, personal computer, radio
2001	telephone, radio, television, newspaper
2003	sneakers, furniture, ski boot, table, fetus, knife, roller blade, bright, television, telephone
2005	telephone, wireless telephone industry, television, cable, telephone company

NINA N. TAHMASEBI

PERSONAL INFORMATION

Born in Iran, 29 October 1982

title	M.Sc.
nationality	Swedish
email	nina@tahmasebi.se
website	http://www.tahmasebi.se

WORK EXPERIENCE

2013–ongoing — PostDoc, Chalmers University of Technology

Chalmers

Project assistant/PostDoc in Computer Science & Engineering Department working in the project *Towards a knowledge-based culturomics*.

2008–Aug 2013 — Research Assistant, L3S Research Center

L3S Research Center

Research assistant working on a EU project *Archive Communities Memories* (Arcomem). Work package leader for work package on Events, Topics, Entities and their Dynamics. Previously I worked on EU project *Living Web Archives* (LiWA) on terminology evolution. http://www.larcomem.eu , http://www.liwa-project.eu.

Sep-Dec 2009 — Research Visit, MPII — Saarbrücken

MPII

Two month visit at Max Planck Institute of Informatics within the scope of the LiWA project under the supervision of Prof. Gerhard Weikum.

2007 — Teaching assistant, Chalmers University of Technology — Sweden

Chalmers

Teaching assistant for a basic course in Mathematics at Chalmers Lindholmen (2 months)

2007 — Teaching assistant, Chalmers University of Technology — Sweden

Chalmers

Teacher for a group of 10 students where I planned and gave lectures as well as exercise sessions in a Summer course "Introduction to mathematics"(2 months)

2007 — Teaching assistant, Chalmers University of Technology — Sweden

Chalmers

Teaching exercises in "Matematisk Statistik för K" (2 months)

EDUCATION

2008–2013 — Leibniz Universität Hannover

Ph.D. Studies

Ph.D. Student in Computer Science at L3S Research Center and Leibniz Universität Hannover. Title of thesis: *Models and Algorithms for Automatic Detection of Language Evolution. Towards Finding and Interpreting of Content in*

Long-Term Archives. Best paper award for Ph.D. paper in On The Move Academy 2009. Courses taken in Internet Security, Networks and Protocols, Information Retrieval and Requirements Engineering
Advisors: Prof. Wolfgang Nejdl & Prof. Erich Neuhold
Mentor: Dr. Thomas Risse

2006-2008 Chalmers University of Technology, Sweden

Master of Science in Engineering Mathematics

Principal topics: Optimization and Advanced Algorithms. Title of thesis: *Improving Web Recommendation Systems Using Hierarchical Text Classification Methods*
Examiner: Ass. Prof. Devdatt Dubhashi
Supervisor: Oskar Sandberg

2003-2007 Göteborg University, Sweden

Bachelor of Science in Mathematics

Studies in theoretical Mathematics and Mathematical Statistics. Title of thesis: *Isolating Subgraph Algorithm, An approach to Personalized Search Engines*
Examiner: Ass. Prof. Patrik Albin & Ass. Prof. Devdatt Dubhashi
Supervisor: Oskar Sandberg & Libertad Tansini

PUBLICATIONS

2013 On the Applicability of Word Sense Discrimination on 201 Years of Modern English

IJDL

International Journal on Digital Libraries. Authors: Nina Tahmasebi, Kai Niklas, Gideon Zenz & Thomas Risse

2012 Ambient bloom: new business, content, design and models to increase the semantic ambient media experience

MTAP

Multimedia Tools and Applications. Authors: Bogdan Pogorelc, Artur Lugmayr, Björn Stockleben, Radu-Daniel Vatavu, Nina Tahmasebi, Estefanìa Serral, Emilija Stojmenova, Bojan Imperl, Thomas Risse, Gideon Zenz & Matjaž Gams

2012 *fokas*: Formerly Known As – A Search Engine Incorporating Named Entity Evolution

Coling

International Conference on Computational Linguistics. Authors: Helge Holzmann, Gerhard Gossen & Nina Tahmasebi

2012 NEER: An Unsupervised Method for Named Entity Evolution Recognition

Coling

International Conference on Computational Linguistics. Authors: Nina Tahmasebi, Gerhard Gossen, Helge Holzmann, Nattiya Kanhabua & Thomas Risse

2012 Which Words Do You Remember? Temporal Properties of Language Use in Digital Archives

TPDL

International Conference on Theory and Practice of Digital Libraries. Authors: Nina Tahmasebi, Gerhard Gossen & Thomas Risse

2012 Towards mobile language evolution exploitation

MTAP

Multimedia Tools and Applications. Authors: Gideon Zenz, Nina Tahmasebi & Thomas Risse

2011 Towards automatic language evolution tracking, A study on word sense tracking

EvoDyn *Workshop on Knowledge Evolution and Ontology Dynamics* (ISWC). Authors: Nina Tahmasebi , Thomas Risse & Stefan Dietze

2011 Using Events for Content Appraisal and Selection in Web Archives

Derive *Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web* (ISWC). Authors: Thomas Risse, Stefan Dietze, Diana Maynard, Nina Tahmasebi & Wim Peters

2011 Detecting Credential Abuse in the Grid using Bayesian Networks

Grid *IEEE/ACM Conference on Grid Computing*. Authors: Christopher Kunz, Nina Tahmasebi, Thomas Risse & Matthew Smith

2010 Language Evolution On The Go

SAME *Workshop on Semantic Ambient Media Experience* (AmI). Authors: Gideon Zenz, Nina Tahmasebi & Thomas Risse

2010 Terminology Evolution Module for Web Archives in the LiWA Context

IWAW *International Web Archiving Workshop* (iPRES). Authors: Nina Tahmasebi, Gideon Zenz, Tereza Iofciu & Thomas Risse

2010 Using Word Sense Discrimination on Historic Document Collections

JCDL *ACM/IEEE Joint Conference on Digital Libraries*. Authors: Nina Tahmasebi, Kai Niklas, Thomas Theuerkauf & Thomas Risse

2009 Automatic Detection of Terminology Evolution

OTMA *On the Move Academy* (OTM). Authors: Nina Tahmasebi

2009 First Results on Detecting Term Evolutions

IWAW *International Web Archiving Workshop* (ECDL). Authors: Nina Tahmasebi, Sukriti Ramesh & Thomas Risse

2008 Terminology Evolution in Web Archiving: Open Issues

IWAW *International Web Archiving Workshop* (ECDL). Authors: Nina Tahmasebi, Tereza Iofciu, Thomas Risse, Claudia Niederée & Wolf Siberski

October 18, 2013