Discovering and Disambiguating Named Entities in Text

Johannes Hoffart

Dissertation
zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
der Naturwissenschaftlich-Technischen Fakultäten
der Universität des Saarlandes

Saarbrücken
2015

Dean	Prof. Dr. Markus Bläser
Colloqium	12.02.2015 Saarbrücken

Examination Board

Supervisor and Reviewer	Prof. Dr.-Ing. Gerhard Weikum
Reviewer	Prof. Dr. techn. Wolfgang Nejdl
Reviewer	Prof. Dr. Hans Uszkoreit
Chairman	Prof. Dr. Dietrich Klakow
Research Assistant	Dr. Lili Jiang

Abstract

Discovering entities such as people, organizations, songs, or places in natural language texts is a valuable asset for semantic search, machine translation, and information extraction. A key challenge is the ambiguity of entity names, requiring robust methods to disambiguate names to canonical entities registered in a knowledge base. Additionally, in this dynamic world, new entities are constantly emerging, and disambiguation methods need to cope with the resulting incompleteness of knowledge bases.

This dissertation develops methods to discover and disambiguate named entities, thus linking texts to knowledge bases. The first contribution is a robust disambiguation method using a graph algorithm that makes use of the coherence among entities in the input. The second contribution is a novel model to compute the coherence among entities that works especially well for lesser known entities and is applicable to newly emerging entities. The third contribution addresses the discovery of emerging entities by modeling the entities not present in the knowledge base in an explicit manner. Finally, two applications using the developed entity disambiguation methods are presented.

Kurzfassung

Die Erkennung von Entitäten wie Personen, Organisation, Liedern oder Orten in Texten ist ein wichtiger Baustein für semantische Suche, maschinelle Übersetzung und Informationsextraktion. Ein Kernproblem der Erkennung ist die Mehrdeutigkeit aller Eigennamen. Diese erfordert robuste Methoden, um Eigennamen mit den passenden kanonischen Entitäten einer Wissensbasis zu verknüpfen. Zusätzlich müssen Verknüpfungsmethoden in dieser dynamischen, sich stetig wandelnden Welt von unvollständigen Wissensbasen ausgehen, da ständig neue Entitäten entstehen.

Diese Dissertation entwickelt Methoden, Eigennamen zu erkennen und mit kanonischen Entitäten zu verknüpfen, und verbindet somit Texte mit Wissensbasen. Der erste Beitrag ist eine robuste Methode zur Verknüpfung von Eigennamen mit Entitäten, die auf einem Graphalgorithmus basiert und sich die Koheränz zwischen Entitäten im Text zu Nutze macht. Der zweite Beitrag ist ein neues Modell, diese Kohärenz zu berechnen, das besonders gut für weniger bekannte und neu entstehende Entitäten funktioniert. Der dritte Beitrag adressiert spezifisch die Erkennung solcher neu entstehenden Entitäten, indem Entitäten, die nicht in der Wissensbasis vorhanden sind, explizit modelliert werden. Der letzte Beitrag besteht aus zwei Anwendungen, welche die in dieser Arbeit entwickelten Methoden zur Erkennung und Verknüpfung von Entitäten als Bestandteil verwenden.

Summary

Identifying all occurrences of entities such as people, organizations, cities, and songs in natural language texts is a fundamental task in natural language understanding. Ambiguous names are disambiguated by linking them to the corresponding entities registered in a knowledge base. Such knowledge bases are often derived from Wikipedia, where each article about a person or organization becomes an entity in the knowledge base. Entities are a valuable asset in semantic search, and a prerequisite for machine translation and information extraction tasks. The key challenge of entity disambiguation is the ambiguity of names and the incompleteness of knowledge bases. This dissertation makes the following contributions for advancing the discovery and disambiguation of entities.

Entity disambiguation resolves the ambiguity of names by linking them to entities in a knowledge base. State-of-the-art approaches combine various features such as name-entity context overlap and the semantic relatedness between entities to disambiguate input texts. However, existing approaches are not robust enough with respect do different input texts, or use compute-intensive inference algorithms for the actual disambiguation. Our disambiguation method, called AIDA, improves robustness issues by judiciously selecting the features to use based on the given input text. A fast approximation algorithm for dense subgraphs is used for the inference, combining high disambiguation quality with low inference complexity. To evaluate the quality of our approach, a large news corpus consisting of 1,393 documents and 34,956 entity names was annotated with all correct entities. On this corpus, AIDA's accuracy of 82% significantly outperforms state-of-the-art baselines which reach only 77%.

Keyphrase-based semantic relatedness is a key feature for entity disambiguation, capturing the notion of semantic coherence of entities in an input text. Existing methods of computing the pair-wise relatedness between entities make use of Wikipedia's link structure, achieving good performance for highly interlinked entities that are derived from Wikipedia. However, for less popular entities in Wikipedia with few links, or even entities that are not present in Wikipedia at all, such measures do not perform well. This dissertation presents KORE, for keyphrase-overlap relatedness, which uses salient keyphrases associated with the entities as a basis for measuring the semantic relatedness. Such keyphrases are easily gathered from Wikipedia, but also from other sources such as personal or company homepages. The higher computational cost of a keyphrase-based measure is addressed by a two-stage hashing scheme that efficiently and effectively pre-clusters entities. KORE improves the quality of entity disambiguation for long-tail entities, and enables the discovery of previously unknown entities.

Emerging entity discovery is essential for dealing with incomplete knowledge bases. Wikipedia-derived knowledge bases are incomplete because of two reasons: First, not everything can be added to Wikipedia because of guidelines enforcing a certain minimum popularity of entities; The second reason for incompleteness is the dynamic nature of the

world, where unknown persons suddenly become famous or new songs are written. All disambiguation methods have to deal with this incompleteness. Most state-of-the-art approaches use a threshold on some form of disambiguation score, discarding entities for which the algorithm cannot find enough evidence. This dissertation presents an emerging entity discovery algorithm that explicitly models unknown entities instead. The key insight is that one can construct a placeholder entity contrasting two representation models of any ambiguous name, the global model and the in-knowledge-base model. The in-knowledge base model is simple to construct, as for each name all candidates in the knowledge base are known. To construct the placeholder entity, it is subtracted from the global model, constructed from occurrences on Web or news pages. The integration of this placeholder entity in existing entity disambiguation methods is evaluated on a manually annotated corpus of 300 documents comprising 9,976 entity names. The method significantly improves the precision of discovering names whose entity is not present in the knowledge base.

Finally, two applications based on the developed entity disambiguation methods are presented.

Zusammenfassung

Ein fundamentaler Bestandteil der automatischen Sprachverarbeitung ist es, in Texten Entitäten wie Personen, Organisationen, Städte oder Lieder zu erkennen. Die Mehrdeutigkeit von Namen wird durch die Verknüpfung der Namen mit den passenden eindeutigen Entitäten in einer Wissensbasis aufgelöst, der Name wird disambiguiert. Wissensbasen erstellt man häufig mit Hilfe von Wikipedia, indem jeder Eintrag über eine Person oder Organisation eine Entität der Wissensbasis wird. Entitäten selbst sind eine wichtige Grundlage für die semantische Suche und eine Voraussetzung für die maschinelle Übersetzung oder Informationsextraktion. Das Kernproblem der Erkennung von Entitäten in Texten besteht in der Mehrdeutigkeit von Namen und in der Unvollständigkeit von Wissensbasen. Diese Dissertation trägt neue Modelle und Methoden zur Erkennung und Disambiguierung von Entitäten bei.

Die Disambiguierung von Entitäten löst mehrdeutige Namen auf, indem diese mit Entitäten einer Wissensbasis verknüpft werden. Methoden des aktuellen Standes der Technik setzen zur Disambiguierung verschiedene Merkmale wie die Ähnlichkeit des Kontextes von Name und Entität sowie die semantische Verwandtschaft zwischen Entitäten ein. Solche Methoden sind jedoch bei stark unterschiedlichen Eingabetexten nicht robust genug oder verwenden für die eigentliche Disambiguierung sehr rechenintensive Algorithmen. Unsere Disambiguierungsmethode, AIDA, wählt zu nutzende Merkmale abhängig vom Eingabetext aus und erhöht so die Robustheit. Weiter setzt AIDA zur Disambiguierung einen Graph-Algorithmus ein, der hohe Ausgabequalität mit geringer Laufzeit kombiniert. Die Qualität der AIDA Methode wurde auf einem großen Korpus von Nachrichtentexten mit 1.393 Dokumenten und 34.956 Namen evaluiert. Auf diesem Korpus erreicht AIDA mit 82% korrekt verknüpften Namen eine deutliche Verbesserung gegenüber dem Stand der Technik, der nur 77% der Namen korrekt zuordnet.

Die Phrasen-basierte semantische Verwandtschaft ist ein Schlüsselmerkmal für die Disambiguierung von Entitäten, da sie die semantische Kohärenz von Entitäten in einem Text abbilden kann. Bestehende Methoden berechnen die paarweise Verwandtschaft zwischen Entitäten mittels der Link-Struktur von Wikipedia, und erreichen so für Entitäten aus stark verlinkten Wissensbasen, die auf Wikipedia basieren, eine gute Qualität. Bei weniger populären Entitäten aus Wikipedia, die kaum verlinkt sind, oder sogar Entitäten, die gar nicht in Wikipedia vorhanden sind, erzielen solche Methoden allerdings keine guten Ergebnisse. Diese Dissertation präsentiert KORE, ein Modell zur Berechnung der semantischen Verwandtschaft, die nur mit den Entitäten assoziierte Phrasen verwendet. Der Vorteil dabei besteht darin, dass Phrasen nicht nur einfach aus Wikipedia abgeleitet werden können, sondern auch von persönlichen Homepages oder Firmenwebseiten. Dem höheren Berechnungsaufwand gegenüber den Maßen, die auf der Link-Struktur basieren, wirkt ein zweistufiges Hash-Verfahren entgegen, das Entitäten effizient und effektiv vorgruppiert. KORE verbessert die Qualität der Disambiguierung für weniger populäre Entitäten und ebnet den Weg für die Erkennung bisher unbekannter Entitäten.

Die Erkennung von neu entstehenden Entitäten ist für die Arbeit mit unvollständigen Wissensbasen essentiell. Wissensbasen, die auf Wikipedia basieren, sind aus zwei Gründen unvollständig. Zum einen gibt Wikipedia als Richtlinie ein gewisses Mindestmaß an Popularität vor, ohne die eine Entität nicht in Wikipedia aufgenommen wird. Der zweite Grund ist schlicht die Tatsache, dass die Welt, die täglich neue Berühmtheiten und neue Lieder hervorbringt, sich ständig ändert. Methoden zur Disambiguierung von Entitäten müssen mit dieser Unvollständigkeit umgehen. Im Stand der Technik werden hier häufig Schwellwerte auf den Bewertungsskalen der Disambiguierungsalgorithmen verwendet, also Entitäten mit zu geringer Bewertung fallen gelassen. Diese Dissertation stellt einen Algorithmus vor, der zur Erkennung von neu entstehenden Entitäten bisher unbekannte Entitäten explizit modelliert. Die Kern-idee hierbei ist es, eine Platzhalter-Entität zu konstruieren, indem für einen gegebenen, mehrdeutigen Namen ein globales Modell mit einem Modell der Wissensbasis kontrastiert wird. Das Modell der Wissensbasis ist leicht zu erstellen, da für jeden Namen alle möglichen Entitäten innerhalb der Wissensbasis bekannt sind. Das globale Modell hingegen wird aus Webseiten oder Nachrichtenartikeln, die den Namen enthalten, generiert. Die Platzhalter-Entität wird nun konstruiert, indem das Modell der Wissensbasis vom globalen Modell subtrahiert wird. Die Integration der Platzhalter-Entität in existierende Disambiguierungsmethoden wurde auf einem manuell annotierten Korpus von 300 Nachrichtentexten mit insgesamt 9.976 Namen evaluiert. Hier verbessert die neue Methode die Präzision der Erkennung von Namen, deren korrekte Entität nicht in der Wissensbasis vorhanden ist, deutlich.

Den Abschluss der Dissertation bilden zwei Anwendungen, die auf den entwickelten Methoden zur Disambiguierung von Entitäten aufbauen.

Acknowledgments

I want to thank my supervisor Gerhard Weikum for his great guidance throughout my doctoral studies. Without his insights and support this work would have not been possible. I especially enjoyed the freedom he gave me to pursue so many projects during my PhD. I am grateful to the Max Planck Institute for Informatics for creating a superb working environment in which few if any obstacles hindered me from focusing on the work at hand, and to all contributers of the AIDA project, most notably Mohamed Amir Yosef. My colleagues at the Databases and Information Systems group provided much needed encouragment when the work at hand was not going as smoothly as I wished.Daniel Bär and Simon Wiesmann gave me a lot of support in the more intense periods of the PhD, thank you for never closing my chat window. I also want to thank my parents, Doris and Josef Hoffart, who always supported me in my decisions and continue to do so. Last but not least, I want to thank Stephan Seufert and Niklas Jakob for the many hours spent encouraging me to start my PhD at Max Planck Institute for Informatics, it has been the best decision in my professional life so far.

Most importantly, thank you Oksana Neb for sharing your life with me — before and more importantly during my PhD.

Contents

1. Introduction

The question of whether machines can be intelligent has been famously tied to the test of whether a machine can behave indistinguishably from a real human being in a conversation. This so called "Turing Test", named after its inventor and computer science founding-father Alan Turing, was introduced as early as 1950 [Tur50], in a time when only very few computers even existed. The idea behind this test is simple: the intelligence of human beings is hard to define, but it is clear that intelligent behavior can be judged by other human beings, as the cognitive processes of communicating clearly demarcates humans from other species. By defining a test for intelligence on the observed behavior, in the form of text produced by it, Turing avoids the problem of defining intelligence directly. As a consequence, intelligent behavior has been strongly coupled with understanding and producing natural language by machines, which is still, more than sixty years later, one of the great and unsolved goals in computer science.

1.1. Motivation

While the goal of fully understanding natural language spoken and written by humans is still out of reach, there have been significant recent advances on "machine reading" at a coarse level: computers are beginning to understand single words or phrases in texts by explicitly representing their meaning. This is the first and fundamental step towards a full understanding of any text: once the actual senses of all words in a sentence are known, all speakers and concepts identified, this opens the way to understanding how they relate to each other. Let's make this notion concrete with the following example sentence:

> The opener on Dylan's 1976 record Desire is a song about the black fighter Carter. It ends with a tribute to his wife Sara.

Example 1.1

Looking at the single words, it becomes clear why this seemingly simple task is difficult: almost all of the words are ambiguous, something that a human reader barely notices at first glance, as one resolves the ambiguity subconsciously most of the time. In general, the noun "opener" can refer to a tool for opening cans, or something that is done first at an event, as it is used here. "Record" can mean an extremely good performance in some sports discipline, or a sound recording. Not only the nouns, but even more so the names are ambiguous. "Dylan" can refer to `Dylan Thomas`, the `Dylan programming language`, `Bob Dylan`, or his 1973 self-titled album. Printing all potential persons named "Sara" would quickly exhaust the ink in everyone's printer. Once all words and names

are understood, the next step is understanding what relations they form between them, e. g. that `Bob Dylan created Desire` in the year `1976`. However, **the key problem addressed in this work is the ambiguity of the single names**, asking the question which people, or, more generally, entities, are meant by "Dylan", "Carter", and "Sara"?

The ability to understand single words of a text was made possible by comprehensive lexical resources that associate words and phrases with their *senses*. A collection of senses for all nouns, verbs, and adjectives is available in the form of WordNet[1] [Fel98], and the research on word sense disambiguation [Nav09], has benefited enormously from WordNet.

More recently, Wikipedia[2], the online-encyclopedia anyone can edit, has opened the field for the disambiguation of names. As of the time of writing, Wikipedia contains articles about more than 4 million things, ranging from general *concepts* like "cat" or "peace" to a large number of persons and locations, but also organizations, events, songs, books, ..., which we refer to as *named entities* or simply *entities*. While primarily created for humans to look up information about a particular person or concept, each article can also be seen as one unit in a sense inventory which can be referred to by a program that resolves ambiguous names automatically. Resolving all words in a text to their corresponding Wikipedia articles is often called *Wikification*, introduced by Mihalcea and Csomai [MC07]. The name "Wikification" comes from the goal of reproducing the way human editors interlink articles in the Wikipedia. However, the mix of general concepts and individual entities that comes with Wikification is not always desirable, and machine readable knowledge bases derived from Wikipedia strive to make a distinction between these two. Such knowledge bases store entities alongside additional knowledge, e. g. that `Bob Dylan is a singer` or that `Dylan Thomas is a writer`, and that `Bob Dylan created Desire`. Bunescu and Pasca [BP06] were the first to consider only entities as disambiguation target, calling the task *named entity disambiguation*, or *NED* for short. Here, given an input text, the goal is to detect any phrase referring to an entity and disambiguate it correctly. Each such phrase is called a *mention*. An important aspect of this task is to determine if the correct entity is contained in the knowledge base. If not, it should be linked to a placeholder entity for *out-of-knowledge-base entities*, abbreviated *OOE*.

The most straightforward way to perform disambiguation is to simply assign the most likely entity to a mention, based on some notion of *prior probability* or importance for an entity given a mention, derived for example from the length of the Wikipedia article (the longer the more important) or from Wikipedia's link structure (how often does a name refer to a particular entity). Such an approach often works well, but is blind to the context surrounding a mention. Take the mention "Carter" from the above example sentence; just looking at "Carter", without additional context, the most likely entity that comes to mind is most probably `Jimmy Carter`.

A good disambiguation method thus needs to take into account the *context* of a mention in the input text, e. g. "black fighter" in the example sentence, which is a very

[1] `http://wordnet.princeton.edu`
[2] `http://wikipedia.org`

good clue for `Rubin Carter`. This mention context can be compared to the descriptive context of all possible entities that can be referred to by this mention. The entity context is either created from the knowledge base directly or derived from a textual description of the entity, e. g. by extracting all salient and descriptive phrases, or *keyphrases*, from the corresponding Wikipedia article.

Another key feature for identifying the correct entity for a mention is that a text is usually coherent in its topic, it might be a review of a single album or a news article covering a recent event, and thus all resolved entities should also be coherent. Ensuring *coherence* is possible when all mentions are *jointly disambiguated*, instead of one after the other. The notion of coherence among entities is usually captured by a measure of *semantic relatedness* between pairs of entities, which scores highly entity-entity pairs that are strongly related, e. g. `Bob Dylan` and his album `Desire`.

This line of research has seen numerous approaches since the initial publications, see the detailed discussion of related work given in Section 2.2.2. However, some challenges still remain.

1.2. Challenges

C1. Brittle disambiguation quality Existing methods, like the one by Kulkarni et al. [KSRC09], already employ very powerful algorithms to jointly disambiguate entities for all mentions in an input document, based on the features described above: prior, context, and coherence. However, their quality still varies a lot from document to document, as not all of the features are useful for each kind of document. Especially the notion of coherence can be very misleading, if there are multiple groups of coherent entities that could potentially make sense for a document. This might sound unlikely, however it happens frequently in practice. Even for the above example about Dylan and the album Desire, there are multiple coherent entity groups for the mentions "Dylan" and "Desire". The already elaborated `Bob Dylan` and his album `Desire`, but also the DC comic character `Desire` and the writer and graphical artist `Dylan Horrocks` working at the publisher of these comics.

C2. Limited ground truth data for evaluation Evaluating the quality of a disambiguation method depends on the availability of extensive and clean ground truth data, where human annotators mark every mention with the correct entity. Existing ground truth data is often taken from Wikipedia, where each link can be viewed as a disambiguated mention, which narrows the applicability to other domains. For other domains, e. g. news or news-wire, only small samples of documents with few annotations are available.

C3. Efficiency bottleneck in high-quality disambiguation Many state of the art disambiguation methods use a measure of semantic relatedness between entities to determine the best candidate. This measure is often computed pairwise between all entity candidates for a given input text, leading to a quadratic complexity in the number of candidates before the actual algorithm runs.

C4. Incomplete knowledge Our world is highly dynamic; so no knowledge base will ever be complete. New songs are composed every day, new movies are released, new companies are founded, there are new weddings, sports matches, and disasters and formerly unimportant people become noteworthy. Even if these entities make their way into Wikipedia or another knowledge base eventually, there is often a significant delay between the event taking place and Wikipedia editors adding the entities [KGC13]. Existing disambiguation algorithms do not deal with these new entities in a robust manner. Furthermore, existing entities change, they get new jobs, marry someone, get their PhD degrees. Keeping track of these changes and thus being able to disambiguate known entities in previously unrelated contexts is another problem where existing methods fall short.

C5. Strong focus on Wikipedia Almost all current work on entity disambiguation takes Wikipedia as repository of potential entities to disambiguate to. While this is a perfectly valid choice, the narrow focus leads to the use of features that are strongly tied to Wikipedia and are non-existent in other knowledge bases. Due to the problem of incompleteness mentioned above, it is necessary to take entity disambiguation beyond Wikipedia.

1.3. Contributions

This work addresses the challenges outlined above, developing new methods to advance the state of the art:

AIDA: Robust disambiguation of named entities in text AIDA, our accurate disambiguation method for named entities in text, **improves the quality** (**C1**) over the existing state of the art of entity disambiguation employing a robust joint inference algorithm. The improvement is made possible by the judicious combination of three feature classes in a single framework: the context-independent prior probability of an entity given a mention, the overlap between mention context and entity context where keyphrases are used for entities, and the semantic relatedness between entities based on the Wikipedia link structure. All features are used to create a graph representation of the disambiguation problem, which is in turn solved with a fast greedy algorithm. Additional robustness is achieved by selectively enabling only some of the features to better cope with the variety of input texts.

Another key contribution of this work is the **creation of a large, manually labeled ground truth data set** (**C2**) to evaluate existing methods alongside ours, comprising 1,393 documents and nearly 35,000 mentions. The annotations are publicly available, and a number of follow-up works have relied on this data, including [CFC13, HNR14]. AIDA results were presented at EMNLP 2011 [HYB+11] and as a demo at VLDB 2011 [YHB+11].

KORE: Efficient and general-purpose relatedness measure The most often used measure for computing the relatedness between entities is based on co-occurrence of entities in Wikipedia, introduced by Milne and Witten [MW08a]. This measure allows a high-quality estimation of the semantic relatedness between entities. However, it crucially depends on the availability of explicit entity markup, something that for the time being can only be found for Wikipedia entities.

Our measure, dubbed KORE for **K**eyphrase **O**verlap **RE**latedness, opens up **semantic relatedness for emerging, out-of-Wikipedia entities** (**C5**). It achieves this by using keyphrases associated with each entity, estimating the semantic relatedness as the strength of overlap among these keyphrases. Keyphrases are easily available for entities in Wikipedia, but can also be harvested for non-Wikipedia entities. For example, keyphrases for researchers can be found on their personal homepages, keyphrases for small bands or not-so-popular songs can be found on social Websites like `last.fm`.

The semantic relatedness computation using keyphrases is computationally more expensive than a simple co-occurrence based measure. We thus devised a hashing scheme that efficiently partitions entities into sets of highly related ones in linear time, achieving a **significant reduction in computation time** (**C3**) over the naive pairwise semantic relatedness computation. The hashing scheme is also applicable to other methods of semantic relatedness computation. The work on KORE was presented at CIKM 2012 [HSN+12].

NED-EE: Discovering emerging entities While Wikipedia is a great source of knowledge on a large number of topics and entities, it will never be complete: first of all, Wikipedia has strict guidelines as to the relevance of an entity, and a person or event has to reach a certain level of prominence before it can be entered into Wikipedia. Second, and more importantly, the world changes very quickly, and although Wikipedia is fast, it still lags behind new events, sometimes only for a few hours, but sometimes for weeks [KGC13].

Our extension of entity disambiguation methods is able to **discover emerging entities that are not yet in Wikipedia** (**C4**) by explicitly modeling such emerging entities with keyphrases harvested from a stream of news articles, in contrast to existing methods that discover such entities based solely on the absence of indication for existing entities. Additionally, we devised a robust way to measure confidence in the disambiguation of entities, allowing our methods to **harvest additional keyphrases for entities that are already part of Wikipedia** (**C4**). This is especially important as the same lag for the creation of new articles also holds for the updating of existing articles, where new events surrounding an entity sometimes take weeks to be added to an article. The results were presented at WWW 2014 [HAW14].

1.4. Organization

The remainder of this dissertation is organized as follows. **Chapter 2** first details the tasks of entity recognition and entity disambiguation and discusses the state of the art.

This is followed by an overview of the foundations in the fields of natural language understanding and knowledge bases. The chapter concludes by presenting a wider scope of related work. **Chapter 3** presents the work on entity disambiguation, with a focus on robustly dealing with different inputs. **Chapter 4** introduces a measure for computing entity relatedness based on keyphrases, which is especially suitable for long-tail or out-of-knowledge base entities in the context of entity disambiguation. **Chapter 5** adds a model for unknown entities to the disambiguation process, enabling the entity disambiguation to cope with out-of-knowledge base entities and creating the foundation for enriching the underlying knowledge base. **Chapter 6** showcases two applications built with the previously introduced methods, namely an entity-centric search and an entity-based analytics system. **Chapter 7** gives conclusions and possible directions for future work.

2. Background & Related Work

This chapter explains the tasks of entity recognition and entity disambiguation and discusses the state of the art. It then gives an overview of the foundations, both in the fields of natural language understanding and knowledge bases. It concludes by placing the work of entity disambiguation in the context of a wider scope of related work.

2.1. Recognizing Entities in Text

2.1.1. Task

The first step in discovering entities in a text is the recognition of single-word or multi-word phrases such that the phrase denotes an individual entity in the real world, without knowing which entity. The task of recognizing these entity mentions has received a lot of attention over the past two decades under the name of named entity recognition, NER for short.

Examples are names of persons, organizations, locations, or works of art in text. To give an example, the output of doing named entity recognition for the sentence in Example 1.1 is:

"The opener on Dylan 's 1976 record Desire is a song about the black fighter Carter. It ends with a tribute to his wife Sara."

Often, the strict requirement to refer to an individual entity is relaxed, allowing for other phrases that are used in a similar manner to be recognized as well. This is due to the observation that the recognition of these phrases is useful in downstream tasks like entity disambiguation or relation extraction. These phrases are:

- **Product names**, e. g. "PlayStation". Even though the product name does not refer to a unique individual entity, but rather to a set of entities (all actual PlayStation systems ever built), product names are useful from an information extraction point of view. The core issue here is that it is not always clear where to draw the line between what is an *entity* and what is a *class*. In the "PlayStation" example, one could define each individual PlayStation connected to a TV in someones living room, each with it's own serial number, as the actual entity. However, these individual systems are actually clones of production lines of a specification of the system, numbering in the thousands. These production lines are then usually grouped under a common name for marketing, such as "PlayStation 4". While not really an individual entity, for many applications dealing with Web or news data, this is the right level of granularity.

- **Terminology of specific domains** such as medicine or biology, e. g. disease names like "influenza", "diabetes", or "stroke". The same argument of usefulness as for the product names holds here.

2.1.2. Methodology

Most state-of-the-art methods for NER use machine learning methods to label the input texts. The input data used for training has been mostly created in benchmarking challenges, so-called shared tasks, the main ones being MUC (Message Understanding Conference) [GS96], where the task has been originally introduced, and CoNLL (Computational Natural Language Learning) [SDM03]. Nadeau and Sekine [NS07] give an overview of the development of the field, also detailing all shared tasks that drove the field forward by providing labeled data. The models used for training differ; one of the most widely known NER systems, the Stanford NER [FGM05] models the task using conditional random fields.

The machine learning methods are applicable for all languages for which labeled data exists, which is the case for a large number: English, Chinese, Spanish, Arabic, German, and French, just to name a few.

Labeled data by both MUC and CoNLL is available in the form of news-wire texts, with very clear language and frequent use of named entities, and the state-of-the-art methods achieve more than 90% in terms of F1 on texts of these types. Models trained on these texts do not perform so well in other domains, due to the great variety in texts ranging from highly formalized clinical texts to microblog posts, resulting in a significant drop in performance [NS07]. This resulted in additional research in the field, focusing especially on the rapid growth of user generated content on the Web in the form of forums, blogs, and tweets [LZWZ11].

2.2. Entity Disambiguation and Discovery

2.2.1. Task

Named entity disambiguation, or NED for short, is the task of linking ambiguous names in texts to canonical entities. NED usually assumes that the input text is first processed by a method for named entity recognition (see Section 2.1). We refer to the phrases recognized by this method as *mentions*. The goal of NED then is to link the set M of mentions in the input text to entities in a knowledge base KB (see Section 2.3), or determine that a mention refers to an entity that is not in the KB, an out-of-KB entity OOE. The KB consists of the entity repository E containing all entities, as well as a dictionary $D \subset (N \times E)$ of pairs of names $\in N$ and entities $\in E$. For each mention $m \in M$, the dictionary D provides a set of candidate entities E_m for m whose name(s) $n \in N$ matches m for some lookup function, e. g. a full string match. If D does not give any candidates for mention m, E_m is empty and the mention is trivially regarded as pointing to an OOE. Additionally, the KB also contains features F_e describing each

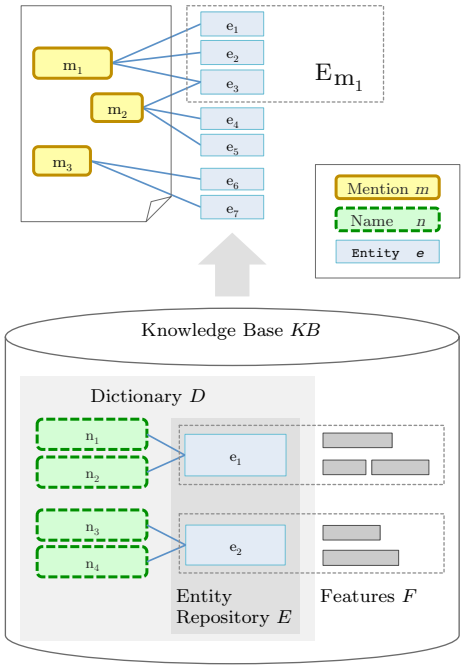

Figure 2.1.: Entity Disambiguation

entity e in the repository as a basis for the disambiguation. F_e differs from method to method, and will be introduced whenever a method is mentioned.

The overview of all components is given in Figure 2.1. An example instance of a disambiguation problem is depicted in Figure 2.2 on page 12, where the set of mentions M in a text are connected to candidate entities $E_d \subseteq E$ by means of a lookup of $m \in M$ in the dictionary D.

2.2.2. Related Work

The following section presents the most important related work in the field of NED in historical order, followed by a summary of works focusing on different aspects. An overview of the works discussed here is given in two tables: Table 2.1 compares the basic setup of the systems, Table 2.2 details the features used for disambiguation.

Initial work on disambiguating named entities was limited to special domains; for example Woodruff and Plaunt disambiguate place names to a list of 80,000 location entities [WP94].

Bunescu and Pasca [BP06] were the first to realize the potential of using Wikipedia for NED, proposing an approach to disambiguate proper names to Wikipedia articles. Interestingly, they already restrict the Wikipedia articles to the subset corresponding to named entities instead of including all of Wikipedia's articles about common concepts. The key feature that is exploited is the similarity of a Wikipedia article and the context surrounding the mention to be disambiguated, improved by a word-category association

Table 2.1.: Overview of the basic setup of key NED methods

Method	Input	Target Concepts	Dictionary	Evaluation Data
Bunescu & Pasca [BP06]	single queries	entities (subset of WP* by heuristic filter)	titles, redirects, disambiguation pages	WP links
Cucerzan [Cuc07]	documents	WP (mention recognition is only named entities)	titles, redirects, disambiguation pages, link anchors	WP docs; 20 news articles with 756 mentions
Mihalcea & Csomai [MC07]	documents	WP	link anchors	WP docs; user study comparing editor links vs. automated links; impact on reading task
Milne & Witten [MW08b]	documents	WP	link anchors	WP docs; 50 news articles
Kulkarni et al. [KSRC09]	documents	WP	titles, redirects, disambiguation pages, link anchors	100 Web and news documents
Ratinov et al. [RRDA11]	documents	WP	link anchors	WP docs; 130 news articles
Ferragina & Scaiella [FS12]	(short) documents	WP	titles, redirects, link anchors	short WP paragraphs
Guo et al. [GCK13]	tweets	WP	link anchors	ca. 1,200 tweets
Jin et al. [JKWL14]	documents	social network profiles (long tail persons)	-	555 Web documents
Li et al. [LWH+13]	single query	WP	link anchors	TAC, Tweets
This thesis	documents	entities (YAGO, subset of WP, see Sec. 2.3.3)	titles, redirects, disambiguation pages, link anchors	1,393 + 300 news-wire articles, Wikipedia sentences

* Wikipedia

Table 2.2.: Feature overview of key NED methods

Method	Recognition	Features	Joint Inference	Unknown Entities
Bunescu & Pasca [BP06]	-	tokens in word window and WP* article; correlation tokens w/ entity categories	-	threshold
Cucerzan [Cuc07]	Own NER	link anchors, categories, list pages	simulated by iterative disambiguation w/ context expansion	When not in dictionary
Mihalcea & Csomai [MC07]	phrase extraction for WP style links	classifier trained on words and pos-tags	-	-
Milne & Witten [MW08b]	WP-link trained classifier	prior probability	simulated using all-candidates relatedness**	-
Kulkarni et al. [KSRC09]	-	prior, keyword-context similarity	ILP solver w/ MW**relatedness	threshold
Ratinov et al. [RRDA11]	NER + NP chunks	prior, importance, top-article-tokens, top-inlink-context-tokens	simulated by all-candidates relatedness (inlink-outlink-overlap)	classifier on disambiguation-confidence and other features
Ferragina & Scaiella [FS12]	dictionary + heuristic for phrase length	prior	simulated by all-candidates relatedness (MW)	threshold
Guo et al. [GCK13]	joint NERD with filters	WP page view count, type, top keywords from WP article + additional	approximated by all-candidates relatedness (Jaccard on inlinks)	threshold
Jin et al. [JKWL14]	-	textual phrases	-	placeholder w/ random features of In-KB entities
Li et al. [LWH+13]	-	topic-context similarity	-	threshold
This thesis	Stanford NER	prior, keyphrase-overlap similarity	Graph algorithm w/ MW or keyphrase relatedness	explicit model of unknown entity

* Wikipedia ** Milne & Witten relatedness measure, see Equation 3.7

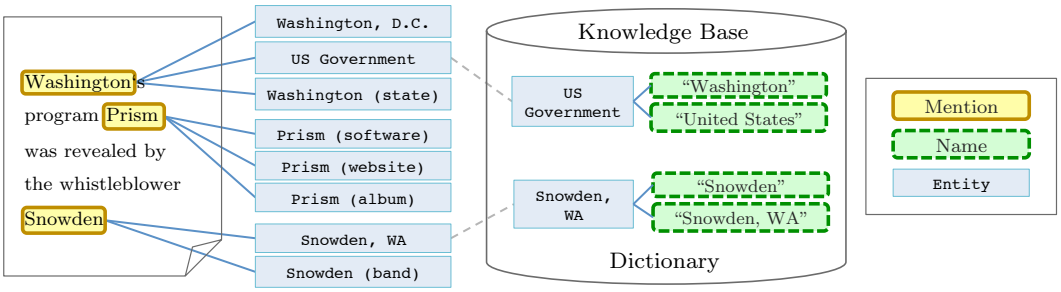

Figure 2.2.: Entity Disambiguation example

to counter context-sparseness issues. Additionally, other foundations re-used by most of the follow-up work are laid, namely the creation of the name-entity dictionary by means of redirects and disambiguation pages and the thresholding on the final score for the best entity to determine if it should be regarded as out-of-KB. As the seminal work in the field, the evaluation is restricted to using Wikipedia links as ground truth, thereby also focusing on the disambiguation task and ignoring the NER aspect.

Cucerzan [Cuc07] presents the first full-fledged system, which takes full documents as input (as opposed to the link-in-context queries of Bunescu and Pasca), recognizing named entities with a statistical recognition method, and finally disambiguating them to entities in Wikipedia. The Wikipedia articles are not restricted to entities like Bunescu and Pasca do it, but the recognition method is for named entities only, which essentially limits the disambiguation phase by means of a restricted mention set to entity candidates. Cucerzan recognizes the potential of doing joint disambiguation for all mentions at once, however due to reasons of computational complexity approximates this by including the entity categories as parts of the context similarity vector. The evaluation is done on a small corpus of manually annotated news articles, in addition to Wikipedia.

Mihalcea and Csomai [MC07] coined the term Wikification for the task where the entity repository is the full Wikipedia. Additionally, even the recognition is now focused on reproducing Wikipedia-like links, including non-named entities. A user study where existing Wikipedia links were compared against the automatically generated ones found a very good annotation quality, producing links that are hardly distinguishable from human-generated ones. While the disambiguation methods are very basic, namely a classifier trained on a small window around existing Wikipedia links, this work is the first one to introduce a most-frequent sense baseline, which has been used as a strong feature for disambiguation methods by subsequent works.

Milne and Witten [MW08b] tackle the same task as Csomai and Mihalcea, trying to create Wikipedia-style links. Two key ideas stand out: The first one is incorporating the disambiguation confidence of phrases into the decision of whether a phrase should be used as mention, switching the more intuitive order of recognize-then-disambiguate. The second idea is capturing the semantic relatedness between disambiguation candidates to determine the correct entity. This feature has been used by a significant fraction of important follow-up work.

Kulkarni et al. [KSRC09] improve on the two previous Wikification systems by implementing a real joint inference. The actual algorithm method starts with a supervised learner for a similarity prior, and models the pair-wise coherence of entity candidates as a probabilistic factor graph with all pairs as factors, where the same entity-entity relatedness measure as above by [MW08b] is used. The MAP (maximum a posteriori) estimator for the joint probability distribution of all mappings is shown to be an NP-hard optimization problem, so that [KSRC09] resorts to approximations and heuristics like relaxing an integer linear program (ILP) into an LP with subsequent rounding or hill-climbing techniques. However, even approximate solving of the optimization model has high computational costs.

Ratinov et al. [RRDA11] address the global coherence of mentions in a document in an iterative manner. First, all mentions are disambiguated independently, using the outcome as input for an optimization step that takes into account the semantic relatedness between the entities — however still one-by-one. There is no direct comparison against the real joint-inference method of [KSRC09], but both works show improvements for the coherence-aware methods. Another key point addressed is the detection of out-of-KB entities OOE for mentions which should not be linked. Here, a classifier is trained on the disambiguation confidence and the link-likelihood of a phrase (derived from Wikipedia), among other features. While the classifier works in some cases, the authors state that it is very sensitive to domain and corpus changes.

2.2.3. Further Aspects of Entity Disambiguation

More recent work focuses on different aspects and sub-problems of NED, detailed in the following.

Domain and Genre

A large number of initial work on NED has addressed Wikipedia-style texts or news. More recently, short texts, especially Twitter and other micropost platforms, have received increasing attention.

The TagMe system by **Ferragina and Scaiella** [FS12] uses very light-weight features, combining only the prior probability of an entity given a mention with the semantic relatedness of all candidate entities. Using only entity relatedness and no other form of context overlap restricts the usability to short texts with a high density of mentions, long texts need to be processed chunk-by-chunk. The upside is the high performance and low memory footprint of the system, and the high quality on short texts.

Other work has specialized only on Twitter, improving the quality by focusing on the specifics of this micropost platform. In the short messages, special syntactic structures like #-Tags and @-Mentions, as well as abbreviated wording and slang are prevalent. This has a high impact especially on the correct recognition of mentions in the Tweets, and consequently this is what [MWdR12, GCK13] focus on. Twitter-specific entity disambiguation was also the goal of the challenge organized by the *Making Sense of Microposts* workshop at the World Wide Web 2014 conference [CRV+14].

Incomplete Knowledge Bases

Using person entities taken from a social networking site as entity repository, **Jin et al.** [JKWL14] create a method to link these mostly long-tail entities occurring on arbitrary Websites. By restricting the annotation to a single entity per document, together with a powerful method to select the most salient textual phrases from the input website, their system achieves a very high precision with good recall. To deal with the large fraction of Websites containing out-of-KB entities, an *unknown entity* is introduced. Based on the assumption that the knowledge base is representative of the unknown population as well, this unknown entity is represented by randomly sampling all existing ones for their features.

The work presented in Chapter 5 models the unknown entity by mining features from documents outside of the knowledge base. The main features used are keyphrases, so the approach extracts keyphrases for mentions from Web and news documents, taking them either as additional evidence for existing entities or, by subtracting all keyphrases for existing entities, as keyphrases for an unknown entity referred to by the given mention.

The aspect of incomplete knowledge bases in the form of missing keyphrases for existing entities is also addressed by **Li et al.** [LWH+13], who mine additional keyphrases for entities from the Web to counter sparseness and missing evidence. The mining itself is done using an incremental algorithm. All entity links in Wikipedia are taken as initial initial input and fixed ground truth. Then, additional keyphrases from the Web are mined so that the original Wikipedia assignments are kept. The mining phase is then repeated.

Semantic Relatedness

One of the key features for disambiguation is the semantic relatedness between two entities. For joint inference approaches, like Kulkarni [KSRC09] or ours detailed in Chapter 3, the use is obvious. However, semantic relatedness is also used as a strong feature by a large number of works that do mention-by-mention inference. Milne and Witten [MW08b] were the first to use it, but also Ferragina et al. [FS12] and Ratinov et al. [RRDA11] use it prominently. Almost all the works are using the semantic relatedness measure defined by Milne and Witten [MW08b].

While the high quality of the Milne and Witten measure has been shown repeatedly, recent works have addressed additional aspects of semantic relatedness measures. **Ceccarelli et al.** [CLO+13] use learning to rank methods to combine a large number of relatedness measures, ranging from very simple ones like the existence of a direct Wikipedia link between entities, over Jaccard similarity on the entity link sets up to Kullback-Leibler divergence and χ^2. This combination improves the quality of a number of NED systems relying on relatedness. Additionally, their work shows that when it comes to a single measure to use, Milne and Witten's measure [MW08a] is not the best one to use. While it provides good quality, a number of measures like Jaccard similarity and the conditional probability of finding a candidate entity given an entity work better. An interesting point to note is that all of them are based on the Wikipedia link graph.

The work on semantic relatedness presented in **Chapter 4** strives to find a good relatedness measure that does not depend on Wikipedia links. This opens up joint inference methods for NED on knowledge bases that are not interlinked as Wikipedia is, but simply a collection of entities described by textual phrases.

2.2.4. Benchmarking Challenges

The task of named entity disambiguation and discovery has been addressed in multiple challenges over the past years.

INEX Link-the-Wiki

The initial challenge has been held as part of the INEX in 2008 [HGT08], where the goal was to recreate Wikipedia links on a per-mention basis. An additional twist in this challenge was the goal of identifying the actual part in target Wikipedia article where the link should point to. This is in contrast to almost all other work on entity disambiguation, where the target is always the full article.

TAC KBP

In 2009, the entity linking at the TAC Knowledge Base Population workshop [MD09] was first held. The task was to decide for a single mention in a text the correct entity in a Wikipedia derived knowledge base, or that it is missing. While the focus on a single mention per text simplifies the creation of the corpus and the evaluation, it also makes the task less appealing for joint-inference methods, where all mentions in a text are deemed relevant. In the following years, the task was expanded to cluster the mentions marked missing from the knowledge base so that they all refer to the same thing. In 2014, the task is still ongoing, now for the first time with full documents as input. A number of TAC participants subsequently published their work at major NLP conferences [DMR+10, HS11], more references are given in [JGD11].

ERD Challenge

The Entity Recognition and Disambiguation challenge was held first in 2014 as a SIGIR workshop. The challenge was divided into two tasks, both of which had the goal of recognizing all mentions of entities in the input text and disambiguating them to the correct entity. The difference between the tasks was the type of the documents, which for the *short document* task was Web queries, for the *long document* task crawled Web documents. For each task, participants could provide web services that were called and evaluated by the organizers, no gold standard data for training or evaluation was provided. More details will be presented in a forthcoming overview paper [CCG+14].

2.3. Knowledge Bases

2.3.1. Overview

Knowledge bases are a prerequisite for entity disambiguation, without a knowledge base cataloging and describing the entities, there is nothing to disambiguate to. Comprehensive knowledge bases in machine-readable representations have been an elusive goal of AI for decades. Seminal projects such as Cyc [Len95] and WordNet [Fel98] manually compiled common sense and lexical (word-sense) knowledge, yielding high-quality repositories on intensional knowledge: general concepts, semantic classes, and relationships like hyponymy (subclass-of) and meronymy (part-of). These early forms of knowledge bases contain logical statements that songwriters are musicians, that musicians are humans and that they cannot be any other species, or that Canada is part of North America and belongs to the British Commonwealth.

However, early resources like the original Cyc and WordNet lacked extensional knowledge about individual entities of this world and their relationships (or had only very sparse coverage of such facts). They do not contain entities like `Bob Dylan` and `Jimmy Page`, nor the knowledge that both are musicians, that Dylan is born in Duluth, that Duluth is a city in Minnesota, or that both Dylan and Page have won the Grammy Award. These individual entities and their descriptions are the crucial ingredient for the task of named entity disambiguation.

In the last few years, the great success of Wikipedia and algorithmic advances in information extraction have revived interest in large-scale knowledge bases and enabled new approaches that could overcome the prior limitations of sparse entity coverage. Notable endeavors of this kind include DBpedia [ABK+07], KnowItAll [ECD+05, BCS+07], BabelNet [NP12], DeepDive [NZRS12], NELL [CBK+10], Omega [PHP08], WikiTaxonomy [PN09, PS11], and YAGO [SKW07, HSBW13], and meanwhile there are also commercial services such as `freebase.com`, a part of Google's Knowledge Graph. These contain many millions of individual entities, their mappings into semantic classes, and relationships between entities.

2.3.2. Data Model

These knowledge bases are most easily represented as a graph structure, such as the snippet shown in Figure 2.3 where facts about `Bob Dylan` are shown. These graphs are in turn most often stored as subject-property-object triples (SPO triples) according to the RDF data model, which provides convenient query interfaces based on languages like SPARQL.

2.3.3. YAGO

The knowledge base that underlies the work presented in this dissertation is YAGO [SKW07, HSBW13], which extracts facts from the semi-structured parts of Wikipedia, mostly the infoboxes and the categories. YAGO has been chosen as entity repository for the following reasons:

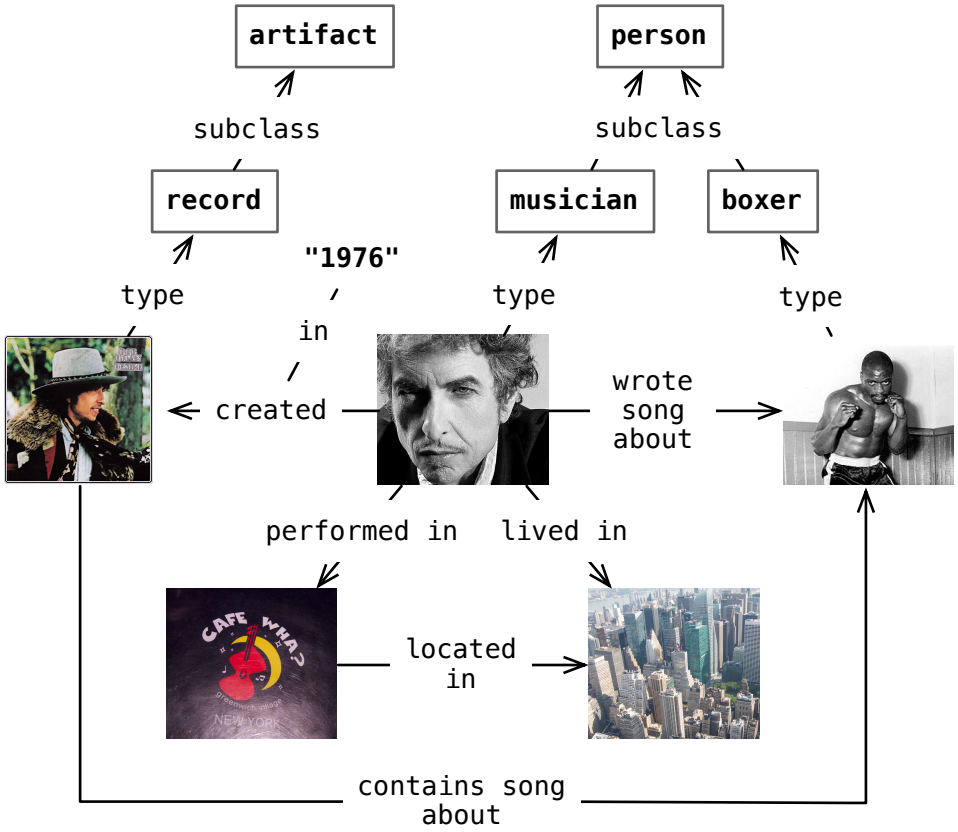

Figure 2.3.: Example Knowledge Graph (Bob Dylan photo [k])

- The clear distinction between individual entities and classes. As we are focusing on disambiguating named entities, we are interested only in the entities as disambiguation targets. In YAGO, every encyclopedic Wikipedia article becomes an entity unless WordNet contains the title as part of a synset.

- The use of WordNet as a taxonomic backbone, which enables the integration of other work, e. g. named entity classification (see Section 2.4.4).

- The one-to-one mapping between YAGO and Wikipedia. This is especially important for comparability to other, Wikipedia-based works.

- YAGO's extraction architecture is easily extensible for harvesting additional data from Wikipedia necessary for NED, e. g. descriptive keyphrases for entities or links between entities.

2.4. Related Tasks

The following section gives a brief overview of tasks related to named entity disambiguation, namely word sense disambiguation, wikification, coreference resolution, and named entity classification.

2.4.1. Word Sense Disambiguation

Before large repositories of individual entities became available, enabling the disambiguation of named entities, a large body of work had already been done in word sense disambiguation. This task is similar in nature to named entity disambiguation: the goal is to disambiguate all words (often excluding named entities) in an input text to their correct meanings. This requires a complete dictionary of senses for all words in a given language. One such resource for English is WordNet [Fel98], which represents nearly all senses of the English language as clearly defined synsets. Take as an example the following sentence.

> Dylan released a record in 1976.

Example 2.2

The meaning of "record" in this sentence is defined in WordNet by: "(noun) sound recording consisting of a disk with a continuous groove; used to reproduce music by rotating while a phonograph needle tracks in the groove" with the synonyms "disc", "platter", "disk", "record", "phonograph record", and "phonograph recording", whereas the correct sense for "release" is "(verb) prepare and issue for public distribution or sale; 'publish a magazine or newspaper' ".

While this task seems superficially very similar, the are some key differences:

- The ambiguity of named entities can be a lot higher — just take a common first name occurring like "Bob" as example, which can occur frequently in texts and has about 3,300 candidate entities in Wikipedia.
- In word sense disambiguation, all words in a text need to be considered, not just a few named entities, which requires different methodologies for automated disambiguation.

There are surveys covering details of word sense disambiguation by Navigli [Nav09] and McCarthy [McC09].

2.4.2. Wikification

The task of Wikification sits somewhere in between named entity disambiguation and full-blown word sense disambiguation, taking the pragmatic approach of disambiguating all words whose meaning has a corresponding Wikipedia article. The use of all of Wikipedia as repository of senses contributes the "Wiki" in Wikification. Recall that

Wikipedia articles cover both individual entities like `Bob Dylan`, but also concepts like `musician`, which are discarded in systems doing pure NED. However, as the individual entities in Wikipedia are a subset of all articles considered by Wikification, NED and Wikification systems can be compared by restricting the repository.

The benefits of doing Wikification are obvious: more words in a text are assigned with the meaning, which is both useful for downstream applications and helpful during the disambiguation process by naturally forming a larger context for joint inference methods. The downside is that the task definition is not as clean as NED: whereas NED works on named entities as a specific class of words in the input text, Wikification can in theory work on all words in a text. In practice, though, Wikipedia does not cover all senses of words, as WordNet does, leading to the somewhat circular task definition of assigning meaning to all words where the correct meaning is part of Wikipedia. This is why in this thesis the focus is on NED.

Representative work in this area was done by Mihalcea and Csomai [MC07], Milne and Witten [MW08b], Kulkarni et al. [KSRC09], and Ratinov et al. [RRDA11]. These and other works are introduced in Section 2.2 alongside works on NED.

2.4.3. Coreference Resolution

Coreference resolution takes a complementary view of disambiguation, one that does not rely on a repository of entities. Instead, it tries to link together phrases – not only named entities – in a single input text that denote the same entity. On the one hand, this is complementary to doing full NED, as phrases beyond named entities are considered. Take the following example sentence.

> Dylan played at the Newport Folk Festival. There, the folk singer shocked the audience by playing electric.

Example 2.3

NED would only consider "Dylan" and "Newport Folk Festival" as input phrases. Coreference resolution would also pick up "the folk singer", determining that this refers to the same entity as "Dylan", and "There" referring to the festival. Coreference resolution on a named-entity-only input set is on the other hand subsumed by NED, under the assumption that all entities mentioned in a text exist in the entity repository.

The extension of coreference resolution to a collection of documents is called cross-document coreference resolution, which comes closer to actual NED [MAD+09]. Here, phrases denoting the same entity across a collection should be linked, thus forming an implicit entity repository. An overview of the recent works in coreference resolution is given in [Ng10].

2.4.4. Named Entity Classification

Named entity classification, or NEC for short, abstracts over the entity-level, labeling mentions with their semantic types instead of going for the concrete entity directly. In

the example sentence

> Dylan recorded the album Desire in 1976

Example 2.4

it would label "Dylan" as `person`, maybe even `musician`, instead of the entity `Bob Dylan`.

Early work on NEC — in the 90s in the context of the MUC-6 conference [GS96] and early 00s in the CoNLL-2003 shared task [SDM03] — was mostly done in combination with the actual recognition of the named entities (see Section 2.1) in text. The labels to assign were very coarse, distinguishing only between entities of type `person`, `location`, `organization`, and `miscellaneous`.

The limitations of the coarse-grained types were first addressed by Fleischmann and Hovy [FH02], who introduce more fine-grained person subclasses as labels, e. g. `athlete` or `scientist`. Rahman and Ng's work [RN10] further increased the granularity of the types, but still limited the number of assigned types per mention to a single one.

More recent works [LME12, YBH+12, NTW13] have gone beyond this limitation and assign multiple types to mentions, where the types stem from an even richer type hierarchy. These works have been made possible by the fine-granular type hierarchies of recent knowledge bases like Freebase and YAGO (see Section 2.3). The works also benefit from Wikipedia as a corpus for training type classifiers, as all Wikipedia links can be used as training instances due to the direct connection of their entities and Wikipedia articles.

To give a brief glance of how well current approaches perform, the work by Yosef et al. [YBH+12] achieves precision and recall of more than 90% when training and testing on Wikipedia data. The quality on very fine-granular types (more than 500 types in total) is about 5% worse than on coarse-grained types.

3. Disambiguating Named Entities

Disambiguating named entities in natural-language text maps mentions of ambiguous names onto canonical entities like people or places, registered in a knowledge base such as DBpedia or YAGO. This chapter presents a robust method for joint disambiguation, by harnessing context from knowledge bases and using a new form of a coherence graph. It unifies prior approaches into a comprehensive framework that combines three measures: the prior probability of an entity being mentioned, the similarity between the contexts of a mention and a candidate entity, as well as the coherence among candidate entities for all mentions together. The method builds a weighted graph of mentions and candidate entities, and computes a dense subgraph that approximates the best joint mention-entity mapping. Experiments show that the new method significantly outperforms prior methods in terms of accuracy, with robust behavior across a variety of inputs.

3.1. Introduction

As defined in Section 2.2, the *named-entity disambiguation* (NED) task works on input texts like "They performed Kashmir, written by Page and Plant. Page played unusual chords on his Gibson." The question is how can a method tell that in this input sentence, "Kashmir" denotes a song by Led Zeppelin and not the Himalaya region?

The simplest heuristics for name resolution is to choose the most prominent entity for a given name. This could be the entity with the longest Wikipedia article or the largest number of incoming links in Wikipedia; or the place with the most inhabitants (for cities) or largest area, etc. Alternatively, one could choose the entity that uses the mention most frequently as a hyperlink anchor text. For the example sentence given above, all these techniques would incorrectly map the mention "Kashmir" to the Himalaya region. We refer to this suite of methods as a *popularity-based (mention-entity) prior*.

Key to improving the above approaches is to consider the *context* of the mention to be mapped, and compare it — by some *similarity measure* — to contextual information about the potential target entities. For the example sentence, the mention "Kashmir" has context words like "performed" and "chords" so that we can compare a bag-of-words model against characteristic words in the Wikipedia articles of the different candidate entities (by measures such as cosine similarity, weighted Jaccard distance, KL divergence, etc.). The candidate entity with the highest similarity is chosen.

The key to further improvements is to jointly consider multiple mentions in an input and aim for a *joint disambiguation* onto entities [KSRC09]. This approach should consider the *coherence* of the resulting entities, in the sense of semantic relatedness, and it should combine such measures with the context similarity scores of each mention-

entity pair. In our example, one should treat "Page", "Plant" and "Gibson" also as named-entity mentions and aim to disambiguate them together with "Kashmir".

Joint disambiguation works very well when a text contains mentions of a sufficiently large number of entities within a thematically homogeneous context. If the text is very short or is about multiple, unrelated or weakly related topics, collective mapping tends to produce errors by directing some mentions towards entities that fit into a single coherent topic but do not capture the given text. For example, a text about a football game between "Manchester" and "Barcelona" that takes place in "Madrid" may end up mapping either all three of these mentions onto football clubs (i.e., Manchester United, FC Barcelona, Real Madrid) or all three of them onto cities. The conclusion here is that none of the prior methods for named-entity disambiguation is robust enough to cope with such difficult inputs.

We cast the joint disambiguation into the following graph problem: mentions from the input text and candidate entities define the node set, and we consider weighted edges between mentions and entities, capturing context similarities, and weighted edges among entities, capturing coherence. The goal on this combined graph is to identify a dense subgraph that contains exactly one mention-entity edge for each mention, yielding the most likely disambiguation. Such graph problems are NP-hard, as they generalize the well-studied Steiner-tree problem. We develop a greedy algorithm that provides high-quality approximations, and is customized to the properties of our mention-entity graph model.

In addition to improving the above assets for the overall disambiguation task, our approach — dubbed *AIDA* for Accurate Online Disambiguation of Named Entities — gains in robustness by using components selectively in a self-adapting manner. To this end, we have devised the following multi-stage procedure.

- For each mention, we compute popularity priors and context similarities for all entity candidates as input for our tests.
- We use a threshold test on the prior to decide whether popularity should be used (for mentions with a very high prior) or disregarded (for mentions with several reasonable candidates).
- When both the entity priors and the context similarities are reasonably similar in distribution for all the entity candidates, we keep the best candidate and remove all others, fixing this mention *before* running the coherence graph algorithm.

We then run the coherence graph algorithm on all the mentions and their remaining entity candidates. This way, we restrict the coherence graph algorithm to the critical mentions, in situations where the goal of coherence may be misleading or would entail high risk of degradation.

This chapter then makes the following novel contributions:

- A framework for combining popularity priors, similarity measures, and coherence into a robust disambiguation method (Section 3.3).

- New measures for defining mention-entity similarity (Section 3.3.4).
- A new algorithm for computing dense subgraphs in a mention-entity graph, which produces high-quality mention-entity mappings (Section 3.4.2).
- An empirical evaluation on a demanding corpus (based on additional annotations for the dataset of the CoNLL 2003 NER task), with significant improvements over state-of-the-art competitors (Section 3.6).

3.2. Related Work on Joint Entity Disambiguation

Cucerzan [Cuc07] was the first to recognize the usefulness of disambiguating all mentions in an input document at once to capture the topical coherence in a document. The main idea was to expand the context of candidate entities with all their associated Wikipedia categories and preferring entities that agree with other candidates' categories. However, Cucerzan's method does not perform joint disambiguation; instead it addresses each mention separately and prefers those that work well with all other mentions' candidates — without knowing the correct one yet.

The first work with an explicit collective-learning model for the joint disambiguation of all mentions has been done by Kulkarni et al. [KSRC09]. This method models the disambiguation problem as a graph, combining the mention-entity prior and context with the entity-entity coherence in a way very similar to ours. There are three key differences to our AIDA approach:

- The way the context similarity is computed. AIDA represents the entity context as bag of phrases instead of bag of words, while still allowing for partial matches.
- The means by which the solution is derived from the initial graph. AIDA uses a greedy graph algorithm, Kulkarni et al. use an ILP-solver.
- Kulkarni et al. use the prior, context, and coherence for all input texts in the same manner. AIDA selectively enables and disables the features for certain inputs, and fixes the candidate entities for some mentions to guide the joint disambiguation.

The experiments in [KSRC09] show that this method is superior to the best prior approaches, most notably [MW08b]. Both [KSRC09, MW08b] and other related works are described in Section 2.2.2.

3.3. AIDA Components

3.3.1. Mentions and Ambiguity

We consider an input text (Web page, news article, blog posting, etc.) with mentions (i.e., surface forms) of named entities (people, music bands, songs, universities, etc.) and aim to map them to their proper entries in a knowledge base, thus giving a disambiguated meaning to entity mentions in the text. We first identify noun phrases (e.g., "Larry

Page", "Apple", "Chief Seattle", "Dances with Wolves", etc.) that potentially denote named entities. We use the Stanford NER Tagger [FGM05] to discover these and segment the text accordingly.

3.3.2. Entity Candidates

For possible entities (with unique canonical names) that a mention could denote, we harness YAGO. For each entity it provides a set of short names (e.g., "Apple" for `Apple Inc.`) and paraphrases (e.g., "Big Apple" for `New York City`). In YAGO, these are available by the `label` relation, which in turn is harvested from Wikipedia disambiguation pages, redirects, links, and the title of the entity itself. An entity becomes a candidate for a mention if any of its names matches the mention fully.

One additional heuristic makes sure that all-upper-case mentions are also considered, even if no entity name is present. Writing a word in all-upper-case is often used as a syntactic marker in texts (instead of bold face or italic). This is especially important in news-wire texts where there are no other formatting options. Normalizing names by lower-casing everything is one possibility, however this conflates mentions like "US" and "us", artificially inflating the candidate space. A large fraction of short, all-upper-case mentions are acronyms, mostly three-letter-acronyms like "USA", "NSA", "CIA", "FBI", etc. To distinguish these from lower-case words, AIDA matches all names of 3 or less characters case-sensitive. For all mentions that have more than 3 characters, we normalize both the mention and all entity names to all-upper-case before matching, retrieving `Apple Inc.` as candidate for the mention "APPLE" even though our knowledge base only contains "Apple".

3.3.3. Popularity Prior

Prominence or popularity of entities can be seen as a probabilistic prior for mapping a name to an entity. The most common way of estimating this are the Wikipedia-based frequencies of particular names in link anchor texts referring to specific entities, or number of inlinks. We found a model based on Wikipedia link anchors to be most effective: For each surface form that constitutes an anchor text, we count how often it refers to a particular entity. For each name, these counts provide us with an estimate for a probability distribution over candidate entities. For example, "Kashmir" refers to `Kashmir` (the region) in 90.91% of all occurrences and in 5.45% to `Kashmir (Song)`.

3.3.4. Mention-Entity Similarity

The key for mapping mentions onto entities are the contexts on both sides of the mapping.

On the **mention side**, we use all tokens in the entire input text (except stopwords and the mention itself) as context. This way, we can represent a mention as a set of (weighted) words or phrases that it co-occurs with. We experimented with a distance discount to discount the weight of tokens that are further away, but this did not improve the results for our test data.

On the **entity side**, we associate each entity with characteristic keyphrases or salient words, pre-computed from Wikipedia articles. For example, `Larry Page` would have keyphrases like "Stanford", "search engine", etc., whereas `Jimmy Page` may have keyphrases "Gibson guitar", "hard rock", etc. These are the inputs for an offline data-mining step to determine characteristic *keyphrases* for each entity and their statistical weights. As keyphrase candidates for an entity we consider its corresponding Wikipedia article's link anchors texts, including category names, citation titles, and external references. We extended this further by considering also the titles of articles linking to the entity's article. All these phrases form the keyphrase set of an entity: $KP(e)$.

For each word w that occurs in a keyphrase, we compute a *specificity weight* with regard to the given entity: the NPMI (normalized pointwise mutual information) between the entity e and the keyword w is computed as follows:

$$\text{npmi}(e,k) = \frac{\text{pmi}(e,k)}{-\log p(e,k)} \ , \tag{3.1}$$

where

$$\text{pmi}(e,k) = \log \frac{p(e,k)}{p(e)p(k)} \ . \tag{3.2}$$

Keywords with $\text{npmi}(e,k) \leq 0$ are discarded for the actual NED. The co-occurrence probability is estimated as follows:

$$p(e,w) = \frac{\left|w \in \left(KP(e) \cup \bigcup_{e' \in IN_e} KP(e')\right)\right|}{N} \ . \tag{3.3}$$

reflecting if w is contained in the keyphrase set of e (if w matches any token of the keyphrase) or any of the keyphrase sets of an entity linking to e, $IN(e)$, with N denoting the total number of entities.

Keyphrases may occur only partially in an input text. For example, the phrase "Grammy Award winner" associated with entity `Jimmy Page` may occur only in the form "Grammy winner" near some mention "Page". Therefore, our algorithm for the similarity of mention m with regard to entity e computes partial matches of e's keyphrases in the text. This is done by matching individual words and rewarding their proximity in an appropriate score. To this end we compute, for each keyphrase, the shortest window of words that contains a maximal number of words of the keyphrase. We refer to this window as the phrase's *cover* (cf. [TKW11]). For example, matching the text "winner of many prizes including the Grammy" results in a cover length of 7 for the keyphrase "Grammy award winner". By this rationale, the score of partially matching phrase q in a text is set to:

$$\text{score}(q) = z \left(\frac{\sum_{w \in cover} weight(w)}{\sum_{w \in q} weight(w)} \right)^2 \tag{3.4}$$

where $z = \frac{\#\ matching\ words}{length\ of\ cover(q)}$ and $weight(w)$ is either the NPMI weight (see Equation 3.1) or the collection-wide IDF weight of the keyphrase word w. Note that the second factor is squared, so that there is a superlinear reduction of the score for each word that is missing in the cover.

The IDF is computed using the standard formula:

$$\text{idf}(k) = \log_2 \frac{N}{df(k)}, \tag{3.5}$$

where N is the size of the collection (in this case the total number of entities in the knowledge base). $df(k)$ is the number of entities that have at least one keyphrase k (for keyphrase IDF), or at least one keyphrase that contains the token k (for keyword IDF).

For the similarity of a mention m to candidate entity e, this score is aggregated over all keyphrases of e and all their partial matches in the text, leading to the *similarity score*

$$\text{simscore}(m, e) = \sum_{q \in KP(e)} score(q) \tag{3.6}$$

3.3.5. Entity-Entity Coherence

Coherence is a key asset because most texts deal with a single or a few semantically related topics such as rock music or Internet technology or global warming, but not everything together.

An asset that knowledge bases like DBpedia and YAGO provide us with is the same-as cross-referencing to Wikipedia. We quantify the coherence between two entities by the *incoming links* I_e that their Wikipedia articles share. This approach has been refined by Milne and Witten [MW08a], based on the work by Vitanyi [LCL$^+$04, CV07]. It takes into account the total number N of entities in the (Wikipedia) collection, leading to the following computation of relatedness between two entities e and f:

$$\text{MW}(e, f) = 1 - \frac{\log\left(\max\{|I_e|, |I_f|\}\right) - \log(|I_e \cap I_f|)}{\log(N) - \log(\min\{|I_e|, |I_f|\})}. \tag{3.7}$$

if > 0 and else set to 0.

3.3.6. Overall Objective Function

To aim for the best disambiguation mappings, our framework combines prior, similarity, and coherence measures into a combined objective function: for each mention m_i, $i = 1..k$, select entity candidates e_{j_i}, one per mention, such that

$$\begin{aligned} &\alpha \cdot \sum_{i=1..k} \text{prior}(m_i, e_{j_i}) + \\ &\beta \cdot \sum_{i=1..k} \text{sim}(\text{cxt}(m_i), \text{cxt}(e_{j_i})) + \\ &\gamma \cdot \text{coh}(e_{j_1} \in E_{m_1} \ldots e_{j_k} \in E_{m_k}) = \max! \end{aligned}$$

where $\alpha + \beta + \gamma = 1$, E_{m_i} is the set of candidate entities for m_i, cxt( ) denotes the context of mentions and entities, respectively, and coh( ) is the coherence function for a set of entities.

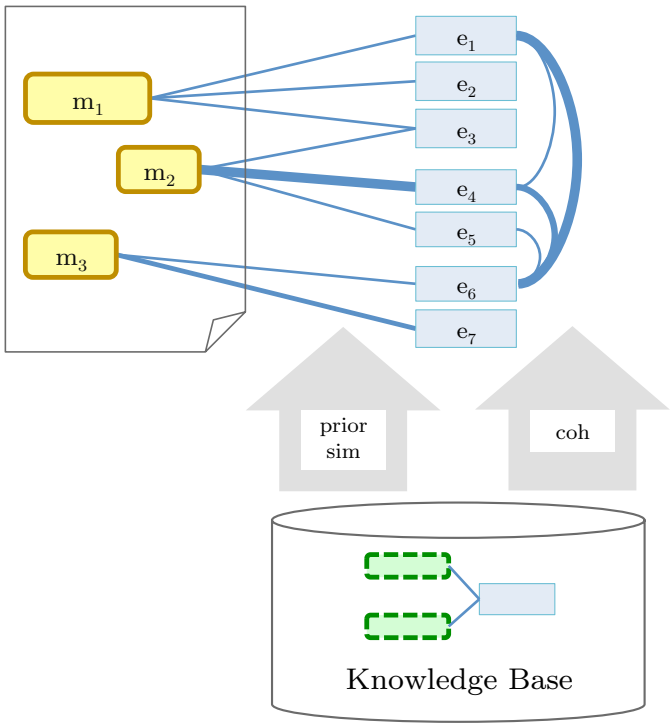

Figure 3.1.: Disambiguation Framework

This framework is illustrated in Figure 3.1. For robustness, our solution selectively enables or disables the three components, based on tests on the mentions of the input text; see Section 3.4.

3.4. Graph Model and Disambiguation Algorithm

3.4.1. Mention-Entity Graph

From the popularity, similarity, and coherence measures discussed in Section 3.3, we construct a weighted, undirected graph with mentions and candidate entities as nodes. As shown in the example of Figure 3.2, the graph has two kinds of edges:

- A mention-entity edge is weighted with a similarity measure or a combination of popularity and similarity measure. Our experiments will use a linear combination with coefficients learned from withheld training data.
- An entity-entity edge is weighted based on Wikipedia-link overlap.

There are several details to the graph construction. Once all mention-entity weights and entity-entity weights have been computed, they are both scaled to be in $[0.0, 1.0]$

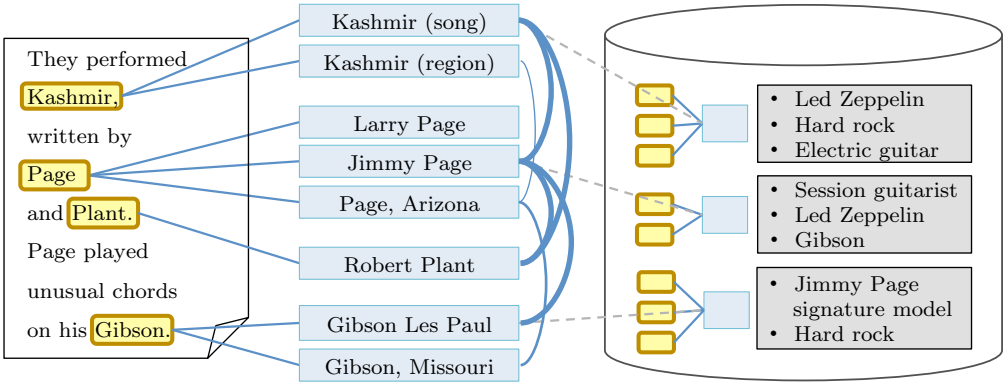

Figure 3.2.: Mention-Entity Graph Example

respectively. Then the edge weights are rescaled so that the average weight of all mention-entity edges is equal to the average weight of all entity-entity edges. These final weights can then be adjusted by the γ parameter balancing coherence vs. similarity and prior.

Note that the mention-entity graph tends to be very large even for relatively short input texts. The reason is that the YAGO knowledge base offers a wealth of possible meanings. For example, for country names, the candidate entities also include national sports teams in a variety of sports; for common lastnames (e.g., "Smith") or firstnames-only mentions, there are even more candidates. Moreover, the mention-entity graph is often very dense on the entities side, because of the rich inter-linkage in Wikipedia.

3.4.2. Graph Algorithm

Given a mention-entity graph, our goal is to compute a *dense subgraph* that would ideally contain all mention nodes and exactly one mention-entity edge for each mention, thus disambiguating all mentions. We face two main challenges here. The first is how to specify a notion of density that is best suited for capturing the coherence of the resulting entity nodes. The seemingly most natural approach would be to measure the density of a subgraph in terms of its total edge weight. Unfortunately, this will not work robustly for the disambiguation problem. The solution could be dominated by a few entity nodes with very high weights of incident edges, so the approach could work for prominent targets, but it would not achieve high accuracy also for the long tail of less prominent and more sparsely connected entities. We need to capture the weak links in the collective entity set of the desired subgraph. For this purpose, we define the *weighted degree* of a node in the graph to be the total weight of its incident edges. We then define the density of a subgraph to be equal to the *minimum weighted degree* among its entity nodes. Our goal is to compute a subgraph with maximum density, while observing constraints on the subgraph structure.

The second critical challenge that we need to face is the computational complexity. Dense-subgraph problems are almost inevitably NP-hard as they generalize the Steiner-

Algorithm 1: Graph Disambiguation Algorithm

Input: weighted graph of mentions and entities
Output: result: graph with one edge per mention
`/* pre-processing phase */`
foreach entity **do**
  compute shortest (weighted) path to each mention;
  $\text{distance}_e \leftarrow$ sum of shortest paths;
keep entities_c with the lowest distance_e, drop the others;
`/* main loop */`
objective $\leftarrow \frac{\text{minimum weighted degree of entities}_c}{\#\text{entities}_c}$;
while graph *has* non-taboo entity **do**
  `/* entity is taboo if last candidate for any mention */`
  `nte` $\leftarrow$ non-taboo entity with lowest weighted degree;
  $\text{entities}_c \leftarrow \text{entities}_c \setminus$ `nte`;
  Remove all incident edges of `nte` from graph;
  `mwd` $\leftarrow \frac{\text{minimum weighted degree of entities}_c}{\#\text{entities}_c}$;
  if *mwd* $>$ objective **then**
    solution $\leftarrow$ current graph;
    objective $\leftarrow$ `mwd`;
`/* post-processing phase */`
if mention-entity *combination count feasible* **then**
  enumerate all possible mention-entity pairs in solution, add pairs maximizing the sum of all edge weights to result;
else
  do a hill-climbing search for feasible number of mention-entity pairs, add pairs of the maximum-edge-weight-sum solution to result;

tree problem. Hence, exact algorithms on large input graphs are infeasible.

To address this problem, we extend an approximation algorithm of Sozio and Gionis [SG10] for the problem of finding strongly interconnected, size-limited groups in social networks. The algorithm starts from the full mention-entity graph and iteratively removes the entity node with the smallest weighted degree. Among the subgraphs obtained in the various steps, the one maximizing the minimum weighted degree will be returned as output. A graph with fewer nodes is preferred, so the minimum weighted degree is divided by the number of nodes in the graph. To guarantee that we arrive at a full mention-entity mapping for all mentions, we enforce each mention node to remain connected to at least one entity. However, this constraint may lead to suboptimal results. For this reason, we apply a pre-processing phase to prune the entities that are too distant from the mention nodes. For each entity node, we compute the distance from the set of all mention nodes in terms of the sum of the squared shortest-path distances.

We then restrict the input graph to the entity nodes that are closest to the mentions. An experimentally determined good choice for the size of this set is five times the number of the mention nodes. Then the iterative greedy method is run on this smaller subgraph. Algorithm 1 summarizes this procedure, where an `entity` is `taboo` if it is the last candidate for a mention it is connected to.

The output of the main loop would often be close to the desired result, but may still have more than one mention-entity edge for one or more mentions. At this point, however, the subgraph is usually small enough to consider an exhaustive enumeration and assessment of all possible solutions. This is one of the options that we have implemented as post-processing step. Alternatively, we can perform a faster local-search algorithm. Candidate entities are randomly selected with probabilities proportional to their weighted degrees. This step is repeated for a pre-specified number of iterations, and the best configuration with the highest total edge-weight is used as final solution.

3.5. Robustness Issues

The graph algorithm generally performs well. However, it may be misled in specific situations, namely, if the input text is very short, or if it is thematically heterogeneous. To overcome these problems, we introduce two *robustness tests* for individual mentions and, depending on the tests' outcomes, use only a subset of our framework's features and techniques.

3.5.1. Prior Robustness Test

Our first test ensures that the popularity prior does not unduly dominate the outcome if the true entities are dominated by false alternatives. We check, for each mention, whether the popularity prior for the most likely candidate entity is above some threshold ρ, e. g. above 90% probability. If this is not the case, then the prior is completely disregarded for computing the mention-entity edge weights. Otherwise, the prior is combined with the context-based similarity computation to determine edge weights. We never rely solely on the prior.

3.5.2. Coherence Robustness Test

As a test for whether the coherence part of our framework makes sense or not, we compare the popularity prior and the similarity-only measure, on a per-mention basis. For each mention, we compute the $L1$ distance between the popularity-based vector of candidate probabilities and the similarity-only-based vector of candidate probabilities:

$$\sum_{i=1..k} |\text{prior}(m, e_i) - \text{simscore}(m, e_i)|$$

This difference is always between 0 and 2. If it exceeds a specified threshold λ (e.g., 0.9), the disagreement between popularity and similarity-only indicates that there is a situation that coherence may be able to fix. If, on the other hand, there is hardly any

articles	1,393
mentions (total)	34,956
mentions with no entity	7,136
words per article (avg.)	216
mentions per article (avg.)	25
distinct mentions per article (avg.)	17
mentions with candidate in KB (avg.)	21
entities per mention (avg.)	73
initial annotator disagreement (%)	21.1

Table 3.1.: *CoNLL* Dataset Properties

disagreement, using coherence as an additional aspect would be risky for thematically heterogeneous texts and should better be disabled. In that case, we choose an entity for the mention at hand, using the combination of prior and similarity. Only the winning entity is included in the mention-entity graph, all other candidates are omitted for the graph algorithm. The robustness tests and the resulting adaptation of our method are fully automated.

3.6. Evaluation

3.6.1. Setup

System

For the experiments, we assume all mentions to be present as input, the YAGO2 knowledge base [HSBW13] as a repository of entities, and the English Wikipedia edition (as of 2010-08-17) as a source of mining keyphrases and various forms of weights.

Datasets

There is no established benchmark for NED on fully annotated documents, the very popular TAC-KPB challenge evaluates only one mention per input document. The best prior work by [KSRC09] compiled its own manually annotated dataset, sampled from online news. Unfortunately, this data set is fairly small (102 short news articles, about 3,500 proper noun mentions). Moreover, its entity annotations refer to an old version of Wikipedia, and the mentions are annotated with an automated method. To avoid unfair comparisons, we created our own dataset based on CoNLL 2003 data, extensively used in prior work on NER tagging [SDM03].

This consists of proper noun annotations for 1,393 Reuters news-wire articles. We manually annotated all these proper nouns with corresponding entities in YAGO2. Each mention was disambiguated by two students and resolved by us in case of conflict. This data set is referred to as *CoNLL* in the following and fully available at `http://www.`

`mpi-inf.mpg.de/yago-naga/aida/`. Table 3.1 summarizes properties of the dataset. For our experiments, we use the same splits as in the original CoNLL shared task, namely the documents from 1 to 946 (referred to as *train* originally) are used for training, 947 to 1162 (referred to as *testa* originally) are used as development set and for setting the hyperparameters, and 1163 to 1393 (referred to as *testb* originally) are used as test set.

Methods under comparison

Our framework includes many variants of prior methods from the literature. We report experimental results for some of them. AIDA's parameters were tuned by line-search on 216 withheld development documents. We found the following to work best:

- Threshold for prior test: $\rho = 0.9$.
- Weights for popularity, similarity, coherence: $\alpha = 0.34$, $\beta = 0.26$, $\gamma = 0.40$. For the graph representation this means:
  - The entity-entity edge weights are multiplied with 0.40.
  - For the mention-entity edge weights: if $\rho > 0.9$, the edge weight w is the linear combination of the *prior* and the keyphrase-similarity *sim*:

    $$w = 0.566 \cdot \text{prior} + 0.433 \cdot \text{sim}$$

    Otherwise only *sim* will be used as weight, $w = sim$. Finally, w is multiplied by 0.60.
- Initial number of entities in graph: $5 \cdot \#mentions$.
- Threshold for coherence test: $\lambda = 0.9$.

We checked the sensitivity of the hyper-parameter settings and found the influence of variations to be small, e. g. when varying λ within the range [0.5,1.3], the changes in accuracy are within 1%.

The baseline for our experiments is the collective-inference method of [KSRC09], which outperforms simpler methods (such as [MW08b]). We refer to this method as *Kul_CI*. Since neither source code nor executables for this method are available, we re-implemented it. In addition, we compare against (our re-implementation of) the method of [Cuc07], referred to as *Cuc*. For all methods, weights for combining components were obtained by training a SVM classifier on 946 withheld *CoNLL* training documents.

Evaluation Measures

We are evaluating the quality of the NED methods with measures defined over the following sets: D, the collection of documents; G_d, all unique mentions in document $d \in D$ annotated by a human annotator with a gold standard entity; and A_d, all unique mentions in document $d \in D$ automatically annotated by a method. The measures used are:

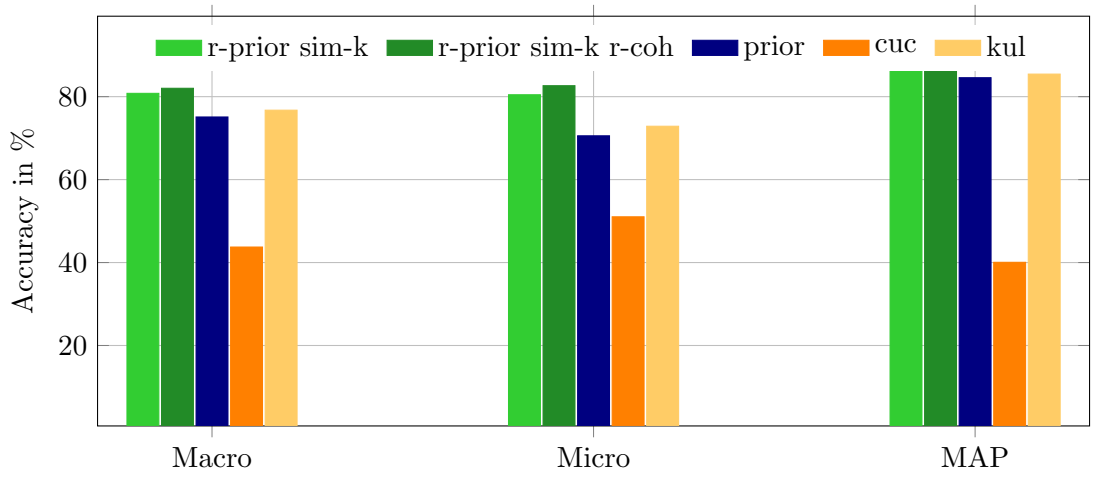

Figure 3.3.: Experimental results on 231 *CoNLL* documents

- **Micro Average Accuracy**, the fraction of correctly disambiguated gold mentions in the whole collection:

$$\text{MicA} = \frac{\left|\bigcup_{d\in D} G_d \cap \bigcup_{d\in D} A_d\right|}{\left|\bigcup_{d\in D} G_d\right|}$$

- **Document Accuracy**, the fraction of correctly disambiguated mentions in a single document d:

$$\text{DA}_d = \frac{|G_d \cap A_d|}{|G_d|}$$

- **Macro Average Accuracy**, the document-averaged fraction of correctly disambiguated mentions:

$$\text{MacA} = \sum_{d\in D} \frac{DA_d}{|D|}$$

Mentions with Out-of-Knowledge-Base Entities

As we use a knowledge base with millions of entities, we decided to neglect the situation that a mention may refer to an unknown entity not registered in the knowledge base. We consider only mention-entity pairs where the ground-truth gives a known entity, and thus ignore roughly 20% of the mentions without known entity in the ground-truth. We get back to the problem of addressing the discovery of such mentions in Chapter 5.

3.6.2. Experimental Results

The results of AIDA vs. the collective-inference method of [KSRC09] and the entity disambiguation method of [Cuc07] on 231 test documents (CoNLL 1163 to 1393) are summarized in Figure 3.3, detailed results are given in Table 3.2.

	Our Methods					Competitors				
	sim-k	prior sim-k	r-prior sim-k	r-prior sim-k coh	r-prior sim-k r-coh	prior	Cuc	Kul_s	Kul_sp	Kul_CI
MacA	79.11	71.57	80.47	82.25	**82.63**	70.57	43.74	58.06	76.74	76.74
MicA	77.99	76.15	80.80	81.56	**82.03**	75.10	51.03	63.42	72.31	72.87

Table 3.2.: Experimental results on *CoNLL* test set (all values in %)

The table includes variants of our framework, with different choices for the similarity and coherence computations. The shorthand notation for the combinations in the table is as follows:

prior popularity prior

r-prior popularity prior with robustness test

sim-k keyphrase based similarity measure

coh graph coherence

r-coh graph coherence with robustness test

The shorthand names for competitors are:

Cuc Cucerzan's [Cuc07] disambiguation method

Kul_s Kularni et al. [KSRC09] similarity measure only

Kul_sp Kul_s combined with plus popularity prior

Kul_CI Kul_sp combined with coherence

All coherence methods use the Milne-Witten inlink overlap measure *mw_coh*.

The most important measure is macro/micro accuracy, which corresponds to the overall correctness of the methods for all mentions that are assigned to an entity in the ground-truth data. Our *sim-k* precision is already very good. Unconditionally combining *prior* and *sim-k* degrades the quality, but including the prior robustness test (*r-prior sim-k*) improves the results significantly. The precision for our best method, the prior- and coherence-tested keyphrase-based mention-entity similarity (*r-prior sim-k r-coh*), significantly outperforms all competitors (with a p-value of a paired t-test < 0.01). Our macro-averaged accuracy is 82.63%, whereas *Kul_CI* only achieves 76.74%. Even *r-prior sim-k*, without any coherence, significantly outperforms *Kul_CI* (with coherence) with a p-value of < 0.01. In micro-average accuracy, the differences are even higher, showing that we perform better throughout all documents.

We also tried the [MW08b] web service on a subset of our test collection, but this was obviously geared for Wikipedia linkage and performed poorly.

3.6.3. Discussion

Our keyphrase-based similarity measure performs better than the *Kul_s* measure, which is a combination of 4 different entity contexts (abstract tokens, full text tokens, inlink anchor tokens, inlink anchor tokens + surrounding tokens), 3 similarity measures (Jaccard, dot product, and tf.idf cosine similarity), and the popularity prior. Adding the prior to our similarity measure by linear combination degrades the performance. We found that our measure already captures a notion of popularity because popular entities have more keyphrases and can thus accumulate a higher total score. The popularity should only be used when one entity has a very high probability, and introducing the robustness test for the prior achieved this, improving on both our similarity and *Kul_sp*.

Further investigating potential problems, we found that the coherence can be led astray when parts of the document form a coherent cluster of entities, and other entities are then forced to be coherent to this cluster. To overcome this issue, we introduced the coherence robustness test, and the results with *r-coh* show that it makes sense to fix an entity for a mention when the prior and similarity are in reasonable agreement. Adding this coherence test leads to a significant (p-value < 0.05) improvement over the non-coherence based measures in both micro- and macro-average precision. Our experiments showed that when adding this coherence test, around $\frac{2}{3}$ of the mentions are solved using local similarity only and are assigned an entity before running the graph algorithm. In summary, we observed that the AIDA configuration with *r-prior*, keyphrase-based *sim-k*, and *r-coh* significantly outperformed all competitors.

3.6.4. Interesting Examples

Metonymy "In **metonymy** (Greek for 'a change of name') the literal term for one thing is applied to another with which is has become closely associated because of a recurrent relation in common experience" [AH12]

This figure of speech occurs frequently in news. Take the example sentence from Document 1223 in the CoNLL collection.

> Interfax news agency quoted First Deputy Foreign Minister Igor Ivanov as saying *Moscow* was ready for 'most active and constructive' work with Albright.

Example 3.5

It is not the city of Moscow that is ready to work, but the government of Russia. In the creation of the CoNLL-YAGO corpus, a guideline told annotators to choose the entity that is really referred to, dealing with metonymy. In the commonly occurring case of *state capital as state* we had an explicit guideline to annotate the country.

The AIDA entity disambiguation is able to deal with metonymy by ensuring coherence among entities. In the above example, the entities `Interfax` and `Madeline Albright` are not related to `Moscow`, but they are strongly related to `Russia`. Without the coherence feature, AIDA incorrectly disambiguates to the actual city.

Another case where coherence is essential to deal with metonymy is sports news. Here,

city or country names usually refer to sports teams, and city names can also refer to the stadiums where the actual games take place. The following example sentence is taken from Document 1164.

> Italy recalled Marcello Cuttitta on Friday for their friendly against Scotland at Murrayfield [...].

Example 3.6

Both Italy and Scotland refer to the national rugby teams, not the country, and Murrayfield to the stadium, not the city. As Marcello Cuttitta is a rugby player, the sports teams and the stadium is chosen correctly thanks to coherence.

Unfortunately, coherence can also cause errors, for example in the following sentence from Document 1223.

> His comments came just minutes after the latest set of open skies talks ended in London with no deal signed

Example 3.7

Here, AIDA incorrectly disambiguates "London" to `United Kingdom`, as the other entities in the document are mostly from the politics domain. For such cases, the coherence robustness test is essential, fixing those mentions before the graph algorithm is run and coherence can create such errors. In this specific example, however, it fails.

Entity Granularity Frequent problems arise when entities are very close in meaning because of the fine granularity of articles in Wikipedia. For example in Document 1270, AIDA disambiguates the mention "Sumitomo Bank" is to `Sumitomo Mitsui Banking Corporation` instead of `The Sumitomo Bank`. The former is a merger of the second with another company, so it is not fully correct. However, it is still very close to the correct one, and for the average reader and a large number of use cases, this distinction is not important.

Another issue with granularity stems from the temporal nature of entities. A very common example are sports events. For each world cup, sports season, or Olympics, Wikipedia has an article covering the actual event as of the year. It usually also contains an article about the event series. Often, there is enough evidence to actually distinguish both, as in this sentence Document 1163.

> Japan began the defence of their Asian Cup title with a lucky 2-1 win against Syria [...].

Example 3.8

AIDA correctly disambiguates to `1996 AFC Asian Cup`, and not the more general `AFC Asian Cup`, however this is not the case for all such examples.

Bad Dictionary The dictionary that provides candidate entities for mentions is derived from Wikipedia redirects, disambiguation pages, and link anchors. Some of these are noisy, which can lead to errors in the disambiguation. As an example, there is a Wikipedia link that refers to `English Language` from the mention "German". Any disambiguation algorithm will face problems with such wrong candidates, and especially coherence-based disambiguation can be lead astray.

3.7. Summary

The AIDA system provides an integrated NED method using popularity, similarity, and graph-based coherence, and includes robustness tests for self-adaptive behavior. AIDA performs significantly better than state-of-the-art baselines, mitigating the problem of brittle disambiguation quality of previous systems. Furthermore, all the annotations of the news-wire corpus are provided to the research community, forming the largest gold standard corpus for entity recognition and disambiguation to date.

The system is fully implemented and accessible online (`http://www.mpi-inf.mpg.de/yago-naga/aida/`). The software and source code are available for download and use. The CoNLL-YAGO dataset of 1,393 annotated news-wire articles is available at the same URL.

4. Computing Entity Relatedness

Measuring the semantic relatedness between two entities is the basis for numerous tasks in information retrieval, natural language processing, and Web-based knowledge extraction. This chapter focuses on the role of semantic relatedness as a coherence measure for entity disambiguation. To this end, we have developed a novel notion of semantic relatedness between two entities represented as sets of weighted (multi-word) keyphrases, with consideration of partially overlapping phrases. This measure improves the quality of prior link-based models, and also eliminates the need for (usually Wikipedia-centric) explicit inter-linkage between entities. Thus, our method is more versatile and can cope with long-tail and newly emerging entities that have few or no links associated with them. For efficiency, we have developed approximation techniques based on min-hash sketches and locality-sensitive hashing. Our experiments on semantic relatedness and on named entity disambiguation demonstrate the superiority of our method compared to state-of-the-art baselines.

4.1. Introduction

Semantic relatedness measures between entities are a fundamental asset for many applications dealing with natural language processing, text mining, or Web analytics [BH06].

Its usefulness has been shown in a wide variety of tasks: general word sense disambiguation [SM07, Nav09, PN10], query expansion for information retrieval [KZ12, PF11], Web-based information extraction [APRA10], knowledge base population [DMR+10, SSPM11], and more. In this thesis, we focus on semantic relatedness of entities and its role in NED, as defined in Section 2.2.

Consider the terms "Cash" and "Jackson" as an example. When trying to measure the relatedness between these surface forms, we face high ambiguity and would assess them as weakly related only. Most possible meanings of "Cash" and "Jackson", such as movies entitled "Cash" and people or cities named "Jackson", have nothing in common. At the entity level, however, once we identify the surface form "Cash" with the singer *Johnny Cash* and "Jackson" with a song he performed, *Jackson (song)*, the relatedness is very high. In our example, the entity names in a sentence like "The audience got wild when Cash performed Jackson." can be correctly disambiguated because only the singer-song combination, out of thousands of other pairs of entities named "Cash" and "Jackson", yields a semantically coherent interpretation.

State-of-the-art measures for entity relatedness that perform well in NED and other tasks are based on the extensive link structure of Wikipedia. Most notably, the *Milne-Witten measure* [MW08a], previously introduced in Section 3.3.5, considers the overlap of the incoming links to two entity articles as a notion of their relatedness. This is a

great asset for the millions of entities known to Wikipedia, but it is a limitation for other entities that do not (yet) have a Wikipedia page. For example, in the sentence "The audience was haunted when Cave performed Hallelujah.", the correct meaning of "Hallelujah" is neither the Händel chorus nor the Leonard Cohen composition, but a totally different song by the Australian singer Nick Cave. That song has a Web page in the music portal last.fm but there is no Wikipedia article for the song. Link-based relatedness has no chance to capture the tight connection between Nick Cave and his song. This limitation already applies to the long tail of not so prominent Wikipedia entities whose articles have only few incoming links. In these cases, the Milne-Witten measure does not properly capture the semantic relatedness between entities. Our goal is to overcome this limitation of Wikipedia-link-based relatedness measures for named entities.

A relatedness measure for link-poor and out-of-Wikipedia entities needs to tap into information that is more widely available and can be gathered with reasonable effort. In this work, we use entity-specific *keyphrases* to this end [MT04, CCCX11]. Keyphrases can be mined from any text describing an entity: people not popular enough for Wikipedia have personal homepages, small companies have websites, singers and songs are discussed in online communities like last.fm or YouTube, and so on. For our difficult example, the last.fm page on Nick Cave's song contains keyphrases like "No More Shall We Part", "Australian singer", "Warren Ellis", "eerie cello", "Nick Cave and the Bad Seeds", etc. These exhibit significant overlap with phrases contained in the Wikipedia article about Nick Cave (or, alternatively, his Web page), thus giving cues for the high relatedness. We propose a new notion of entity relatedness, coined KORE, based on the overlap of two sets of keyphrases.

While the idea for this approach may seem obvious, it entails a number of technical difficulties, addressed in this chapter:

Keyphrases and Weights Multi-word keyphrases are more informative than single-token keywords and thus preferable for characterizing entities. On the other hand, this makes it less clear which word sequences should be considered as salient keyphrases, and how we could associate them with weights that can be computed in an effective and efficient manner.

Partial Matches When comparing the keyphrases of two entities, the chance of finding exact matches drops with the length of multi-word phrases. For example, we may not find the phrases "Australian singer" and "Nick Cave and the Bad Seeds", associated with the song Hallelujah, in this exact form in the keyphrase-set of Nick Cave, where we may instead have "Australian male singer" and "Cave's Bad Seeds". Therefore, a viable notion of keyphrase-based overlap measure must consider partial matches between phrases as well.

Efficient Computation In NED, we are interested in the relatedness of all entity pairs in the candidate space. For n potentially relevant entities, a straightforward approach

would require $\binom{n}{2}$ computations. If we consider a short news article with say 10 mentions of ambiguous entity names, each of which has 10 possible meanings, we obtain 100 candidate entities and would need to compute the relatedness for 5000 pairs. This is too expensive for most online applications, and totally infeasible for longer input texts or mentions with higher degrees of ambiguity.

Our approach addresses the above challenges in the following way. We extract particular kinds of noun phrases from entity-specific pages, and compute weights for both entire phrases and their constituent words. We use MI (mutual information) for phrases, contrasting the entity page with the entire Wikipedia corpus (or the Web), and idf (inverse document frequency) for single words, based on the Wikipedia corpus. We define the KORE measure in a two-stage manner, thus allowing for partial matches between keyphrases:

1. two keyphrases are paired up if there is a partial match of at least one word, with word-level weights influencing the matching score;

2. a weighted Jaccard coefficient captures the overlap between the keyphrase-sets of two entities, where the scores of the partially matching keyphrase pairs are aggregated and the phrase-level weights are considered.

Finally, to solve the efficiency challenge, we use a two-level approximation technique: keyphrases, as sequences of words, are approximated by min-hash sketches, and these sketches are organized by locality-sensitive hashing (LSH). This way, for a given input set of entity pairs, we compute their relatedness only if their keyphrase representations are mapped to at least one common LSH bucket.

The novel contributions in this chapter are as follows:

- KORE improves the quality of assessing entity relatedness, compared to state-of-the-art link-based methods, while reducing the dependence on Wikipedia linkage and tapping into out-of-Wikipedia entities (Section 4.3.3).

- KORE is integrated into a joint-inference NED method, outperforming the best prior methods on mentions of long-tail entities.

- KORE can be efficiently computed and avoids the quadratic complexity of all-pairs comparisons in NED-like tasks by a judiciously devised two-stage hashing technique (Section 4.4).

- Our improvements in the quality of semantic relatedness and NED accuracy are demonstrated by systematic experiments with different corpora. Run-time measurements show the efficiency gains of our approximation techniques (Section 4.5).

4.2. Related Work

4.2.1. Semantic Relatedness of Words

The classical approaches for quantifying the relatedness of words (common nouns, verbs, etc.) make use of the WordNet thesaurus, by measuring the token overlap of the glosses describing the concepts (Lesk) or by means of paths in the concept taxonomy or other lexical relations (Resnik, Wu & Palmer). Budanitsky and Hirst [BH06] give an overview of all these measures.

More recently, Gabrilovich and Markovitch [GM07] and Radinsky et al. [RAGM11] have exploited Wikipedia (and Zesch et al. Wiktionary [ZMG08]) to compute word relatedness. These methods represent words by vectors of the concepts they are used to describe, i. e. a word is represented by all the articles it occurs in. Then they use vector-space measures (e.g., cosine similarity) for relatedness. A powerful variant of this approach, proposed by Hassan and Mihalcea [HM11], is to represent words as vectors of co-occurring Wikipedia entities. However, the method crucially depends on the availability of a rich link structure like that of Wikipedia.

4.2.2. Semantic Relatedness of Entities

Ponzetto and Strube [PS07] applied previously WordNet-based measures to Wikipedia, showing that some work better on the larger though less rigorous collection of entities and classes. Milne and Witten [MW08a] used the link structure in Wikipedia to model entity relatedness. Links within Wikipedia refer to canonical entities; thus, the source-target pairs of links can be used as the basis for a relatedness measure. So far, the MW measure has achieved the best results on quantifying entity relatedness. The actual formula and details of the measure are given in Equation 3.7 on page 26.

4.3. Keyphrase-Based Relatedness

4.3.1. Definitions

All measures of semantic relatedness between entities described in this section are based on (multi-word) key*phrases* and their constituting (single-word) key*words*. When we do not need to distinguish between (multi-word) key*phrases* and (single-word) key*words*, we speak of key*terms*.

Example. The keyphrase English rock guitarist is represented by the set of keywords {English, rock, guitarist}.

We gathered keyterms from the Wikipedia pages of each entity:

- *link anchors* of both internal and external outgoing links
- *titles of citations*
- *names of categories*

The choice of Wikipedia is merely for convenience; keyterms can also be mined from other textual descriptions of entities, such as homepages of singers or scientists, or pages about songs in online communities. In these cases, we can use similar heuristics like focusing on link anchors, headings, etc., or alternatively extract all noun phrases. The latter will pick up noise, but our weighting scheme, described next, will keep the noise level low.

All keyterms are weighted by a combination of Inverse Document Frequency (IDF), which is global and entity-independent, and Mutual Information (MI). This is very similar to how the keywords are weighted in Chapter 3, with the difference that here not only keywords but also keyphrases are weighted, and that we are using a different variant of MI for keyphrases.

IDF weights capture a global notion of how important a keyterm is. The weights are computed using the standard formula, see Equation 3.5 on page 26. The document frequency *df* is the number of entities that have the phrase among their keyphrases (for keyphrase IDF) or have at least one keyphrase that contains the keyword as token (for keyword IDF).

MI weights capture the notion of how important a keyterm is with respect to a given entity. It is based on the relative entropy between the events of an entity-keyterm pair occurring individually or jointly. For its computation, we define the superdocument of an entity to be the union of its keyphrases with the keyphrases of all entities linking to it. The joint occurrence of a keyterm and an entity is the occurrence of the keyterm in the superdocument. In Chapter 3 we used normalized pointwise mutual information (NPMI, see Equation 3.1 on page 25) for the entity-keyword weights. We also experimented with NPMI for entity-keyphrase weights, however we found that this variant of normalized MI works better for KORE:

$$\mu(E,T) = 2 \cdot \frac{H(E) + H(T) - H(E,T)}{H(E) + H(T)}, \tag{4.1}$$

where $H(E)$ and $H(T)$ are the marginal entropies of the entity and term, respectively, and $H(E,T)$ is the joint entropy. We compute the μ weights for all entity-keyphrase pairs and NPMI weights for all entity-keyword pairs. Note that all weights are with respect to the keyphrase space, not the original texts they were mined from.

4.3.2. Keyterm Cosine Relatedness

If no links are available upon which the entity relatedness can be computed, a baseline method is as follows. We compare two entities (e, f) represented by keyterm sets $T(e)$ and $T(f)$ by casting the keyterms into a weighted vector. This works for both keyphrases and keywords. If the terms are phrasal in nature, they can either be treated as a single unit in the vector or tokenized before constructing the vector. In our experiments, the vectors are filled with the MI weights of the keyterms. The relatedness between these vectors can be calculated using cosine similarity.

Let (e, f) denote a pair of entities with keyterm vectors V_e and V_f, respectively. The

keyterm relatedness is:

$$\mathrm{ktr}(e, f) = \frac{V_e \cdot V_f}{\|V_e\| \, \|V_f\|} \tag{4.2}$$

If keywords are derived from keyphrases by tokenization, their weights should take the phrase weights into account. We simply multiply the weights of the words with the average weight of the phrases from which the words are taken.

4.3.3. Keyphrase Overlap Relatedness

Concepts are often represented as phrases (at least in the English language, but to a large extent also in other languages); so relatedness measures should consider multi-word keyphrases instead of single keywords only. However, when entities are represented by keyphrases, it is crucial to consider partial matches rather than focusing solely on exact-match comparisons. For example, "English rock guitarist" should be more similar to "English guitarist" than to "German President". To solve this issue, the relatedness measure needs to match keyphrases based on overlap and needs to take into account both keyphrase and keyword weights.

Let (e, f) denote a pair of entities with keyphrase sets $P_e = \{p_1, p_2, \ldots\}$ and $P_f = \{q_1, q_2, \ldots\}$, respectively. A phrase is identified with the set of constituent terms $p_i := \{w_1, w_2, \ldots\}$. We associate with every keyword w a weight $\gamma_e(w)$ with respect to the entity e.

In the following, let $(p, q) \in P_e \times P_f$ denote a pair of phrases. This allows us to introduce a measure of *phrase overlap* (PO) for the pair (p, q), where its constituent words are weighted with respect to the entities (e, f), given by the weighted Jaccard similarity of the keywords:

$$\mathrm{PO}(p, q) = \frac{\sum_{w \in p \cap q} \min\{\gamma_e(w), \gamma_f(w)\}}{\sum_{w \in p \cup q} \max\{\gamma_e(w), \gamma_f(w)\}} \tag{4.3}$$

We use PO to measure the *keyphrase overlap relatedness* (KORE) of a pair of entities (e, f) based on the sets of their associated phrases P_e and P_f. The measure captures the spirit of weighted Jaccard similarity while at the same time allowing keyphrases to contribute to the measure based on their overlap with *all* keyphrases of the other entity. Keyphrases are weighted with respect to the entities; recall that these weights φ_e are different from the keyword weights γ_e. We define the keyphrase overlap relatedness measure:

$$\mathrm{KORE}(e, f) = \frac{\sum_{p \in P_e, q \in P_f} \mathrm{PO}(p, q)^2 \cdot \min\{\varphi_e(p), \varphi_f(q)\}}{\sum_{p \in P_e} \varphi_e(p) + \sum_{q \in P_f} \varphi_f(q)} \tag{4.4}$$

The factor $\mathrm{PO}(p, q)$ is squared to penalize phrases which do not fully overlap. Our experiments have shown that using MI weights for keyphrases (φ_e) and IDF weights for keywords (γ_e) works best. The numerator re-weights the phrase overlap PO with the lesser weight of the two phrases that match. The denominator sums up all the keyphrase weights of each entity to normalize the numerator. Notice that we do not normalize by maximum possible intersection, which would be the sum over the full

Cartesian product $\sum_{p \in P_e, q \in P_f} \max\{\varphi_e(p), \varphi_f(q)\}$. Using this for normalization would unduly penalize popular entities with a larger keyphrase set, as the Cartesian product grows much faster than the intersection.

4.4. Efficient Computation

4.4.1. Need for Speed

Computing similarities between a set of n objects is an important step in many applications, not only in entity disambiguation, but also in clustering or coreference resolution (record linkage). The KORE measure is relevant for all these tasks, assuming that the input data is a keyphrase set and partial matching improves the quality of the similarity measure.

The naive approach is to compute all $\binom{n}{2}$ pairwise similarities. This quickly becomes a bottleneck for large n. Even if the task is parallelizable, overcoming the $O(n^2)$ complexity is necessary to achieve good scalability, especially for interactive tasks such as on-the-fly NED (e.g., for news streams) or high-throughput tasks such as NED on an entire corpus (e.g., one day's social-media postings).

Precomputing the similarities is prohibitive with regard to both space and time. Today's knowledge bases contain millions of entities. The quadratic space and time complexity would lead to more than 10^{12} pairs. Even if we merely needed a single byte to store a pair's relatedness value, we would consume Terabytes of storage, and potentially much more. This is not practically viable. So the goal is to avoid the quadratic effect and devise appropriate pre-processing that facilitates fast relatedness computations on the fly. To this end, we have developed hash-based approximation techniques, using min-hash sketches [BCFM98] and the method of locality-sensitive hashing (LSH) [IM98, GIM99]. LSH has been used in numerous applications, including clustering in the context of NLP tasks (e.g., [RPH05]). To the best of our knowledge, no prior work has considered such hashing techniques for entity relatedness and NED tasks.

4.4.2. Two-Stage Hashing

Entities are represented as sets of (weighted) keyphrases, which in turn are represented as sets of (weighted) keywords. Thus, keyphrases should not be compared atomically. "President of the United States" should be more similar to "United States President" than to "German President".

If we want to use LSH to speed up the computation, partial matches among phrases must be taken into consideration. As a solution, we propose the following two-stage hashing scheme:

1. **Grouping highly similar keyphrases.** Entities are then represented as a set of identifiers of such keyphrase buckets without losing the notion of partial phrase matches.

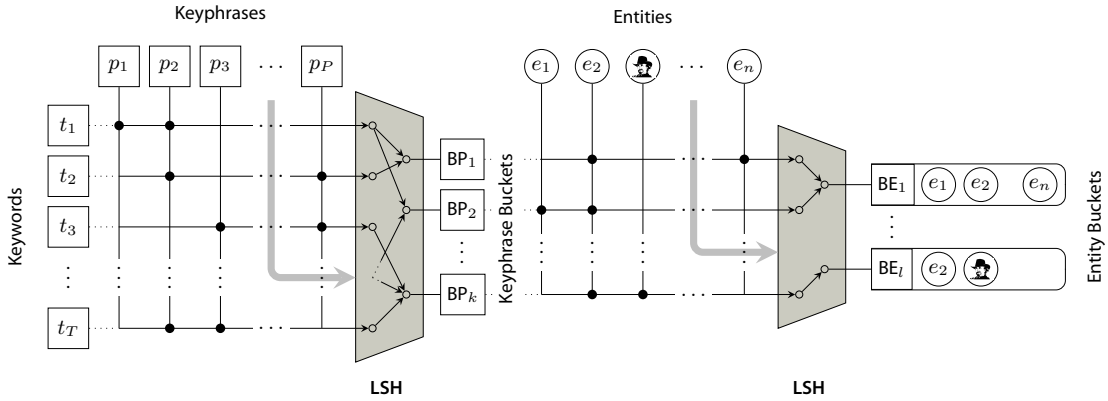

Figure 4.1.: Overview of the two-stage hashing technique

2. **Grouping related entities** together that have sufficiently high overlap in terms of keyphrase bucket identifiers. The exact pairwise relatedness score can then be computed within a single entity-group, reducing the complexity of $O(n^2)$ to linear in the number of entities if the hashing techniques achieve near-uniform distribution of keyphrases (step 1) and entities (step 2). In principle, skewed distributions could still incur bottlenecks; however, our experiments with millions of entities did not exhibit such adverse effects in practice.

In the following, we explain the two steps in more detail. Figure 4.1 gives a pictorial overview of our technique.

Grouping Highly Similar Keyphrases

The goal of this step is to group keyphrases that are near duplicates in the sense that their Jaccard similarity is very high. As a compact representation of the words in a keyphrase, we first compute min-hash sketches for each phrase. This form of sketches allows us to approximate (as an unbiased estimator) the Jaccard similarity between sets (of words, in our case). To avoid the pair-wise comparison of all keyphrases, we apply LSH on the min-hash sketches of the phrases.

In detail, we first represent a keyphrase by a set of words and assign each word a globally unique id. For each keyphrase (the average length in our knowledge base is 2.5 words) we sample 4 times by min-hashing, thus representing each keyphrase by a vector of length 4. For LSH, we divide the vectors in two bands of length two, and combine the two ids in each band by summing up their ids, losing the order among them. Each keyphrase is finally represented by the two bucket ids (one per band) it was hashed to, combining near duplicate keyphrases. Note that we do not perform this stage-one hashing to reduce the dimensionality of the keyphrase space, but to capture the notion of partial overlapping keyphrases in KORE and to improve the second stage of grouping entities.

Grouping Related Entities

While the first stage above is precomputed for all entities and their keyphrases (with linear complexity), the second stage, described now, is executed at task run-time with a set of entities as input – for NED these are all candidate entities in a document. The algorithm first retrieves the sets of min-hash sketches for all input entities, pre-computed from the phrase-bucket ids output by the previous stage. These ids are now the input to a second stage of LSH. Algorithmically, this is fairly analogous to the first-stage technique, but the inputs are different from the first stage.

The exact KORE measure between two input entities is computed only if they share at least one hash bucket. Otherwise, we assume the entity relatedness is sufficiently low to consider the entities as unrelated. For the 3 million entities in our knowledge base we can fit the sketches for $\text{KORE}_{\text{LSH-G}}$ into main memory, merely requiring about 2 GBytes.

For $\text{KORE}_{\text{LSH-G}}$, a reasonably fast approximation of KORE which has nearly the same quality as KORE, we partition the sketched phrase-bucket id vectors into 200 bands of size 1. For $\text{KORE}_{\text{LSH-F}}$, a really fast approximation of KORE which degrades the approximation quality a bit, we use 1,000 bands of size 2, again combining the sketched ids by summing them up before hashing. $\text{KORE}_{\text{LSH-G}}$ is geared towards high recall so that the actual computation is executed between all somewhat related entities, but filters out noise. The speed-up is not so high, but the quality is close to exact KORE. Sometimes, quality is even improved as noisy candidates are removed. $\text{KORE}_{\text{LSH-F}}$ is geared towards higher precision with bands of size two, allowing LSH to prune even more entity pairs, speeding up the subsequent computation of the semantic relatedness due to fewer comparisons.

As we create the LSH hashtables dynamically during runtime for a given input set of entities, using more bands of larger size means that we need longer min-hash sketches – for $\text{KORE}_{\text{LSH-F}}$ a total of 2,000 per entity. This increases the time for constructing the hashtables. However, in our experiments the time difference in creating the candidate pairs using LSH for $\text{KORE}_{\text{LSH-G}}$ and $\text{KORE}_{\text{LSH-F}}$ is low; so the overall runtime is improved, especially for large sets of entities.

4.5. Evaluation

4.5.1. Dataset

Existing datasets to evaluate semantic relatedness quantify the relatedness between surface forms of words, not between entities registered in a knowledge base, which makes these datasets unsuitable for our task. We have created a new dataset to evaluate the quality of different measures for the relatedness between entities. A conceivable way to create such data would try to determine numeric scores for the degrees of relatedness (e.g., [FGM+02]). However, for most tasks a good ranking among the candidates is important, not numeric scores. Moreover, semantic relatedness is difficult to judge by humans in an absolute manner. For these reasons, we settled for relative ranking judgments and use crowdsourcing to average out the subjectivity of such judgments.

Seed	Related Entity (Rank)
Apple Inc.	Steve Jobs (1), Steve Wozniak (2) ... NeXT (10), Safari (web browser) (11) ... Ford Motor Company (20)
Johnny Depp	Pirates of the Caribbean (1), Jack Sparrow (2) ... Into the Great Wide Open (10) ... Mad Love (20)
GTA IV	Niko Bellic (1), Liberty City (2) ... New York (10), Bosnian War (11) ... Mothers Against Drunk Driving (20)
The Sopranos	Tony Soprano (1), David Chase (2) ... Golden Globe Award (10), The Kinks (11) ... Big Love (20)
Chuck Norris	Chuck Norris facts (1), Aaron Norris (2), ... Northrop Corporation (10) ... Priscilla Presley (20)

Table 4.1.: Example seed entities and gold-standard ranks of related entities

We selected a set of 20 entities from YAGO2 [HSBW13] from 4 different domains: IT companies, Hollywood celebrities, video games, and television series. For each of the 20 seed entities we selected 20 candidates from the set of entities linked to by the seed's Wikipedia article. Using entities from Wikipedia articles allows us to capture the candidates in the context which relates them to their seed entity. We tried to select the candidates in such a way that their semantic relatedness to the seed entity should be clearly distinguishable among each other, and included highly related as well as only remotely related entities in the set of candidates.

Example. For the entity *Chuck Norris* we have *United States Air Force* and *Chun Kuk Do* as candidates, described by the following context: "After serving in the *United States Air Force*, he began his rise to fame as a martial artist and has since founded his own school, *Chun Kuk Do*". This contextual information is given to the human judges, together with links to the Wikipedia articles for more detailed information. The crowdsourcing worker is then asked to rank *United States Air Force* versus *Chun Kuk Do* in terms of their relatedness to *Chuck Norris*. This included a "They are about the same" option.

The gold-standard ranking of the 20 candidate entities per seed is created as follows:

- All possible comparisons of the 20 candidates with respect to their seed are created (190 in total).

Domain	KWCS	KPCS	MW	KORE	$\text{KORE}_{\text{LSH-G}}$	$\text{KORE}_{\text{LSH-F}}$
IT Companies	0.660	0.759	0.721	0.764	0.586	0.208
Hollywood Celebrities	0.722	0.715	0.667	0.646	0.647	0.522
Television Series	0.577	0.599	0.628	0.519	0.538	0.426
Video Games	0.604	0.760	0.431	0.780	0.722	0.499
Chuck Norris	0.481	0.498	0.571	0.585	0.585	0.653
Average (11 entities with $\leq$ 500 links)	**0.597**	**0.637**	**0.513**	**0.640**	**0.625**	**0.496**
Average (all 21 entities)	**0.633**	**0.698**	**0.610**	**0.673**	**0.621**	**0.425**

Table 4.2.: Spearman correlation of relatedness measures with human ranking

- 10 of the 190 comparison pairs are created as gold standard by the authors. The gold standard is used to compute the confidence in the human judges and to exclude them if they get too many wrong.
- For of the remaining 180 comparison pairs, 5 distinct judges are asked which of the given two entities is more related to the seed entity.
- All comparisons are aggregated by Crowdflower into a single confidence that one entity is more (or equally) related to the seed.
- The 20 candidate entities are then ranked by these confidence values as described by Coppersmith et al. [CFR10], which has shown that the number of times an entity wins and the weights of the wins gives a good ranking.

The final output is a set of 20 ranked lists consisting of 20 entities each, against which we compare the rankings generated by the semantic relatedness measures. For the previous example, *Chun Kuk Do* was ranked 5 and *United States Air Force* 10, reflecting the stronger relatedness of a person to a school of martial arts initiated by him than to an organization he worked for. More examples of entity pairs with their crowdsourcing-derived ranks are given in Table 4.1.

All seed entities were chosen to be among the most popular individuals in their respective domain (*Apple* as IT company, *Johnny Depp* as Hollywood celebrity, *Grand Theft Auto IV* (GTA IV) as video game, and *The Sopranos* as television series). We also added *Chuck Norris* consisting only of *Chuck Norris* (a singleton set). The dataset, which contains a total of 441 entities (20 candidates for 21 seeds), is available at `http://www.mpi-inf.mpg.de/yago-naga/aida`.

4.5.2. Experimental Results

Experimental results for this dataset are given in Table 4.2. The numbers are the Spearman correlation between the gold-standard ranking and the automatically generated rankings by the relatedness measures.

The MW measure is defined in Equation 3.7 on page 26. For all keyterm-based measures, the keyphrases are weighted using μ as defined in Equation 4.1 on page 43, and the keywords are weighted using IDF as defined in Equation 3.5 on page 26. KWCS

is cosine similarity between the keyword vectors derived from the keyphrases, KPCS is the cosine similarity on keyphrase vectors. KORE is the keyphrase overlap relatedness, with two configurations of LSH approximations. All measures are detailed in Section 4.3.

We see that all the measures using keyphrases – and do not depend on links – work better than the link-based MW, with KPCS performing best. The difference becomes even more obvious when we average the Spearman correlation not over all entities but only over those with less than 500 incoming links in Wikipedia (relatively "link-poor" entities) – here KORE works best. The small size of the dataset did not allow meaningful significance tests, though. We emphasize the point that even without links, the keyphrase-based measures perform at least as good as the MW measure. So we no longer depend on the rich Wikipedia link structure without losing quality, and we are well geared for dealing with truly long-tail entities that have very few or no links at all.

4.6. Experiments on Named Entity Disambiguation

For the NED experiments with different relatedness measures, we modified the AIDA framework presented in Chapter 3 to incorporate the KORE measure.

4.6.1. Datasets

We conducted experiments on three datasets:

CoNLL-YAGO The CoNLL-YAGO data set introduced in Section 3.6.1. All numbers are from runs on the original 231 withheld test documents.

KORE50 We hand-crafted 50 difficult test sentences from five domains: celebrities, music, business, sports, and politics. The sentences were formulated according to a set of criteria:

- Short context: on average, only 14 words per sentence.
- High density of entity mentions: 148 mentions in 50 sentences, nearly 3 per sentence on average and a ratio of 20% of words. The mention-to-word ratio in CoNLL-YAGO is 12%.
- Highly ambiguous mentions: 631 candidate entities per mention, on average. In contrast, CoNLL-YAGO has only 27.
- Sentences contain long-tail entities with very few incoming links.

Examples are persons referred to by their first name only instead of last name or full name, or football clubs and players that are not well known outside their own home states. It is available at `http://www.mpi-inf.mpg.de/yago-naga/aida/`.

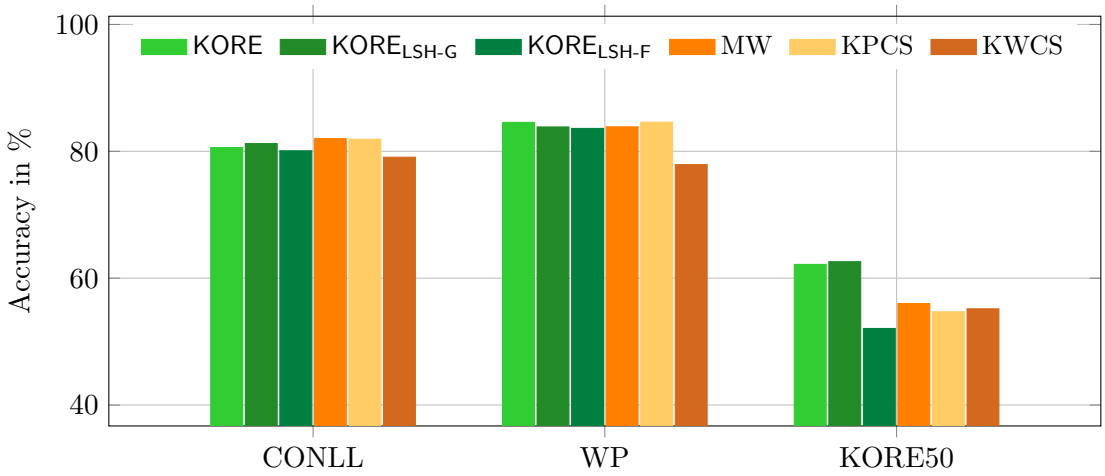

Figure 4.2.: Disambiguation accuracy on all datasets.

WP This dataset is a slice of Wikipedia with similar characteristics as KORE50. It contains all articles in categories ending with "heavy metal musical groups". Each article is split into sentences, and only sentences containing at least 3 named entities as link-anchor texts are kept. The link targets of these mentions provides us with ground-truth mappings. As a stress test, to make disambiguation more difficult, we replaced all occurrences of person names with the family name only (e. g. "Johnny Cash" replaced by "Cash"). In the same vein, we disabled the popularity-based prior for name-entity pairs for this dataset in all methods under comparison. We gathered 2,019 sentences this way, with an average length of 52 words. (The automatic sentence boundary detection did not always work properly, so some output segments were longer than expected). Each sentence has an average of 5 mentions with 64 candidates.

4.6.2. Experimental Results

We measured NED accuracy as the fraction of mentions that were correctly disambiguated, ignoring all mentions that do not have a candidate entity at all, following the evaluation methodology of Section 3.6.1. An overview of the results is given in Figure 4.2, detailed results are shown in Table 4.3: micro-averaged numbers aggregate over all mentions, macro-averaged numbers aggregate over all documents of a dataset. We also broke down the per-mention results by the number of Wikipedia inlinks of the mention's true entity. Link-averaged numbers are the macro-average over the groups of mentions with the same number of inlinks.

Overall, the KORE-based NED performed about as well as the original AIDA method, which uses the MW measure based on the rich link structure of Wikipedia. MW performs better on the CoNLL-YAGO dataset, KORE performs better on the KORE50 and WP datasets. A paired t-test shows that there is no significant difference between MW and KORE on CoNLL-YAGO or KORE50 for macro-average accuracy; however, KORE is

Dataset	Evaluation	KWCS	KPCS	MW	KORE	KORE$_{\text{LSH-G}}$	KORE$_{\text{LSH-F}}$
CoNLL-YAGO	Micro Avg.	79.53%	82.18%	**82.31%**	80.71%	81.76%	81.18%
	Macro Avg.	79.07%	81.91%	**82.00%**	80.59%	81.22%	80.08%
	Link Avg.	79.28%	**82.31%**	81.34%	80.21%	81.80%	80.80%
WP	Micro Avg.	83.29%	85.30%	84.73%	**85.36%**	84.68%	84.50%
	Macro Avg.	82.50%	**84.59%***	83.86%	84.56%*	83.84%	83.61%
	Link Avg.	77.90%	80.49%	**82.45%**	80.12%	80.64%	80.36%
	Link Avg. $\leq$ 500 links	84.80%	86.80%	85.59%	**87.15%**	86.43%	86.29%
	Link Avg. $\leq$ 50 links	83.39%	85.19%*	83.87%	**85.52%***	84.56%	84.90%*
	Link Avg. $\leq$ 5 links	80.89%	81.10%	80.89%	**82.40%***	81.75%	81.38%
KORE50	Micro Avg.	55.56%	55.56%	57.64%	63.89%	**64.58%**	53.19%
	Macro Avg.	55.17%	54.70%	56.00%	62.17%	**62.60%**	52.07%
	Link Avg.	62.11%	61.79%	63.21%	70.75%*	**71.70%***	58.58%

Table 4.3.: Disambiguation accuracy (best method per row is in boldface; statistically significant differences from MW are marked with an asterisk)

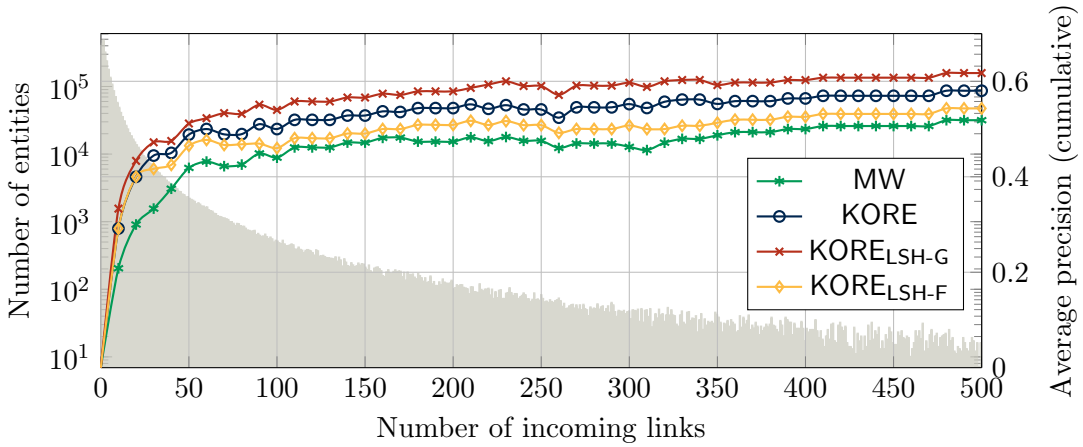

Figure 4.3.: Accuracy of relatedness measures on entities with fewer than 500 incoming links on the KORE50 dataset

significantly better on the WP dataset (p-value < 0.01 in a paired t-test). With link-averaged accuracy, KORE and KORE$_{\text{LSH-G}}$ also perform significantly better than MW on KORE50 (p-value < 0.01). On the WP dataset, Table 4.3 additionally gives the average accuracy for mentions whose true entities have few links, to emphasize the accuracy of KORE for long-tail entities.

Figure 4.3 shows a detailed comparison of MW and the KORE variations on the KORE50 dataset. The accuracy at each point x on the x-axis is the average accuracy of all entities with up to x links. As expected, KORE works better for really link-poor entities, because it builds on keyphrases instead of links. With more links, the performance of KORE is still significantly better, but the difference between KORE and MW becomes smaller. Performing well on link-poor entities is really important considering the fact that entities with ≤ 50 incoming links make up more than 80% of Wikipedia.

4.6.3. Interesting Examples

As found above, KORE works better than MW for **long-tail entities**. One example from Document 1199 from the CoNLL-YAGO dataset demonstrates this, announcing results of German soccer matches along with the scorers.

> Karlsruhe 3 (Reich, 29th, Carl 44th, Dundee 69th) Freiburg 0.

Example 4.9

The MW measure is not able to capture the coherence between the club `Karlsruher SC` and the player `Burkhard Reich`, a long-term player of theirs. Instead it prefers `Marco Reich`, another German footballer who is more prominent and thus more related in general to the other footballers and clubs in the document. KORE is able to capture this fine-grained coherence and correctly disambiguates "Reich".

Unfortunately, KORE also leads to wrong results sometimes. Take the following sentence from Document 1294 about Indian politics as example.

> The BJP backs a hardline Hindu campaign to build a temple at the site of the mosque, which Hindus believe was the birthplace of the Lord Rama.

Example 4.10

KORE assigns the comic character `Rama (comics)` to "Rama" instead of the Hindu god `Rama`. The comic character has some highly salient keyphrases for Hindu religion, as it is based on the actual god, however it is clearly not related to the other entities in the text. For cases such as this, the LSH preprocessing can be beneficial, filtering out noisy relatedness edges between only marginally related entities. In this example, when KORE is run with LSH preprocessing, the comic character is actually filtered out, leading to a correct result.

The LSH filtering is sometimes too aggressive, though, for example in Document 1302 covering a plane crash.

> He said the cargo flight [...] was due to stop at Stephenville for refueling before going to Shannon [...].

Example 4.11

In this example, the correct entities `Stephenville International Airport` and `Shannon Airport` are filtered out by the LSH preprocessing.

4.6.4. Efficiency

To evaluate the efficiency, we compared the running time of AIDA with the link-based MW measure against AIDA with the KORE measure, both in its exact form and with the LSH approximation for speed-up. We tested on the full CoNLL-YAGO collection.

Method	Comparisons			Running time (s)		
	mean	stddev	0.9-quantile	mean	stddev	0.9-quantile
MW	898,253	3,578,524	1,594,226	5.335	27.357	9.738
KORE	898,253	3,578,524	1,594,226	1.884	6.592	3.519
$\text{KORE}_{\text{LSH-G}}$	315,302	1,306,293	548,561	1.285	3.925	2.308
$\text{KORE}_{\text{LSH-F}}$	60,580	301,712	87,240	0.413	1.557	0.586

Table 4.4.: Results of timing experiments

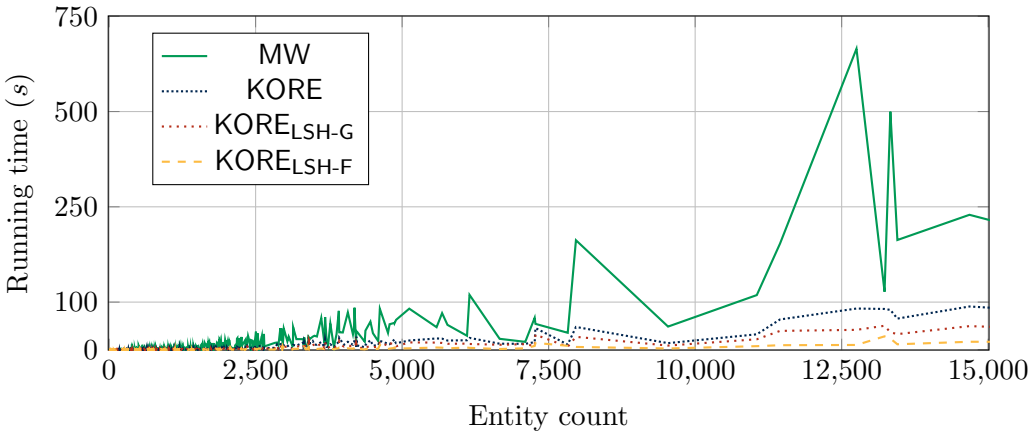

Figure 4.4.: Running time

AIDA computes coherence weights only between candidate entities that are candidates for at least two different mentions. If they share only one mention, they are mutually exclusive candidates anyway.

Thus, the number of comparisons (and the runtime) is still in the order of n^2, but it does not increase monotonically with the number of candidate entities for a document. Figure 4.4 shows the documents on the x-axis and the measured run-times for computing the NED on a document on the y-axis. The documents on the x axis are sorted in ascending order by the respective number of candidate entities ("largest" documents are thus rightmost). Figure 4.5 shows the number of computations instead.

We observe that the exact version of KORE is already much faster than MW. This is because MW is based on links, and some popular entities have a large number of incoming links (sometimes above 100,000, e.g., for places). In such cases, the intersection of bitvectors that AIDA uses to represent inlinks is very time-consuming. The number of keyphrases of popular entities does not nearly grow at the same rate, as it is bounded by the length of the document describing the entity. In contrast, incoming links can grow independently of the document size.

Table 4.4 gives detailed figures for the average run-time, its standard deviation, and the 90%-quantile of the run-time distribution (over the documents), and the same mea-

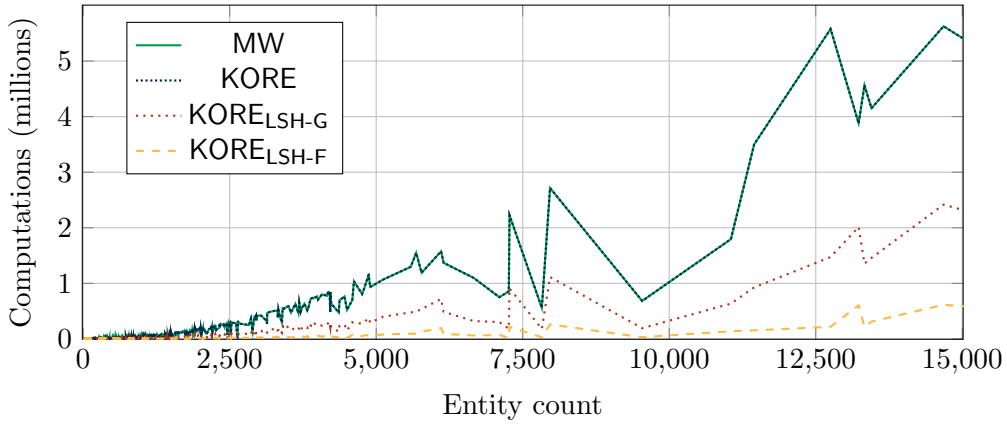

Figure 4.5.: Number of computations

sures also for the number of entity-pair relatedness comparisons. KORE and MW perform the same number of comparisons, but the run-times differ significantly. The accelerated variants of KORE reduce the number of comparisons by a large margin and reduce the run-time further. In particular, the KORE$_{LSH-F}$ variant reduces the number of comparisons and the average run-time by an order of magnitude. For the 90%-quantile, which is relevant for being fast on the hardest cases, KORE$_{LSH-F}$ gains almost a factor of 20.

4.7. Summary

The keyphrase overlap relatedness measure KORE provides a high-quality notion of semantic relatedness of entities. In our experiments, KORE consistently performed at least as good as the best state-of-the-art methods, and often significantly better. Most importantly, KORE eliminates the reliance on explicit linkage between entities, and is thus geared for dealing with long-tail entities with few or no links. This makes the measure usable also on knowledge bases that are not derived from Wikipedia and usually don't come with explicitly linked entities. The approximate computation by the two-stage hashing method speeds up KORE and makes it suitable for online tasks that require interactive responses.

The KORE measure is available as part of the AIDA framework at `http://www.mpi-inf.mpg.de/yago-naga/aida/`. The KORE entity relatedness dataset containing 21 entities with relatedness judgments, as well as KORE50 dataset for entity disambiguation can be downloaded at the same URL.

5. Discovering Emerging Entities

Knowledge bases (KBs) can never be complete due to the dynamics of the ever-changing world: new companies are formed every day, new songs are composed every minute and become of interest for addition to a KB. To keep up with the real world's entities, the KB maintenance process needs to continuously discover newly emerging entities in news and other Web streams. In this chapter we focus on the most difficult case where the names of new entities are ambiguous. This raises the technical problem to decide whether an observed name refers to a known entity or represents a new entity. This chapter presents a method to solve this problem with high accuracy. It is based on a new model of measuring the confidence of mapping an ambiguous mention to an existing entity, and a new model of representing a new entity with the same ambiguous name as a set of weighted keyphrases. The method can handle both Wikipedia-derived entities that typically constitute the bulk of large KBs as well as entities that exist only in other Web sources such as online communities about music or movies. Experiments show that our entity discovery method outperforms previous methods for coping with out-of-KB entities (called *unlinkable* in entity linking).

5.1. Introduction

Our world is highly dynamic; so no KB will ever be complete. New songs are composed every day, new movies are released, new companies are founded, there are new weddings, sports matches, and disasters and formerly unimportant people become noteworthy and should be added to a knowledge base. We refer to such entities as *emerging entities* (EEs) or out-of-KB entities.

5.1.1. Problem and Prior Work

This chapter focuses on how NED methods can cope with this incompleteness, both during the disambiguation process itself as well as to extend the knowledge base with new entities. The problem is particularly challenging because new entities may appear under the same names as existing ones. For example, when hurricane "Sandy" occurred, several singers, cities, and other entities with the name "Sandy" already existed in Wikipedia (and thus also in the big knowledge bases). When faced with the first news about the hurricane, a smart NED method should distinguish the emerging entity from the existing ones, although the new entity may have very sparse context.

Prior methods for NED addressed this problem by thresholding on the scores they computed for mapping a given mention to different candidate entities. If none of the existing entities can be chosen with a score above a specified threshold, then the mention

is marked as an emerging entity (see, e.g., [KSRC09, RRDA11]). The approach has several shortcomings, though. First, its empirical quality is not that good, at least not in difficult situations. Hachey et al. [HRN+13] discussed this aspect for the TAC 2010 entity-linking benchmark. Second, score thresholds of this kind are hard to tune in a robust manner. The same threshold hardly works for different kinds of contents; so these methods may need frequent re-tuning and appropriate training data – not easily feasible in real applications. Third, deciding for each mention separately whether it is an in-KB entity or an EE based on a global threshold comes with the risk that local decisions adversely affect the outcome for all mentions.

Example. Consider the first news about the disclosure of the Prism program by Edward Snowden. Both "Prism" and "Snowden" are ambiguous names, now being primarily used for the two emerging entities `PRISM_(program)` and `Edward_Snowden`. Suppose an NED method is fed with the input sentence "Washington's program Prism was revealed by the whistleblower Snowden." from one of the early articles on this topic. The NED method needs to decide that "Prism" does not refer to the band or the TV network going by the same name, and should instead be mapped to an EE. However, "Snowden" is another mention for which the NED method has low mapping scores. With a thresholding decision per mention (like in state-of-the-art work), it may happen that "Prism" is below the threshold while "Snowden" is still above. Then, the NED method may choose to map "Snowden" to the city of Snowden in the state Washington, and map "Washington" to the this state, for coherence between these two entities. Thus, the poor thresholding decision for "Snowden" adversely affects the NED output for the seemingly easy mention "Washington". Interestingly, it is exactly the best NED methods, with joint inference over all mentions, that are most penalized by such effects.

Another difficulty is that there could be multiple EEs, all out-of-KB, with the same name. For example, the singer Katy Perry has announced a new album, "Prism", which should not be confused with the surveillance program Prism. To deal with such situations, we need to consider the entire life-cycle of a knowledge base. Once we have identified a new EE, it should be added to the knowledge base in a representation that is strong enough to distinguish it from further EEs with the same name that will possibly be encountered. At some point, after having observed an EE sufficiently often, it should be promoted, with assistance by a human editor, to a canonicalized entity in standard form. Both the separation from other EEs and the canonicalization step need contextual information about the EE. In our approach, we harness salient phrases from the texts where the EE is observed.

5.1.2. Our Approach

Our goal is to develop a principled approach for the problem of NED with emerging entities, NED-EE for short. This comprises both distinguishing an EE from existing entities with the same name, and distinguishing an EE from an equally named other EE that is observed later.

Our approach builds on NED methods that use *characteristic (weighted) keyphrases* (or just keywords) of entities as part of their feature space. Wikipedia category names of

an entity or href anchor texts that link to the entity are simple but highly effective forms of such keyphrases. NED methods along these lines include AIDA (Chapter 3), Illinois Wikifier [RRDA11], Kulkarni et al. [KSRC09] among others. The NED-EE method developed here works with any of these NED tools. Our NED-EE method is based on two techniques:

- assessing the confidence of the NED method's mapping of mentions to in-KB entities, and
- enriching a possible EE with a keyphrase representation.

Equipped with these two assets, we can then extend the NED method by adding the potential EE to the method's space of candidate entities, using keyphrase features.

For assessing the confidence, we devise several techniques, based on perturbing the mention-entity space of the NED method. We also consider transforming the NED mapping scores into normalized confidence values. The confidence is then used to decide whether a new EE should be considered or not. Note that this is quite different from prior work, which would drop a mention in a low-confidence situation (i.e., declaring it unlinkable or mapping it to a global "nil" node) although candidate entities might exist for it. Our method never drops any mentions, but instead considers an EE as an *additional candidate*.

For constructing a keyphrase-based representation of EEs, we need to identify phrases that are characteristic for the EE and uncharacteristic for the existing entities with the same name. This cannot be done directly using only the text where we observe the mention. Our approach first builds a global set of keyphrases that frequently co-occur with *any* entity of the ambiguous name. We do this by searching the Web for the name and extracting contexts and phrases from the retrieved matches. With appropriate weighting, we obtain a global representation of *all* entities with the given name, including the EE. Since we already have keyphrase features (or at least keyword features) for the in-KB entities, we can now compute a *model difference* between the global model and the union of all in-KB models, yielding the (weighted) keyphrases that are salient for the EE.

Our key contributions in this chapter are the following:

- New building blocks for robustly coping with emerging entities (EEs) in NED: confidence assessment for mention-entity mappings (Section 5.4), and a keyphrase model for the emerging entities themselves (Section 5.5).
- A principled method, referred to as NED-EE, that extends a family of NED tools by EEs as additional candidate entities and makes judicious decisions about mapping mentions to existing entities or EEs (Section 5.6).
- Experiments with a news corpus that demonstrate the viability of our approach and the superior NED quality compared to state-of-the-art NED methods (Section 5.7).

5.2. Related Work

Discovering Emerging Entities Nearly all of the NED methods introduced in Section 2.2.2 already address the problem of out-of-KB entities. Most of them use a threshold on the best scoring entity. The disambiguation method by Kulkarni [KSRC09] uses a trained threshold to identify EEs. Ratinov et al. [RRDA11] refine the EE discovery by specifically training a separate component that decides whether or not to annotate a mention as EE, using an SVM on results of the initial disambiguation step. The final decision whether to annotate is again taken by thresholding on the SVM score. Other works like TagMe [FS12, CFC13] either use a simple threshold or, like AIDA (Chapter 3) ignore the EE problem completely, focusing only on the disambiguation part.

The key distinction between these works and our work is the way the discovery of non-existent entities is handled. Previous work did not do any modeling of the unknown entities, but rather based the decision on the absence of good context for the existing entities.

Understanding New Names Other work addresses the discovery of emerging entities from a different angle. Both the work by Nakashole et al. [NTW13] and by Lin et al. [LME12] assume that existing names always refer to existing entities, and that only new names are of interest. Their focus is then on trying to assign a type to the emerging entity as a first step to make it useful. For example, the name "Edward Snowden" was previously unknown (to Wikipedia), and would be assigned the types `person`, `computer specialist`, and `whistleblower`, but the methods would ignore the name "Prism" as there are existing entities in Wikipedia going by that name. Our work makes the same assumption that new names refer to new entities, but also tries to discover new entities going by the name of existing entities.

Keyphrase Mining There are a number of works on extracting keyphrases from single documents, both in an unsupervised [MT04] and supervised manner [WPF+99]. Our requirements for keyphrases are more specific, as the extraction needs to fit into the overall architecture of our NED system. Related work that complements our keyphrase harvesting is by Taneva and Weikum [TW13], which focuses on harvesting context for long-tail entities. Another recent work by Li et al. [LWH+13] creates a background model for unknown entities to get additional context for entities for the disambiguation. We do not compare our methods, as our main focus is on the discovery of new entities, whereas they are trying to improve the disambiguation quality for existing ones.

Alias Discovery A related problem to discovering when an existing name refers to a new entity is the discovery of new names for existing entities, also known as alias discovery [CGX09, BMI11, JWL+12]. The key difference is that alias discovery usually does not assume a canonical entity for which to harvest names, but takes existing name-alias pairs as input.

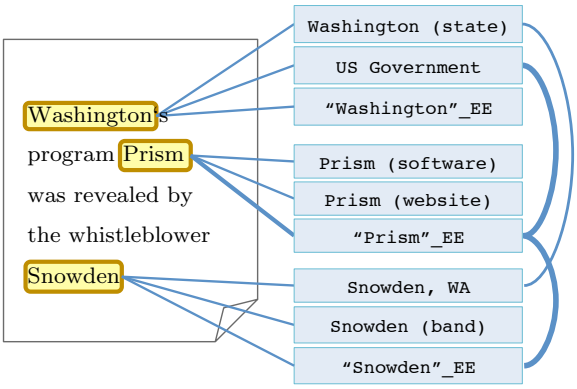

Figure 5.1.: Disambiguation graph with EE nodes

5.3. Model and Architecture

In established NED methods, mentions for which all candidate entities have only low scores according to the method's similarity and coherence measures are often considered unlinkable and mapped to an artificial node "nil" (the same node for all unlinkable mentions). In the current chapter, we extend this NED model into an NED-EE setting which integrates emerging entities (EEs) in a more flexible and principled manner. We assume that each mention m could also be an EE and introduce an additional entity node as a candidate for the disambiguation. In Figure 5.1, there are 3 such EE nodes, marked by the suffix _EE. Section 5.5 will later describe how we obtain keyphrase features for this new out-of-KB entity. Once we have expanded the candidate space E_m and the feature space this way, we can feed the NED-EE problem back to the NED method, which can now treat the EE the same way as it treats in-KB entities.

Although we may apply this expansion of E_m for every mention, it could be wise to omit it for such mentions that have a high-confidence mapping to an existing in-KB entity. Adding an EE candidate would potentially introduce noise even in such clear cases, and would also come with the extra cost of computing keyphrase features. Section 5.4 will later describe how we compute confidence measures and select mentions for which EE candidates need to be created. The outlined building blocks are components of our system architecture for the NED-EE method presented in this chapter. Our implementation uses the Stanford NER Tagger for NER, and YAGO (Section 2.3.3) for the knowledge base. Our primary choice for the NED component is AIDA (Chapter 3). Note, though, that any NED method with suitable API can be plugged in.

5.4. Disambiguation Confidence

We have devised and studied various ways of modeling and computing the confidence for the individual outputs of NED methods. In the following, we first discuss a simple technique, based on normalizing the scores that most NED methods provide. Subsequently, we introduce two more elaborate approaches, based on perturbing the input of NED.

5.4.1. Normalizing Scores

The output scores that most NED systems provide for mention-entity pairs can often be interpreted as a notion of confidence. For probabilistic NED models, the output probability of a mention-entity pair being the right choice would be a direct confidence measure. However, many good NED methods are not probabilistic and yield scores that are not normalized in any way and cannot be directly interpreted. Even for probabilistic methods, all output probabilities could be very small because of (conditional) independence assumptions in the model and not easy to interpret.

To handle a-priori unbounded scores, we propose to normalize scores for m-e pairs on a per mention basis as follows. For each candidate e of a given mention m, compute:

$$\text{norm}_{score}(m, e) = \frac{\text{score}(m, e)}{\sum_{e_i \in E_m} \text{score}(m, e_i)}$$

The intuition is that if the best entity for m accumulates high score mass with respect to the total score mass of all other candidates, the confidence that the disambiguation is correct is high. So we define the confidence for mapping mention m correctly as follows:

$$\text{conf}_{norm}(m) = \text{norm}_{score}(m, \arg\max_{e \in E_m} \text{score}(m, e))$$

5.4.2. Perturbing Mentions

Intuitively, we should have high confidence in a NED method choosing entity e for mention m if that choice remains invariant under many variations of the input context. For example, if leaving out or changing some of the sentences of a news article does not affect the m-e mapping, this is a sign of a robust and high-confidence choice.

One specific technique along the lines of such a perturbed input approach is to drop mentions from the input text. There are many ways of dropping one, two, or a small number of mentions; so we choose them randomly and iterate such perturbations (in the spirit of i.i.d. – independent and identically distributed – sampling).

We feed each of the perturbed inputs into a NED method, treating NED like a black box, and collect the outputs of each run: the best m-e mappings for the given input context. We keep track how often different entities are chosen for the same mention, and aggregate over all perturbations. This procedure works as follows:

1. For a given set of mentions M, run the NED(M) algorithm to get the initial results (m_i, e_i)
2. Keep two counters for each m_i: c_i, k_i
3. For a fixed number of times (e. g. $k = 500$):
    1 Choose R as random subset of M
    2 For each $m \in R$ increase k_i, keeping track of how often m was actually present in the input

3 Run NED on the random subset, producing new results $(m_i, r_i) = \text{NED}(R)$

4 Increase the mention counter c_i if the initial entity e_i equals r_i, keeping track of how often the initial entity remained stable

Based on these counters over all perturbations, the confidence for a mention m_i is then computed as:

$$\text{conf}_{\text{perturb}}(m_i) = \frac{c_i}{k_i}$$

5.4.3. Perturbing Entities

Another way of perturbing the input to the NED method is to fix certain choices of entities for a small set of mentions, and study how the other – freely assignable – mentions are then best mapped. To this end, we first run the NED method at hand to compute its best output, ideally using joint inference over all mentions. Then, we change some of the chosen entities, forcing the corresponding mentions to be mapped to "incorrect" entities – incorrect as far as the NED method over all mentions would see it. Again, we choose the artificially "switched" entities at random, and we repeat this kind of perturbation many times.

The intuition here is that we should have high confidence in an m-e pair if switching entities for a small number of other mentions does not affect the final choice of e by the NED method. So we reward disambiguations that remain stable in the presence of perturbations.

Treating NED as a black box again, we run the following procedure:

1. For a given set of mentions M, run $\text{NED}(M)$ for initial results (m_i, e_i)
2. Keep two counters for each m_i: c_i, k_i
3. For a fixed number of times (e. g. $k = 500$):
   1 Choose random subset R of M
   2 For each e_i the (m_i, e_i) pairs in R, force-map m_i to an alternate e_j, choosing e_j in proportion to the scores of $\text{NED}(M)$ (or just uniformly)
   3 For each $m \in M \setminus R$ increase k_i, keeping track of how often m was actually present in the input
   4 Run NED on $M \setminus R$, but leaving the removed mention strings in the input text and the forced-mapped alternate entities in the coherence model of the NED method; producing new results $(m_i, r_i) = \text{NED}(M \setminus R)$
   5 Increase the mention counter c_i if the initial entity e_i equals r_i, keeping track of how often the initial entity was chosen again

The fraction of times e_r is equal to e_i (the result of the original disambiguation) is the estimated confidence for a mention, the same as with the method perturbing the input.

Note that both perturbation methods can be emulated on a suitable representation of the NED problem without actually re-running the full NED method multiple times.

5.5. Extended Keyphrase Model

Many NED methods use keyphrases as entity features. The NED-EE method developed in this chapter primarily builds on the AIDA tool for standard NED (without EEs); so the techniques presented in the following are explained with respect to the AIDA method. Note, however, that it is straightforward to carry our NED-EE method over to other systems with similar features, e.g., the Illinois Wikifier [RRDA11].

5.5.1. Keyphrases for Existing Entities

All NED methods that use entity-specific keyphrases as features pre-compute these keyphrases and their weights, often based on Wikipedia href anchor texts or similar data. We utilize such pre-computed keyphrases for in-KB entities as well, but extend this approach by additionally harvesting keyphrases from document collections.

We perform this only for the entities that high-confidence mentions are mapped to by the given NED method; so we harness the confidence assessment presented in the previous section. We do this in order to enrich the keyphrase representation of in-KB entities, which in turn gives us a better handle to contrast these existing entities against the EE candidates for this and other mentions.

The dynamic keyphrase acquisition works as follows:

1. Retrieve the context for the high-confidence mention: all sentences in the window (5 preceding, 5 following) surrounding the mention.

2. Identify keyphrases in these sentences:

    1 Part-of-speech tag each sentence with the Stanford POS tagger [TKMS03].

    2 Extract all sequences conforming to a set of pre-defined part-of-speech tag patterns. These patterns extract all proper nouns as well as technical terms such as "surveillance program" (as defined in [JK95]), both serving as keyphrase candidates. See Appendix A for a full list of patterns.

3. Output each distinct entity-keyphrase and entity-keyword (every word occurring in a phrase), for subsequent co-occurrence counting.

AIDA uses these counts to assign weights to keyphrases and keywords based on their frequency (IDF) and mutual information (MI). The IDF score is a global score computed once per keyphrase and keyword, and is taken as the average of the KB IDF and the document collection IDF. The MI score is entity-specific and can be computed by contrasting the occurrence count of the entity c_e, the keyphrase c_k, and the intersection c_{ek}. All of these counts are the sum of both the dynamically harvested and the in-KB counts. From each sum we derive (estimated) probabilities by dividing by the collection size n, e. g. $p(e) = c_e/n$. The normalized pointwise mutual information score for an entity e and a keyphrase k is then computed as per Equation 3.1 in Section 3.3.4.

5.5.2. Modeling Emerging Entities

The key to NED-EE is to make EEs first-class citizens. To this end, we introduce, for each mention, an additional out-of-KB candidate entity that is initially represented just by the mention string. Recall that the interesting and difficult case is when that string is ambiguous and may also refer to one or more already existing in-KB entities. The point of NED-EE is that we nevertheless introduce a new EE candidate node for each mention. Figure 5.1 on page 61 illustrates this expanded situation.

For an informative contrast between equally named existing entities and such newly introduced EEs, we need a richer representation of the EE nodes; merely using the mention string is, by itself, meaningless. Therefore, we also compute characteristic keyphrases for each EE. We do this analogously to the dynamic keyphrases computed for in-KB entities, as described in the previous subsection: constructing text windows around mentions, extracting noun phrases, and so on.

Exploiting news streams Since we are mainly aiming to perform NED-EE over news streams, we address this problem by mining EE-specific keyphrases from chunks of news articles within a news corpus or incoming stream. For the given input document that NED-EE needs to handle, we identify all news articles whose timestamps are in the vicinity of the publication date and time of the input document, for example, the same day and the preceding week, or just the same day. For social-media postings, the temporal granularity would be smaller, say a few hours instead of a few days. The assumption here is that there is likely a fair amount of redundancy, and many EEs would appear in more than one article - but everywhere in ambiguous form.

Model difference Since we cannot a priori distinguish an in-KB entity and an EE with the same name from each other, we cannot directly compute keyphrases that are truly characteristic for the EE and uncharacteristic for the in-KB entities with the same name. This is a fundamental problem that we solve as follows. We actually compute keyphrases for the EE name, that is, the mention string, and the obtained phrases are actually characteristic for *all* entities with that name, covering both in-KB and out-of-KB entities. Now, since we already have a keyphrase model for all the in-KB entities with this name, potentially even dynamically enriched by the technique of the previous subsection, we can compare the keyphrase model for the name with the keyphrase model for the in-KB entities. In a nutshell, the set difference between these models is the characteristic model for the EE. As all phrases are weighted, this *model difference* is more elaborate, as shown in Algorithm 2.

The final result of this keyphrase acquisition and weighting procedure for EE nodes is that we have a rich and crisp characterization of each introduced EE that is discriminative against the in-KB entities that share the EE's name. AIDA also uses keyword-specific MI weights, which are gathered in the same manner. Name-keywords are just the set of distinct tokens of all name-keyphrases. This model is the basis for actually deciding whether a mention refers to an existing or an emerging entity. The extensions to the NED method for this purpose are presented in the following section.

Algorithm 2: Harvesting keyphrases for EE

Input: document, ambiguous name n
Output: EE-keyphrases and weights

1. Define a suitable chunk of news articles based on the input document's publishing date and time.
2. Harvest all keyphrases K_n for all mentions m of n from all articles in the news chunk as explained in Subsection 5.5.1, resulting in a list of tuples of $(k_n, b) \in (K_n \times \mathbb{N})$ of keyphrase occurrence counts for n.
3. Get all in-KB entity candidates E_n for n, and their keyphrases K_E with their co-occurrence counts $(k_e, c) \in (K_E \times \mathbb{N})$. The EE-count d for each keyphrase in K_n is adjusted by the collection size balance parameter $\alpha := \frac{\text{KB collection size}}{\text{EE collection size}}$ in the following way: $d = \alpha(b - c)$, where c is 0 if the keyphrase is not in K_E. The fraction accounts for the differences in collection sizes. In our use case, the KB size is the number of entities in YAGO, the EE size is the number of news articles the keyphrases are extracted from.
4. Adjust the occurrence counts of n: analogously to name-keyphrase co-occurrence counts, subtract the occurrence counts for all candidate entities $e \in E_n$ of n. The count is balanced with the same scalar α as the keyphrase co-occurrence.
5. Compute all the EE-keyphrase MI weights based on the adjusted counts. Compute all keyphrase IDF weights based on the combined document frequencies, where both the document frequencies and the collection sizes are balanced by α.

5.6. Discovering Emerging Entities

A knowledge base comes with an entity repository and a dictionary listing names of the entities. Both are incomplete due to the dynamics of the world. There are different tasks in maintaining and growing the knowledge base:

- To increase the coverage of the dictionary, new names or aliases for in-KB need to be discovered.
- Emerging entities can have a new name or make an existing name more ambiguous. Once an emerging entity is discovered, it needs to be added to the knowledge base with a representation suitable for further disambiguation tasks.
- The keyphrase models of both in-KB entities and emerging entities need to be continuously updated as well, by enriching them with additional phrases and by adjusting weights.

In the following, we focus on the second task, the core of the NED-EE problem. The first task, discovering new names for existing entities, is orthogonal to the topic of this chapter and thus out of scope. The third task has essentially been solved by the model and techniques of the previous section.

Algorithm 3: Discovering Emerging Entities

Input: mentions M, lower threshold t_l, upper threshold t_u
Output: emerging entities EE

1. Run the disambiguation for each $m \in M$ to get the initial entity assignment: $(m_i, e_i) = \text{NED}(M)$;
2. Compute the confidence c_i for each (m_i, e_i)
3. Set m_i to EE for each m_i where $c_i \leq t_l$
4. Set m_i to e_i for each m_i where $c_i \geq t_u$
5. For each m_i where $c_i > t_l$ AND $c_i < t_u$:
   add EE-candidate e_o with EE-keyphrase features
   add m_i to M', the to-be-disambiguated mentions
6. $(m_i, e_i') = \text{NED}(M')$
7. Set e_i to EE for each m_i where e_i' is e_o

A general method for discovering emerging entities with any kind of underlying NED system is described by Algorithm 3. It first runs the regular NED method. Then it (optionally) applies a threshold t_l on the confidences to drop low-confidence mentions in favor of EE or fix mentions to the initially assigned entities for high-confidence mentions above threshold t_u. After this step, the disambiguation problem instance is extended with EE-placeholders with the appropriate keyphrase representation as input to the second run of the NED. Choosing t_l as 0.0 and t_u as 1.0 creates the special case of only using the EE representation to discover emerging entities, running the NED only once as the first stage becomes unnecessary.

Emerging entities are especially interesting in the setting of a continuous stream of news, where they should be identified as quickly as possible to cover the latest events and associated people, organizations, etc. The discovery of emerging entities in a news corpus works by acquiring keyphrase from recent news chunks and constructing EE models, in terms of weighted keyphrases, as described in the previous section.

For this news setting, we introduce a new hyper-parameter γ to balance the influence of Wikipedia-derived weights and news-based weights in the final NED scores. Because of the very different styles and resulting differences in frequencies, merely reconciling these two assets based on counts alone is not sufficient. Thus, all edge weights in the disambiguation graph connected to at least one EE-placeholder are multiplied with a

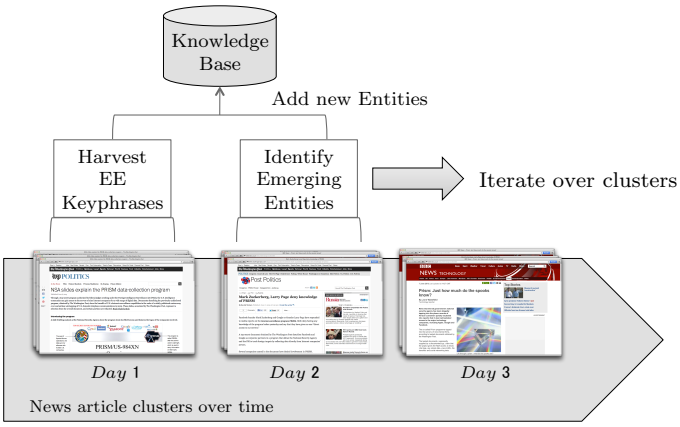

Figure 5.2.: Identifying EE on news data

factor of γ, tuned on withheld data.

The output of running NED on the NED-EE-based extended disambiguation graph is a mapping of all mentions to either in-KB or emerging entities. The mentions that are mapped to the same EE can be grouped together, and this group is added – together with its keyphrase representation – to the KB for the further processing in the KB maintenance life-cycle, as shown in Figure 5.2.

5.7. Experiments

We have experimentally evaluated two main contributions of this chapter: assessing the confidence in the disambiguation of mentions, discussed in Section 5.7.1, and discovering emerging entities for ambiguous names, discussed in Section 5.7.2.

5.7.1. Disambiguation Confidence

The confidence in the disambiguation of a mention to an entity is important both for down-stream applications using the results of NED, as well as for the method itself to improve its own quality. In Section 5.4, we introduced three methods for confidence assessment, one based on normalizing NED scores and two based on input perturbations. We found that of the three assessors, a linear combination of only two of them, the normalized scores and entity perturbation (with coefficients 0.5) gives the best results. Therefore, we use only this method, called CONF, in the following experiment, showing its influence on the overall NED quality.

Experimental Setup One measure for the goodness of a confidence assessor is the quality of the ranking of mentions that the assessor enables. To evaluate the ranking, we use

an aggregated precision-recall measure, (interpolated) MAP (mean average precision):

$$\text{MAP} = \frac{1}{m} \sum_{i=1..m} \text{precision@}\frac{i}{m} \tag{5.1}$$

where precision@$\frac{i}{m}$ is the precision at a specific recall level. This measure is equivalent to the area under the precision-recall curve. For constructing the precision-recall curve, we sort the mention-entity pairs in descending order of confidence, so that x% recall refers to the x% with the highest confidence.

As an additional measure for all the methods whose disambiguation scores are interpretable as confidence we give the precision at the confidence values 95% and 80%. We also give the number of mentions in the document collection that have at least this confidence.

For this study, we used the CoNLL-YAGO testb collection introduced in Section 3.6.1 with ground-truth annotation for 4,485 mentions in 231 news-wire articles. As underlying NED systems we primarily used AIDA, but also studied the Illinois Wikifier.

The confidence-estimation via normalization for AIDA takes different kinds of scores as input:

keyphrase: This is computed based on how well the entity keyphrases match the context of the mention.

weighted-degree: This score combines the keyphrase score and all the relatedness scores of a given entity in the disambiguation graph (see Figure 5.1) by summing the individual scores of all graph edges adjacent to the entity. The benefit of this score is that it captures how well the given entity fits to the other candidate entities instead of being restricted to the mention-entity score only. This score is used for the normalization component of the CONF method.

We compare the following competitors to our CONF method:

prior Disambiguation using only the mention-entity prior, estimated by counting how often the mention is used as a link anchor in Wikipedia to link to a given entity.

AIDA$_{coh}$ AIDA graph-coherence disambiguation ranked by keyphrase score (as in the original AIDA work).

IW Ranked by Illinois Wikifier [RRDA11] linker score.

Results The results for this comparison are given in Table 5.1. We see that CONF, the linear combination of the normalized weighted degree and entity perturbation, leads to substantial improvements in precision and MAP, at different cut-off levels for confidence. The precision-recall curves in Figure 5.3 show a similar picture. Note that the prior is most probably correct for mentions with a very high prior for their most popular entity (by definition), so the initial ranking of the prior and also the MAP are very good. However, it drops more sharply, as seen in Figure 5.3.

Measure	Competitors			Ours
	prior	**AIDA$_{coh}$**	**IW**	**CONF**
Prec@95%conf	91.98%	-	-	97.77%
#Men@95%conf	2020	-	-	1661
Prec@80%conf	82.55%	-	-	93.78%
#Men@80%conf	3038	-	-	2509
MAP	87.87%	86.75%	67.11%	93.72%

Table 5.1.: Influence of confidence assessors on NED quality

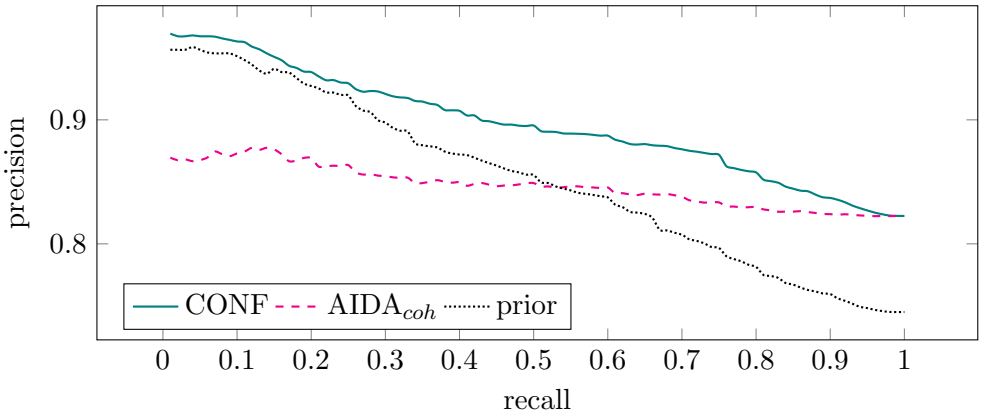

Figure 5.3.: Precision-recall curves with different confidence assessors

An interesting observation is the result for Precision@95% confidence. Being nearly 98% correct at the 95% confidence-level is an important factor in extending the keyphrases for existing entities without introducing a lot of noise. The number of entities that can be disambiguated with this confidence is not marginal, either, it's 1/3 of the mentions in the collection.

5.7.2. Emerging Entity Discovery

Dataset

A natural setup to evaluate the discovery of emerging entities would be to mine the keyphrases for the EE representation from the Web, subtracting the existing keyphrases from a truly global representation. However, these experiments are beyond the scope of an academic setting. Thus, we evaluate discovery in the setting of identifying emerging entities in news. There are no recent corpora available that are both chronological in nature and labeled with entities. The news content in the CoNLL-YAGO dataset used above predates Wikipedia and is not useful here.

To evaluate our emerging entity discovery, we annotated 150 Associated Press news articles published on October 1st and 150 articles published on November 1st, 2010, taken from the GigaWord 5 corpus [PGK+]. We automatically recognized mentions using

documents	300
mentions	9,976
mentions with emerging entities EE	561
words per article (avg.)	538
mentions per article (avg.)	33
entities per mention (avg.)	104

Table 5.2.: AIDA-EE GigaWord dataset properties

the Stanford NER system, then cleaned them manually to remove wrongly recognized mentions. However, we did not add mentions that were missed by Stanford NER. Each mention was annotated with the correct Wikipedia URL as of 2010-08-17, or EE if the correct entity was not present. To verify the absence of an entity, we checked both using the entity dictionary of AIDA as well as the Wikipedia internal and Google search. Otherwise, we annotated the correct entity. Every mention was annotated by two students, conflicts were reconciled by the author.

The dataset is available online at `http://www.mpi-inf.mpg.de/yago-naga/aida/` in the form of annotations for the original data, statistics are given in Table 5.2.

Evaluation Measures

We are evaluating the quality of the NED methods with the **Micro Average Accuracy** and **Macro Average Accuracy** defined in Section 3.6.1. Additional measures to evaluate the quality of the EE are defined over the following sets: D, the collection of documents; G_d, all unique mentions in document $d \in D$ annotated by a human annotator with a gold standard entity; $G_d|_{EE}$, the subset of G_d annotated with an emerging entity EE; $G_d|_{KB}$, the subset of G_d annotated with an with an existing, in-KB entity; and A_d, all unique mentions in document $d \in D$ automatically annotated by a method. The measures used are:

- **EE Precision**, the correct fraction of all mentions disambiguated as EE:

$$\text{EE Prec}_d = \frac{\left|G_d|_{EE} \cap A_d|_{EE}\right|}{\left|A_d|_{EE}\right|}$$

  The EE Precision is averaged over all documents.

- **EE Recall**, the fraction of all EE-labeled gold standard mentions correctly recognized by a method:

$$\text{EE Rec}_d = \frac{\left|G_d|_{EE} \cap A_d|_{EE}\right|}{\left|G_d|_{EE}\right|}$$

  The EE Recall is averaged over all documents.

- **EE F1**, the harmonic mean between the EE Precision and Recall, calculated per document then averaged.

Experiments

All EE experiments are run on the 150 annotated documents from 2010-11-01 in the AIDA-EE GigaWord corpus described in Section 5.7.2, adjusted in the following way:

- The mentions that are not in the entity dictionary are removed, as they can be resolved trivially.
- The full name of (existing) person entities is replaced with the last token of the name only (assumed to be the family name), e.g. we transformed "Edward Snowden" to "Snowden". This is to increase ambiguity and counter data sparsity issues in this small corpus.
- Mentions that occur in less than 10 distinct articles in the last 3 days before the day of the actual article (in the full GigaWord corpus, which has 7 different news sources) are removed. This is done to restrict the mention space to such mentions where the EE method has data.

This leaves a total of 3,436 mentions, out of which 162 are both ambiguous and refer to an emerging entity.

We estimated hyper-parameters on the training set of 150 documents from 2010-10-01 containing 3,935 mentions out of which 187 are EE, adjusted in the same way. In the case of the competitors, the hyper-parameter is the threshold. For the emerging entity method we estimated the γ parameter balancing the EE entity weight against regular entities, the number of days to look back when harvesting keyphrases, and the maximum number of keyphrases to use for each entity (to better balance very popular entities with a large number of keyphrases against long-tail entities). The best values are given below in the description of the methods. When harvesting the keyphrases for existing entities, the number of days is set to 30.

The methods should label each mention with either the correct entity or EE. Note that we restrict the input to the labeled mentions to compare the method's ability to distinguish between existing and new entity, not its ability to recognize names in the input text. The competitors we compare against are:

IW Ranked by Illinois Wikifier linker score. The threshold was estimated as -0.66. Note that we are not using the Wikifier's internal mechanism but perform the equivalent operation of thresholding after the actual run to simplify the integration in our framework.

AIDA$_{sim}$ AIDA keyphrase based disambiguation ranked by keyphrase confidence, thresholded at 0.15.

AIDA$_{coh}$ AIDA graph link-coherence disambiguation ranked by the CONF confidence, thresholded at 0.11.

For both our emerging entity methods the EE keyphrases are harvested from the previous 2 days, the maximum number of keyphrases per entity is set to 3000, and γ is

Measure	Competitors			EE	
Measure	**AIDA**$_{sim}$	**AIDA**$_{coh}$	**IW**	**EE**$_{sim}$	**EE**$_{coh}$
Micro Avg. Acc.	75.03%	75.81%	62.34%	53.75%	62.28%
Macro Avg. Acc.	73.22%	72.58%	62.79%	48.90%	60.90%
EE Prec.	72.84%	53.49%	66.59%	97.97%	93.92%
EE Rec.	89.09%	90.92%	90.88%	70.69%	71.72%
EE F1	66.61%	49.80%	63.89%	68.92%	67.92%

Table 5.3.: Emerging entity identification quality on 150 labeled news documents

	Competitors			NED-EE	
Method	**AIDA**$_{sim}$	**AIDA**$_{coh}$	**IW**	**AIDA-EE**$_{sim}$	**AIDA-EE**$_{coh}$
Micro Avg. Acc.	76.02%	75.81%	70.31%	76.11%	71.33%
Macro Avg. Acc.	73.40%	72.58%	70.52%	72.90%	70.40%
EE Prec.	72.84%	53.49%	66.59%	97.97%	93.92%

Table 5.4.: Quality of NED-EE (EE as preprocessing followed by regular NED)

set to 0.04 for **EE**$_{sim}$ and 0.06 for **EE**$_{coh}$. The knowledge base and dictionary is YAGO2, built from the Wikipedia snapshot from 2010-08-17.

EE$_{sim}$ AIDA keyphrase based disambiguation including EE placeholders and harvested keyphrases for existing entities.

EE$_{coh}$ AIDA graph coherence based disambiguation including EE placeholders and harvested keyphrases for existing entities, using KORE relatedness scores (see Chapter 4) for the coherence. Link-based coherence is not applicable, as it relies on the Wikipedia link graph that does not contain EE placeholders.

The results in Table 5.3 show that the EE methods clearly outperfom the competitors in terms of EE Precision and F1. The EE$_{sim}$ method achieves a very high precision of more than 98% with a recall of 71%. This result includes harvested keyphrases for existing entities, which improves the EE precision by 7 points and the EE F1 by 4.5 points over the same method using only EE-keyphrases (see Figure 5.4). Using graph coherence, EE$_{coh}$, does not change the results much in terms of F1, however it trades off precision in favor of recall. The reason for the average F1 being lower than both the average precision and recall in all cases is that F1 becomes 0 if either precision or recall are 0, which happens for a number of documents.

The results also show that the EE-methods, which include the regular entity disambiguation step, do not perform so well in the general NED setting, reflected by lower accuracy numbers. We also experimented using the EE-method as a preprocessing step, removing all the recognized EE mentions for a subsequent NED run. This NED run uses

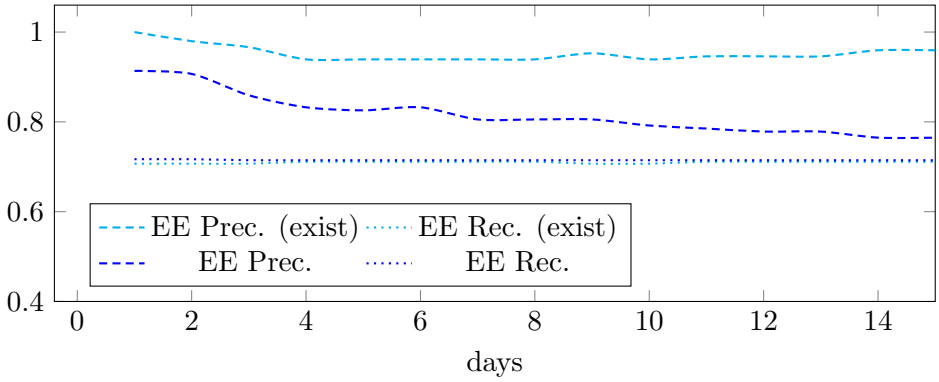

Figure 5.4.: EE discovery over number of days used for the EE representation

the best non-EE AIDA out of the box, which is the AIDA$_{coh}$ variant without thresholding. The results are shown in Table 5.4, where each method from Table 5.3 is used as pre-processing step to identify emerging entities. The EE Precision stays the same, as EEs are used as input and stay untouched. The accuracy numbers show that using pre-identified emerging entities improves AIDA-EE, which achieves the best NED quality in this setting.

However, the results of the AIDA-EE$_{coh}$ method reveal that judging the quality of the EE methods by the EE measures alone is not sufficient and that they need to be evaluated in the full NED setting. Even though the same NED method is run for the outputs produced by EE$_{sim}$ and EE$_{coh}$, which are similar in terms of the EE measures, AIDA-EE$_{coh}$ performs worse. This is because EE$_{coh}$ tends to make mistakes that create a more difficult input for the actual NED method, see Section 5.7.3 below for examples.

Harvesting keyphrases for in-KB entities makes the NED-EE method more robust. Figure 5.4 shows how the precision of the EE-discovery degrades once keyphrases for representing the EE-placeholder are harvested from more days. Adding keyphrases for existing entities not only improves the precision, but keeps EE-placeholders from dominating the existing entities, stabilizing precision over time. The keyphrases for existing entities where harvested from the full month preceding the test data (100,000 GigaWord documents) – here noise is not a big issue as we only include keyphrases for very high confidence disambiguations ($\geq 95\%$), which we found to be accurate for 98% of mentions (cf. Table 5.1).

One more interesting observation is that the achieved EE F1 of the thresholding approaches is better on the training data where the threshold is optimized on, but it does not generalize well. This was remarked numerous times [HRN$^+$13, RRDA11] before. Explicitly representing the EE entity instead of deciding based solely on the features for existing entities is more robust. We also evaluated TagMe [FS12], however the numbers are far worse due to the fact that TagMe does not accept pre-defined mentions in the input. This makes results nearly incomparable, so we do not give numbers.

5.7.3. Interesting Examples

Keyphrases for existing entities Harvesting keyphrases for existing entities is essential to adapt the entity representation to the actual corpus. One example is the mention of "Theresa May" in this sentence of Document APW_ENG_20101101.0164.

> British Home Secretary Theresa May said the bomb discovered [...] was powerful enough to bring down the aircraft.

Example 5.12

While the mention "Theresa May" is unambiguous in Wikipedia, always denoting the British politician, the actual `Theresa May` is dominated by the EE-placeholder entity because none of the recent events covered in the news are part of Wikipedia (yet). However, by augmenting the existing keyphrases with high-confidence disambiguations from the days before to `Theresa May`, context from these events is now associated with her in the form of salient keyphrases like "U.S. intelligence officials" or "chief suspect", which occur in the remainder of the text. Thus, the mention is correctly recognized as in-KB.

sim vs. coh in NED-EE In the NED-EE setting, the choice of the EE discovery method has a bigger impact that the actual EE numbers suggest. While EE_{sim} and EE_{coh} perform very similar in the plain EE setting, the results in NED-EE differ significantly. To given an example, EE_{coh} falsely labels the mention "Kennedy" as EE instead of `John F. Kennedy` in Document APW_ENG_20101101.0020, removing a mention from the input that would be solved correctly by the NED and is crucial to disambiguate the remaining mentions in the same document.

5.8. Summary

We presented a new approach to discovering emerging entities with ambiguous names by discriminating them against existing entities in a knowledge base. Our method is based on the principle to make emerging entities first-class citizens in the NED method itself, by introducing new EEs to the candidate space of the method. To make this idea work, we devised and studied confidence assessors for entity mentions, and we developed an extended form of keyphrase representation that captures EE candidates in terms of a model difference. Our experiments show the viability of our NED-EE approach, the high precision in discovering emerging entities, and improvements in the precision of the NED output.

The corpus of 300 annotated news articles from the GIGAWORD5 dataset are available as annotations at `http://www.mpi-inf.mpg.de/yago-naga/aida/`.

6. Applications

Links from unstructured to structured knowledge in the form of entities is not only a fundamental ingredient for further information extraction methods, but are immediately useful for end user tasks.

In the most straight-forward manner, **hyperlinks to background knowledge about entities**, e. g. their Wikipedia articles, can be provided for any kind of input text. When people read a text they can then easily find background information about potentially unknown entities. Imagine a student of politics reading old news articles or monographies on the initial developments of the European Union in the 1950s. A large number of people where involved and a multitude of new organizations have been formed then, and it is immediately clear that it is very helpful to provide background knowledge in a manner similar to the links in every Wikipedia article.

An even more important aspect is the retrieval of relevant documents to actually read in the first place, even for more recent events. Imagine the same student of politics searching for news on the Ukrainian conflict in 2014. This search can is exacerbated by the ambiguity of names: a search for "Klitschko", a leading figure during the Maidan square riots, will also return documents about his brother, who still sticks to his boxing roots. To remedy this problem, one can imagine **a search engine where entities are built-in**: now the student can retrieve documents the actual person `Vitali Klitschko` and not miss out on parts of the news where he is only referred to as "Klitschko". We developed such a system for searching news collections with strings, things, and categories (or classes), coined Stics [HMW14]. The following sections will detail the approach for search and its extension to analytics.

6.1. Searching for Strings, Things, and Cats

"Things, not Strings" has been Google's motto when introducing the Knowledge Graph and the entity-awareness of its search engine. When you type the keyword "Klitschko" as a query, Google returns Web and news pages and also explicit entities like Wladimir Klitschko and his brother Vitali (with structured attributes from the Knowledge Graph). Moreover, while typing, the query auto-completion method suggests the two brothers in entity form with the additional hints that one is an active boxer and the other a politician.

However, the Google approach still has limitations. First, recognizing entities in a keyword query and returning entity results seems to be limited to prominent entities. Unlike the Klitschko example, a query for the Ukrainian pop singer "Iryna Bilyk" does not show any entity suggestions (neither for auto-completion nor in the search results). Second, Google seems to understand only individual entities, but cannot handle sets of

entities that are described by a type name or category phrase. For example, queries like "Ukrainian celebrities" or "East European politicians" return only the usual ten blue links: Web pages that match these phrases. The search engine does not understand the user's intention to obtain lists of people in these categories.

The entity disambiguation work presented in Chapters 3, 4, and 5 lays the foundation for STICS, a novel system that extends entity awareness in Web and news search by tapping into long-tail entities and by understanding and expanding phrases that refer to semantic types. STICS supports users in searching for strings, things, and categories in a seamless and convenient manner. For example, when posing the query "Merkel Ukrainian opposition", the user is automatically guided, through auto-completion, to the entity `Angela Merkel` and the category `Ukrainian politicians`, and the latter is automatically expanded into `Vitali Klitschko`, `Arseniy Yatsenyuk`, etc. The search results include pages talking about "the German chancellor met the Ukrainian opposition leader and former heavy-weight champion", even if these texts never mention the strings "Angela Merkel" and "Vitali Klitschko".

6.1.1. Related Work

Prior work on semantic search and entity retrieval (e.g., [BBdR11, NWM12]) has mostly focused on keyword input and entities as query results. Some work on searching with entities and relations (e.g., [ERW09]) requires structured input queries and returns RDF triples or Web tables. [BB13, BCSW07] focuses on efficient auto-completion for combined phrase- and entity-queries, with the goal of retrieving entities. All this is quite different from our model which, additionally to phrases, allows entities as input and returns documents as output. No prior work has considered a combination of text phrases, entities, and semantic categories in the query model.

6.1.2. Algorithmic Building Blocks

The basis for querying text documents in terms of entities and categories are the disambiguated entities and their associated types. The entities are disambiguated using AIDA (see Chapter 3), the types from YAGO (see Section 2.3.3) are used as categories. We have run this machinery on a corpus of ca. 1,000,000 news articles collected from 300 important feeds over the course of more than a year. Based on this annotated and indexed corpus, the STICS search engine comprises a number of building blocks for query auto-completion, document ranking, and category expansion at query time.

Auto-Completion. The user does not know the underlying knowledge in advance; so in order to navigate a huge number of entities and categories, some form of input auto-completion is crucial. Our method consists of two components: the first one for retrieving the candidates for a given prefix typed by the user, and the second one for ranking the candidates in a judicious manner. For *candidate retrieval*, the full canonical names of entities and categories are indexed for prefix lookups. For entities, we additionally store all token-level suffixes; so users are able to start typing at any token (e. g. "Merkel" will also show `Angela Merkel`). For categories, where the order of tokens is

somewhat arbitrary ("Presidents of the United States" vs. "United States Presidents"), we additionally index all 3-token permutations. For the *candidate ranking*, the entities are ranked by global popularity, estimated by the number of incoming links to their Wikipedia articles. The categories are ranked by a mixture model capturing both the category's prominence estimated by the prominence of all its entities as well as its specificity. A category's specificity is estimated by the likelihood of its entities being present also in other categories. This way, the suggestions prefer categories whose entities are truly specific and do not appear in many other coarse-grained categories.

Document Ranking. Our demo setting pertains to a continuous stream of news documents that we annotate with entities in real-time. In this setting, ranking is best done chronologically, with the latest news shown at the first spot. However, we can change the ranking model to incorporate statistical language models where token and entity models are combined using a mixture model including time [LC03].

Category Expansion. Once the query is fully specified, categories need to be expanded into a set of individual entities to be looked up in the index. If there is no further context given in the query, the most prominent entities for the categories are used. However, once the user intent is made clear by specifying additional entities, the expansion should prefer entities that are both prominent and related to the user-given entities. Once `Angela Merkel` is specified next to `Ukrainian politicians`, the Klitschko brothers, as long-time German residents, are ranked higher. We realize this context-awareness by a mixture model of global entity prominence (like for auto-completion ranking) and global entity-entity coherence computed by the entity relatedness defined in Equation 3.7 on page 26. The coherence is computed between all entities of a given category and all entities specified directly in the query, averaging over all of them.

6.1.3. Use Cases

Searching with String, Things, and Cats. The possibility to pose queries beyond strings enables a new search experience, allowing users to specify their actual intent in a crisper manner. However, not all search intents can be expressed using entities and categories, so the combination with regular search terms is kept. An example where combining all options that STICS offers is crucial is searching for opinions of `Angela Merkel` on the scandalous "phone call" of the US diplomat `Victoria Nuland` in the context of the Ukrainian riots. Figure 6.1 shows a screenshot of STICS for this example query.

Navigating Documents by Entities. The most frequent entities in the retrieved documents are displayed next to each result, allowing quick refinement of the initial query. In this setting the category search shows its potential, as users can begin their search by a more general category of entities, then refine by clicking the displayed entities. Thanks to the context-aware expansion, the refinement effect is even more pronounced. For instance, adding `Angela Merkel` in the example query raises the expanded `Klitschko brothers` by 5 ranks, thus re-adjusting the query's focus.

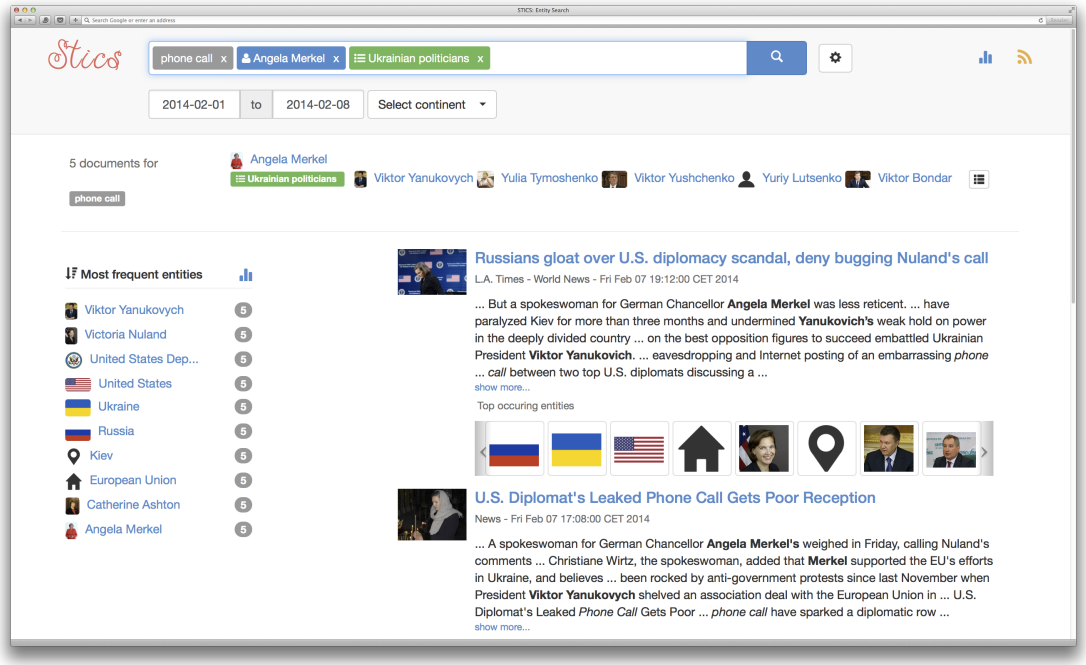

Figure 6.1.: Screenshot of the STICS search engine over a news corpus

6.2. Analytics with Strings, Things, and Cats

Proper analysis and understanding of large amounts of texts has become a major necessity, as the amount of natural-language contents keeps growing, especially in the context of social media and daily news, but also regarding scholarly publications and digitized books. A notable project in this line is the Culturomics [MSA+11] project, which supports an aggregated view of trends in the recent human history captured by the Google books corpus. Here, interesting conclusions are drawn about linguistic changes or cultural phenomena using string-level keywords, by comparing frequencies over time periods and across languages.

The system presented in this section extends the STICS search system presented in the previous section. The analytics extension is called AESTHETICS (short for Analysis and Exploration with Strings, Things, and Categories), and goes beyond mere string-based analysis, by supporting the analysis and exploration of entities ("things") and categories ("cats"). This semantic level is provided by linking text phrases to knowledge bases The underlying technology is, as with STICS, AIDA (see Chapter 3), which disambiguates all mentions to YAGO (see Section 2.3.3). Thus, instead of specifying words or phrases as the target of mining trends and patterns, we can now see and analyze entities directly. AESTHETICS supports the analysis of textual surface phrases ("strings") as well, but its full power comes from combining these with proper entities and semantic categories.

To illustrate why this is a major step with strong benefits, consider the task of vi-

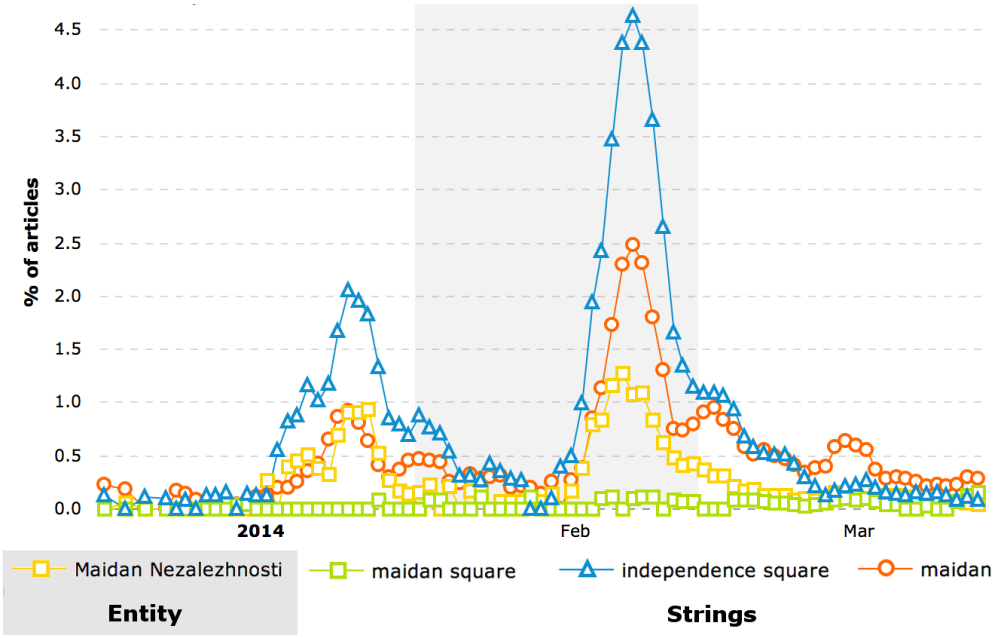

Figure 6.2.: Accurate analytics for `Maidan Square`

sualizing trends around the recent Ukrainian crisis, which originated from the Maidan, the square in Kiev where thousands of Ukrainians protested in early 2014. A search for "Maidan" quickly reveals that the name is highly ambiguous, as it means "square" not only in Ukrainian, but also in Hindi and Arabic. Thus, simply counting the string "Maidan" will result in a large number of false positives, leading to an imprecise analysis, as shown in Figure 6.2. By specifying the canonicalized entity `Maidan Nezalezhnosti`, not only do we get rid of spurious mentions of other Maidans, but also find articles where the square is mentioned only by its English name "Independence Square". Thus, entity-level analytics, as supported by AESTHETICS, is the only way to get accurate numbers.

Additionally, as we now have the full potential of a structured knowledge base in the background, further opportunities are opened up. In all semantic knowledge bases, entities are organized in a category hierarchy, e. g. `Greenpeace` is an `environmental organization`, which in turn is a subclass of a general `organization`. Using this category hierarchy, we can conduct analyses for entire groups of entities, for example, comparing the presence of `environmental organizations` and `power companies` in news of different parts of the world, deriving a picture of how their importance changes over time. The hierarchical organization of categories adds another dimension for aggregation to the usual temporal and spatial dimensions (where news can be aggregated by publishing times and originating regions).

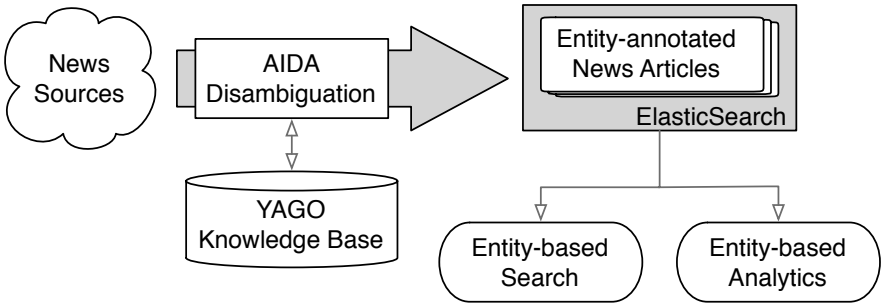

Figure 6.3.: AESTHETICS architecture

6.2.1. Related Work

There is ample work on text analytics for identifying (and visualizing) trends and patterns. Although this is of importance also for enterprise documents, most of the published research addresses social media (see, e.g., [Meh13] and references there). The goals here are manifold: detecting emerging topics [AMRW11], discovering events [DSJY11], mining story-lines [SG12], connecting topics and users [AYAG+12], identifying trendy catchphrases and their temporal evolution [SHE+13], and more.

Applications of such methods include Culturomics [MSA+11] over the Google books corpus, product and market analytics over web contents [CCC+13], and also computational journalism [HM12]. Although some of this work refers to "entities", the granularity of analyses really is entity names, disregarding their ambiguity. Also, there is no awareness of background knowledge bases. Two notable exceptions are the recent works by [SW12] and [HBS13]. The former is a proof-of-concept project for annotating Web archive contents with entities. The latter applies shallow methods for entity markup to the French newspaper Le Monde, to support entity-aware Culturomics. Our system uses much deeper methods for entity disambiguation and supports much richer analytics over both entities and categories.

[TLP+08] developed the NewsStand system for entity-annotated exploration and visualization of news articles. However, this work considers only locations as entities, and disregards people, organizations, products, and events. Also, it does not address analytics.

6.2.2. News Analytics Architecture

The AESTHETICS engine for news analytics allows users to quickly spot trends of entities or groups of entities specified by a semantic category. The entities and all their categories associated in the knowledge base are indexed using ElasticSearch[1], which provides convenient retrieval of entity and category frequencies per day. ElasticSearch also provides a full text index, so we can combine entities, categories, and words or (multi-word)

[1] `www.elasticsearch.org`

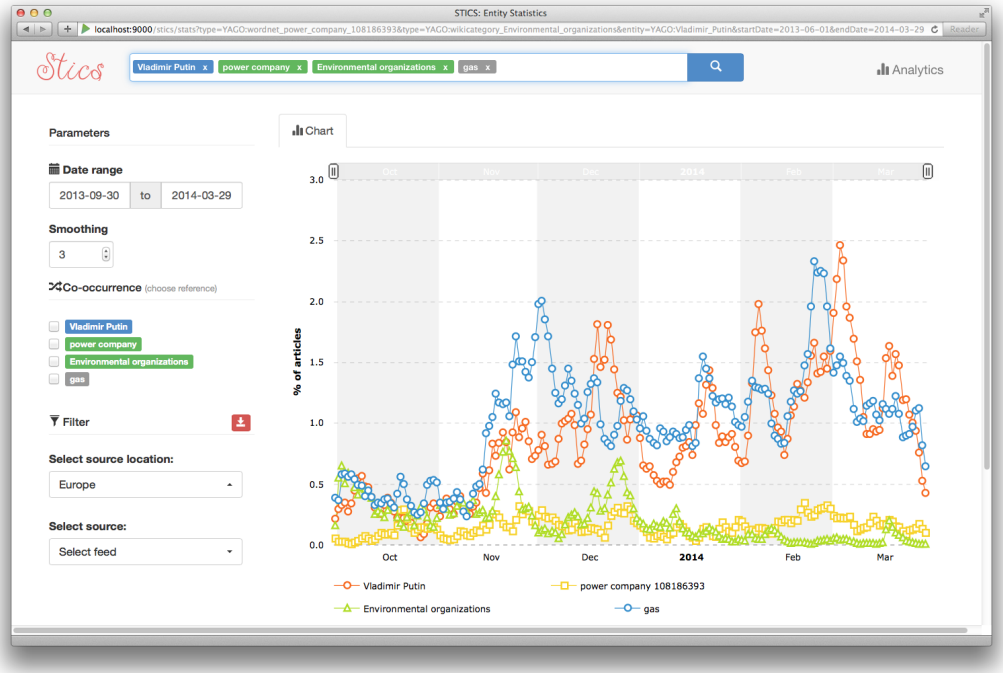

Figure 6.4.: Analyzing the connection between `Vladimir Putin` and "gas" in the AESTHETICS interface.

phrases. The architecture of AESTHETICS is visually summarized in Figure 6.3.

One challenge is to support the user when specifying the entities and categories of interest for the actual analysis. To this end, AESTHETICS uses the same entity and category suggestion mechanism as STICS, described in Section 6.1.2.

A screenshot of the AESTHETICS Web interface is shown in Figure 6.4. In the search box at the top, entities, categories, and strings can be specified using auto-completion. On execution of the query, our system displays a chart showing the trend lines based on the daily occurrences of each item in the central area. Each of the query items can be selected as a reference item, switching the counts for the remaining items from occurrence to co-occurrence counts. Additionally, AESTHETICS provides the more standard temporal and spatial filtering capabilities, allowing to filter by time, by region (like Asia or Europe), or by a selection of news sources. This is especially interesting as it allows users to contrast the occurrences of the query items by different regions or different newspapers. Different filters can be selected and are displayed side by side in the same graph (not shown in the screenshot). In another mode, AESTHETICS provides summary statistics for individual entities. Here, the top co-occurring entities for a given timespan are automatically determined and displayed. For the analyst, this is often a first step towards finding interesting co-occurrences. Mining unexpected co-occurrences of interesting categories is something that no string-based system can ever hope to accomplish.

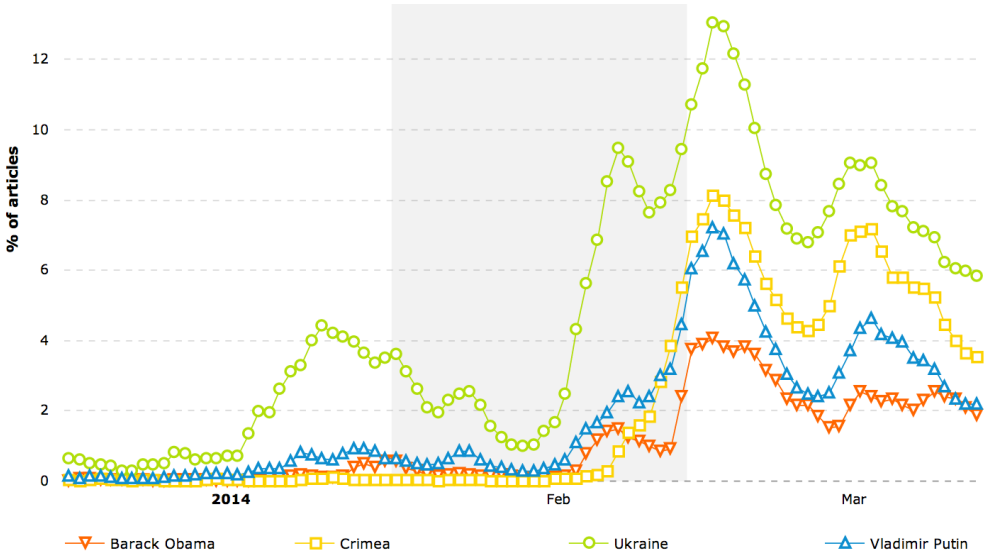

Figure 6.5.: Entities in the context of the `Ukraine` crisis

6.2.3. Use Cases

Accurate entity counts Entity-based analysis improves the accuracy over string-based analysis. The precision is improved as ambiguous names are resolved to the correct entity, removing unwanted occurrences. Better recall is achieved by also finding occurrences of the same entity under different names. Figure 6.2 shows a graph contrasting the various ways of specifying the target of the frequency analysis, using the example of the Maidan square in Kiev. Searching merely for "Maidan" overestimates the actual numbers by a factor of up to 2 (in February), while sometimes underestimating the true count (late January). Searching for its English name "Independence Square" is not useful, either, as there are squares with this name all around the world.

Entity co-occurrence analysis Comparing individual entity occurrences is already a powerful way of analyzing news and spotting trends or interesting topic shifts. A common scenario is that there is one main entity of interest, e. g. the `Ukraine`, and an analyst is interested in viewing and interpreting news in terms of other entities co-occurring with the main entity. Consider again the events around the Ukrainian crisis, analyzing the involvement of `Barack Obama` and `Vladimir Putin` over time, as depicted in Figure 6.5. AESTHETICS provides the advanced functionality of viewing occurrences for the three entities `Crimea, Obama`, and `Putin` given that `Ukraine` has to be mentioned in the text. Without this constraint, analyzing the impact of the event on mentions of `Obama` is hard, as he is mentioned in a large number of totally unrelated contexts. Using the co-occurrence restriction, it becomes easy to spot that he became only involved once the `Crimea` situation escalated, and not in the initial protests in `Ukraine`.

Analyzing groups of entities Analyzing occurrences of all entities in a semantic group (e.g., Ukrainian politicians, female political activists, etc.) is impossible with just strings. With unstructured text linked to entities in a category hierarchy, it becomes a natural thing to do. The screenshot in in Figure 6.4 shows an example of an insightful analysis of this kind. The figure shows in how many news power companies or environmental organizations have been mentioned in news. This analysis reveals that the rise of the Crimea tensions in February had a remarkable effect on the frequency of mentions of power companies – driven by Europe's dependence on Russian gas, which is often transported through Ukraine. On the other hand, mentions of environmental organizations, opposing environmentally hazardous ways of getting gas by fracking, have gone down. This anecdotic evidence is strengthened by the observation that the effect is especially pronounced in European news, and harder to spot when removing the region filter.

7. Conclusions

7.1. Contributions

The first contribution of this dissertation, AIDA, provides a high-quality named entity disambiguation method. The key contributions are a powerful framework and a large evaluation corpus. The framework combines the prior probability of an entity given a mention, the keyphrase-context based mention-entity similarity, and a link-based entity-entity relatedness measure into a graph problem, which allows for the disambiguation of all names in a text jointly. This method improves the quality over the state-of-the-art by judiciously adapting the features used to the input text, experimentally verified on a new and large, manually annotated collection of news-wire documents.

The second contribution, KORE, addresses the disambiguation of long-tail entities and entities beyond richly linked knowledge bases. To this end, KORE replaces the link-based entity relatedness measure used in AIDA by a more generic keyphrase-based one. The computation time of this measure is drastically reduced by a two-stage locality sensitive hashing pre-processing method. Experiments on KORE show an improved disambiguation quality for long-tail entities and a faster computation time than the state-of-the-art link-based measure.

The third contribution is a method to deal with emerging or out-of-knowledge-base entities by explicitly modeling such unknown entities, as well as a model to estimate the disambiguation confidence of AIDA. Both are a crucial building blocks for all disambiguation methods dealing with incomplete knowledge bases, which is almost always the case in real world scenarios. The combined confidence assessment and explicit model of unknown entities achieve a very high precision in experiments on the discovery of emerging entities.

STICS and AESTHETICS, two applications built upon the methods detailed above, bring entities to the scenario of news search and analytics. Users benefit from powerful ways of querying, navigating, and analyzing a large news collection: instead of searching by means of strings, now entities and classes are the unit of search. This enables users to specify their intent more crisply, using entities instead of ambiguous strings, and more completely, using semantic classes of entities instead of querying entity-by-entity.

7.2. Outlook

While a number of key problems have been addressed in this work, there are still possibilities for improvement and further research needs to be done.

7.2.1. Joint Entity Recognition and Disambiguation

In this work, recognizing mentions of entities in an input text and disambiguating these mentions to the correct entity are modeled as two stages in a pipeline. The separation of these stages has some immediate benefits by reducing the complexity of the task. However, combining the recognition and disambiguation can benefit the quality of both tasks. For example, the fact that a particular span of text has a candidate entity is already a good indicator that it is a correct mention. One can even think that after the results of a correct disambiguation can have further impact on quality. Take the following example sentence.

> Zeppelin did a great performance of Kashmir.

Example 7.13

If the disambiguation step correctly assigns `Kashmir (song)` to "Kashmir" based on good context in the surrounding sentences, this can be taken as a strong signal that "Zeppelin" is actually a mention. Had "Kashmir" been disambiguated to the region, the likelihood of "Zeppelin" being a mention would be smaller.

7.2.2. Genre Adaptation

Entity disambiguation works well on factual texts like news or scientific publications. Novels or other forms of fictional texts are very different in nature. Here, entities like characters or places might only exist in the texts, along with entities from the real world like cities where the novel takes pace. Entity repositories thus need to be enriched or even completely built from the texts themselves before being usable.

7.2.3. Domain Adaptation

This work, alongside most related work on entity disambiguation, uses Wikipedia as basis for the entity repository. This works well for news articles and other general-domain texts on the Web, but not for domain-specific texts such as research papers or instruction manuals. In such cases, the Wikipedia-based entity repository needs to be replaced or at least supplemented with domain-specific entities. Often, domain specific glossaries or even structured knowledge bases already exist, but there are some key differences:

- Wikipedia provides long articles describing entities. Entities from domain-specific glossaries often only have a short descriptive sentence.

- Wikipedia articles are interlinked. Each link actually represents the real entity it is linked to, and is a very rich source of features. Domain glossaries might have links between entities themselves, but usually not in the descriptive texts.
- Domain specific vocabularies are created with the specific goal to describe crisply and unambiguously the entity referred to so that specialists can communicate comfortably and precisely with each other. This entails that the ambiguity in domain-specific texts is less of a problem. It still exists however, as sometimes acronyms are overloaded or short forms are used. While the ambiguity is less, usually the names themselves are more complex and harder to recognize in text, so named entity recognition becomes more difficult.

Given these differences it becomes clear that existing disambiguation methods can not be used on domain specific texts and entity repositories out of the box.

Bibliography

[ABK+07] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the 6th International Semantic Web and 2nd Asian Semantic Web Conference, ISWC 2007/ASWC2007, Busan, South Korea*, pages 722–735, 2007.

[AH12] M H Abrams and Geoffrey Galt Harpham. *A Glossary of Literary Terms*. Wadsworth, Cengage Learning, 10th edition, 2012.

[AMRW11] Foteini Alvanaki, Sebastian Michel, Krithi Ramamritham, and Gerhard Weikum. EnBlogue: Emergent Topic Detection in Web 2.0 Streams. *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece*, pages 1271–1274, 2011.

[APRA10] Enrique Alfonseca, Marius Pasca, and Enrique Robledo-Arnuncio. Acquisition of Instance Attributes via Labeled and Related Instances. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland*, pages 58–65, 2010.

[AYAG+12] Sihem Amer-Yahia, Samreen Anjum, Amira Ghenai, Aysha Siddique, Sofiane Abbar, Sam Madden, Adam Marcus, and Mohammed El-Haddad. MAQSA: a System for Social Analytics on News. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA*, pages 653–656, 2012.

[BB13] Hannah Bast and Björn Buchhold. An Index for Efficient Semantic Full-Text Search. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM 2013, San Francisco, CA, USA*, pages 369–378, 2013.

[BBdR11] Krisztian Balog, Marc Bron, and Maarten de Rijke. Query Modeling for Entity Search Based on Terms, Categories, and Examples. *ACM Transactions on Information Systems*, 29(4), 2011.

[BCFM98] Andrei Z Broder, Moses Charikar, Alan M Frieze, and Michael Mitzenmacher. Min-Wise Independent Permutations (Extended Abstract). In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, STOC 1998, Dallas, Texas, USA*, pages 327–336, 1998.

[BCS+07] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open Information Extraction from the Web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI 2007, Hyderabad, India*, pages 2670–2676, 2007.

[BCSW07] H Bast, Alexandru Chitea, Fabian M Suchanek, and Ingmar Weber. ESTER: Efficient Search on Text, Entities, and Relations. In *Proceedings of the 30th ACM SIGIR International Conference on Research and Development in Information Retrieval, SIGIR 2007, Amsterdam, Netherlands*, pages 671–678, 2007.

[BH06] Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

[BMI11] D Bollegala, Y Matsuo, and M Ishizuka. Automatic Discovery of Personal Name Aliases from the Web. *IEEE Transactions on Knowledge and Data Engineering*, 23(6):831–844, 2011.

[BP06] Razvan Bunescu and Marius Pasca. Using Encyclopedic Knowledge for Named Entity Disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2006, Trento, Italy*, pages 9–16, 2006.

[CBK+10] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, Jr, and Tom M Mitchell. Toward an Architecture for Never-Ending Language Learning. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA*, pages 1306–1313, 2010.

[CCC+13] Tao Cheng, Kaushik Chakrabarti, Surajit Chaudhuri, Vivek R Narasayya, and Manoj Syamala. Data Services for E-Tailers Leveraging Web Search Engine Assets. *IEEE 29th International Conference on Data Engineering, ICDE 2013, Brisbane, Australia*, pages 1153–1164, 2013.

[CCCX11] Kaushik Chakrabarti, Surajit Chaudhuri, Tao Cheng, and Dong Xin. EntityTagger: Automatically Tagging Entities with Descriptive Phrases. In *Proceedings of the 20th International Conference Companion on World Wide Web, WWW 2011, Hyderabad, India*, pages 19–20, 2011.

[CCG+14] David Carmel, Ming-Wei Chang, Evgeniy Gabrilovich, Bo-June Hsu, and Kuansan Wang. ERD 2014: Entity Recognition and Disambiguation Challenge. *SIGIR Forum*, 2014.

[CFC13] Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. A Framework for Benchmarking Entity-Annotation Systems. In *Proceedings of the 22nd International World Wide Web Conference, WWW 2013, Rio de Janeiro, Brazil*, pages 249–260, 2013.

[CFR10] Don Coppersmith, Lisa K Fleischer, and Atri Rurda. Ordering by Weighted Number of Wins Gives a Good Ranking for Weighted Tournaments. *ACM Transactions on Algorithms*, 6(3), 2010.

[CGX09] Surajit Chaudhuri, Venkatesh Ganti, and Dong Xin. Mining Document Collections to Facilitate Accurate Approximate Entity Matching. *Proceedings of the VLDB Endowment*, 2(1):395–406, 2009.

[CLO+13] Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. Learning Relatedness Measures for Entity Linking. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM 2013, San Francisco, CA, USA*, 2013.

[CRV+14] Amparo E. Cano, Giuseppe Rizzo, Andrea Varga, Matthew Rowe, Milan Stankovic, and Aba-Sah Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In *#Microposts2014 at WWW2014*, 2014.

[Cuc07] Silviu Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2007, Prague, Cech Republic*, pages 708–716, 2007.

[CV07] Rudi L. Cilibrasi and Paul M.B. Vitanyi. The Google Similarity Distance. *IEEE Transactions on Knowledge and Data Engineering*, 19:370–383, 2007.

[DMR+10] Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. Entity Disambiguation for Knowledge Base Population. In *Proceedings of the 23rd International Conference on Computational Linguistics, COLING 2010, Beijing, China*, 2010.

[DSJY11] Anish Das Sarma, Alpa Jain, and Cong Yu. Dynamic Relationship and Event Discovery. *Proceedings of the 4th International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China*, pages 207–216, 2011.

[ECD+05] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. Unsupervised Named-Entity Extraction From the Web: an Experimental Study. *Artificial Intelligence*, 165(1):91–134, 2005.

[ERW09] Shady Elbassuoni, Maya Ramanath, and Gerhard Weikum. Language-Model-Based Ranking in Entity-Relation Graphs. In *Proceedings of the First International Workshop on Keyword Search on Structured Data, KEYS 2009, Providence, Rhode Island, USA*, pages 43–44, 2009.

[Fel98] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[FGM+02] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing Search in Context: the Concept Revisited. *ACM Transactions on Information Systems*, 20(1):116–131, 2002.

[FGM05] Jenny Rose Finkel, Trond Grenager, and Christopher D Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, ACL 2005, University of Michigan, USA*, pages 363–370, 2005.

[FH02] Michael Fleischmann and Eduard Hovy. Fine Grained Classification of Named Entities. In *Proceedings of the 19th International Conference on Computational Linguistics, COLING 2002, Taipei, Taiwan*, 2002.

[FS12] P Ferragina and U Scaiella. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *IEEE Software*, 29(1):70–75, 2012.

[GCK13] Stephen Guo, Ming-Wei Chang, and Emre Kıcıman. To Link or Not to Link? A Study on End-to-End Tweet Entity Linking. In *Proceedings of the Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, HLT-NAACL 2013, Atlanta, Georgia, USA*, pages 1020–1030, 2013.

[GIM99] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity Search in High Dimensions via Hashing. In *Proceedings of 25th International Conference on Very Large Data Bases, VLDB 1999, Edinburgh, Scotland*, pages 518–529, 1999.

[GM07] Evgeniy Gabrilovich and Shaul Markovitch. Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference for Artificial Intelligence, IJCAI 2007, Hyderabad, India*, pages 1606–1611, 2007.

[GS96] Ralph Grishman and Beth Sundheim. Message Understanding Conference - 6: A Brief History. In *Proceedings of the 6th Conference on Message Understanding, MUC 1995, Columbia, Maryland, USA*, pages 1–6, 1996.

[HAW14] Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. Discovering Emerging Entities with Ambiguous Names. In *Proceedings of the 23rd International World Wide Web Conference, WWW 2014, Seoul, South Korea*, pages 385–395, 2014.

[HBS13] Thomas Huet, Joanna Biega, and Fabian M Suchanek. Mining History with Le Monde. *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction, AKBC 2013, San Francisco, USA*, pages 49–53, 2013.

[HGT08] Wei Che Huang, Shlomo Geva, and Andrew Trotman. Overview of the INEX 2008 Link the Wiki Track. In *Advances in Focused Retrieval, 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008, Dagstuhl Castle, Germany*, pages 314–325, 2008.

[HM11] Samer Hassan and Rada Mihalcea. Semantic Relatedness Using Salient Semantic Analysis. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA*, 2011.

[HM12] Alon Halevy and Susan McGregor. Data Management for Journalism. *IEEE-CS Data Engineering Bulletin*, 35(3):7–15, 2012.

[HMW14] Johannes Hoffart, Dragan Milchevski, and Gerhard Weikum. STICS: Searching with Strings, Things, and Cats. *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2014, Gold Coast, QLD, Australia*, pages 1247–1248, 2014.

[HNR14] Ben Hachey, Joel Nothman, and Will Radford. Cheap and Easy Entity Evaluation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2014, Baltimore, Maryland, United States*, pages 464–469, 2014.

[HRN+13] Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R Curran. Evaluating Entity Linking with Wikipedia. *Artificial Intelligence*, 194:130–150, 2013.

[HS11] Xianpei Han and Le Sun. A Generative Entity-Mention Model for Linking Entities with Knowledge Base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT 2011, Portland, Oregon, USA*, pages 945–954, 2011.

[HSBW13] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. YAGO2: a Spatially and Temporally Enhanced Knowledge Base From Wikipedia. *Artificial Intelligence*, 194:28–61, 2013.

[HSN+12] Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. KORE: Keyphrase Overlap Relatedness for Entity Disambiguation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM 2012, Hawaii, USA*, pages 545–554, 2012.

[HYB+11] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard

Weikum. Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, Edinburgh, Scotland*, pages 782–792, 2011.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, STOC 1998, Dallas, Texas, USA*, pages 604–613, 1998.

[JGD11] Heng Ji, Ralph Grishman, and Hoa Trang Dang. Overview of the TAC2011 Knowledge Base Population Track. In *Text Analysis Conference*, 2011.

[JK95] J Justeson and S Katz. Technical Terminology: Some Linguistic Properties and an Algorithm for Identification in Text. *Natural Language Engineering*, 1:9–27, 1995.

[JKWL14] Yuzhe Jin, Emre Kıcıman, Kuansan Wang, and Ricky Loynd. Entity Linking at the Tail: Sparse Signals, Unknown Entities, and Phrase Models. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, USA*, pages 453–462, 2014.

[JWL+12] Lili Jiang, Juanyong Wang, Ping Luo, Ning An, and Wang Min. Towards Alias Detection Without String Similarity: An Active Learning based Approach. In *Proceedings of the 35th ACM SIGIR International Conference on Research and Development in Information Retrieval, SIGIR 2012, Portland, USA*, pages 1155–1156, 2012.

[k] e r j k p r u n c z y k. Bob Dylan. Flickr / CC BY NC SA.

[KGC13] B Keegan, D Gergle, and N Contractor. Hot Off the Wiki: Structures and Dynamics of Wikipedia's Coverage of Breaking News Events. *American Behavioral Scientist*, 57(5):595–622, 2013.

[KSRC09] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, Paris, France*, pages 457–466, 2009.

[KZ12] Alexander Kotov and ChengXiang Zhai. Tapping into Knowledge Base for Concept Feedback: Leveraging ConceptNet to Improve Search Results for Difficult Queries. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining, WSDM 2012, Seattle, Washington, USA*, pages 403–412, 2012.

[LC03] Xiaoyan Li and W Bruce Croft. Time-Based Language Models. *Proceedings of the 12th ACM International Conference on Information and Knowledge*

Management, CIKM 2003, New Orleans, Louisiana, USA, pages 469–475, 2003.

[LCL+04] Ming Li, Xin Chen, Xin Li, Bin Ma, and Paul M.B. Vitanyi. The Similarity Metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004.

[Len95] Douglas B Lenat. CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communcations of the ACM*, 38(11):32–38, 1995.

[LME12] Thomas Lin, Mausam, and Oren Etzioni. No Noun Phrase Left Behind: Detecting and Typing Unlinkable Entities. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, Jeju Island, Korea*, pages 893–903, 2012.

[LWH+13] Yang Li, Chi Wang, Fangqiu Han, Jiawei Han, Dan Roth, and Xifeng Yan. Mining Evidences for Named Entity Disambiguation. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, USA*, 2013.

[LZWZ11] Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. Recognizing Named Entities in Tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT 2011, Portland, Oregon, USA*, pages 359–367, 2011.

[MAD+09] James Mayfield, David Alexander, Bonnie Dorr, Jason Eisner, Tamer Elsayed, Tim Finin, Clay Fink, Marjorie Freedman, Nikesh Garera, Paul McNamee, Saif Mohammad, Douglas Oard, Christine Piatko, Asad Sayeed, Zareen Syed, Ralph Weischedel, Tan Xu, and David Yarowsky. Cross-Document Coreference Resolution: A Key Technology for Learning by Reading. In *Proceedings of the AAAI 2009 Spring Symposium on Learning by Reading and Learning to Read*, 2009.

[MC07] Rada Mihalcea and Andras Csomai. Wikify! Linking Documents to Encyclopedic Knowledge. In *Proceedings of the 16th ACM International Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal*, pages 233–242, 2007.

[McC09] Diana McCarthy. Word Sense Disambiguation: An Overview. *Language and Linguistics Compass*, 3(2):537–558, 2009.

[MD09] Paul McNamee and Hoa Trang Dang. Overview of the TAC 2009 Knowledge Base Population Track. In *Text Analysis Conference*, 2009.

[Meh13] Sharad Mehrotra, editor. *Special Issue on Social Media and Data Analysis.* IEEE-CS Data Engineering Bulletin 36(3). 2013.

[MSA+11] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, The Google Books Team, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A Nowak, and Erez Lieberman Aiden. Quantitative Analysis of Culture Using Millions of Digitized Books. *Science*, 331(6014):176–182, 2011.

[MT04] Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, EMNLP 2004, Barcelona, Spain*, pages 404–411, 2004.

[MW08a] David Milne and Ian H Witten. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *Proceedings of the AAAI 2008 Workshop on Wikipedia and Artificial Intelligence (WIKIAI 2008), Chicago, IL, USA*, 2008.

[MW08b] David Milne and Ian H Witten. Learning to Link with Wikipedia. In *Proceedings of the 17th ACM International Conference on Information and Knowledge Mining, CIKM 2008, Napa Valley, California, USA*, pages 509–518, 2008.

[MWdR12] Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. Adding Semantics to Microblog Posts. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining, WSDM 2012, Seattle, United States*, pages 563–572, 2012.

[Nav09] Roberto Navigli. Word Sense Disambiguation: a Survey. *Computing Surveys*, 41(2):1–69, 2009.

[Ng10] Vincent Ng. Supervised Noun Phrase Coreference Research: The First Fifteen Years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010, Uppsala, Sweden*, pages 1396–1411, 2010.

[NP12] Roberto Navigli and Simone Paolo Ponzetto. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250, 2012.

[NS07] David Nadeau and Satoshi Sekine. A Survey of Named Entity Recognition and Classification. *Lingvisticae Investigationes*, 30(24):3–26, 2007.

[NTW11] Ndapandula Nakashole, Martin Theobald, and Gerhard Weikum. Scalable Knowledge Harvesting with High Precision and High Recall. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining, WSDM 2011, Hong Kong, China*, pages 227–236, 2011.

[NTW13] Ndapandula Nakashole, Tomasz Tylenda, and Gerhard Weikum. Fine-grained Semantic Typing of Emerging Entities. In *Proceedings of the 51st*

Annual Meeting of the Association for Computational Linguistics, ACL 2013, Sofia, Bulgaria, pages 1488–1497, 2013.

[NWM12] Zaiqing Nie, Ji-Rong Wen, and Wei-Ying Ma. Statistical Entity Extraction From the Web. In *Proceedings of the IEEE*, pages 2675–2687, 2012.

[NZRS12] Feng Niu, Che Zhang, Christopher Ré, and Jude Shavlik. DeepDive: Web-scale Knowledge-base Construction using Statistical Learning and Inference. In *Proceedings of the 2nd International Workshop on Searching and Integrating New Web Data Sources, VLDS 2012, Istanbul, Turkey*, pages 25–28, 2012.

[PF11] Patrick Pantel and Ariel Fuxman. Jigs and Lures: Associating Web Queries with Structured Entities. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT 2011, Portland, Oregon, USA*, pages 83–92, 2011.

[PGK+] Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English Gigaword Fifth Edition. Technical report.

[PHP08] A Philpot, E H Hovy, and P Pantel. Ontology and the Lexicon, chapter: The Omega Ontology. Cambridge University Press, 2008.

[PN09] Simone Paolo Ponzetto and Roberto Navigli. Large-Scale Taxonomy Mapping for Restructuring and Integrating Wikipedia. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI 2009, Pasadena, California, USA*, pages 2083–2088, 2009.

[PN10] Simone Paolo Ponzetto and Roberto Navigli. Knowledge-Rich Word Sense Disambiguation Rivaling Supervised Systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010, Uppsala, Sweden*, pages 1522–1531, 2010.

[PS07] Simone Paolo Ponzetto and Michael Strube. Knowledge Derived from Wikipedia for Computing Semantic Relatedness. *Journal of Artificial Intelligence Research*, 30(1):181–212, 2007.

[PS11] Simone Paolo Ponzetto and Michael Strube. Taxonomy Induction Based on a Collaboratively Built Knowledge Repository. *Artificial Intelligence*, 175(9-10):1737–1756, 2011.

[RAGM11] 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 2011, Hyderabad, India*, pages 337–346, 2011.

[RN10] Altaf Rahman and Vincent Ng. Inducing Fine-Grained Semantic Classes via Hierarchical and Collective Classification. In *Proceedings of the 23rd International Conference on Computational Linguistics, COLING 2010, Beijing, China*, pages 931–939, 2010.

[RPH05] Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. Randomized Algorithms and NLP: Using Locality Sensitive Hash Functions for High Speed Noun Clustering. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL 2005, Ann Arbor, United States*, pages 622–629, 2005.

[RRDA11] Lev-Arie Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and Global Algorithms for Disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT 2011, Portland, Oregon, USA*, pages 1375–1384, 2011.

[SDM03] Erik F Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the 7th Conference on Natural Language Learning, CoNLL 2003, Edmonton, Canada*, 2003.

[SG10] Mauro Sozio and Aristides Gionis. The Community-Search Problem and How to Plan a Successful Cocktail Party. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2010, Washington, DC, USA*, pages 939–948, 2010.

[SG12] Dafna Shahaf and Carlos Guestrin. Connecting Two (or Less) Dots: Discovering Structure in News Articles. *ACM Transactions on Knowledge Discovery from Data*, 5(4), 2012.

[SHE+13] Caroline Suen, Sandy Huang, Chantat Eksombatchai, Rok Sosic, and Jure Leskovec. NIFTY: A System for Large Scale Information Flow Tracking and Clustering. In *Proceedings of the 22nd International Conference on World Wide Web, WWW 2013, Rio de Janerio, Brazil*, pages 1237–1248, 2013.

[SKW07] 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 2007, Banff, Canada*, pages 697–706, 2007.

[SM07] Ravi Sinha and Rada Mihalcea. Unsupervised Graph-based Word Sense Disambiguation Using Measures of Word Semantic Similarity. In *Proceedings of the First IEEE International Conference on Semantic Computing, ICSC 2007, Irvine, California, USA*, pages 363–369, 2007.

[SSPM11] Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. Large-Scale Cross-Document Coreference Using Distributed Inference and Hierarchical Models. In *Proceedings of the 49th Annual Meeting*

of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT 2011, Portland, Oregon, USA, pages 793–803, 2011.

[SW12] Marc Spaniol and Gerhard Weikum. Tracking Entities in Web Archives: the LAWA Project. In *Proceedings of the 21st International Conference Companion on World Wide Web, WWW 2012, Lyon, France*, pages 287–290, 2012.

[TKMS03] Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada*, pages 252–259, 2003.

[TKW11] Bilyana Taneva, Mouna Kacimi, and Gerhard Weikum. Finding Images of Rare and Ambiguous Entities. Technical report, 2011.

[TLP+08] Benjamin E Teitler, Michael D Lieberman, Daniele Panozzo, Jagan Sankaranarayanan, Hanan Samet, and Jon Sperling. NewsStand: a New View on News. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2008, Irvine, CA, USA*, 2008.

[Tur50] Alan M Turing. Computing Machinery and Intelligence. *Mind*, 59:433–460, 1950.

[TW13] Bilyana Taneva and Gerhard Weikum. Gem-based Entity-Knowledge Maintenance. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM 2013, San Francisco, CA, USA*, pages 149–158, 2013.

[WP94] Allison Gyle Woodruff and Christian Plaunt. GIPSY: Automatic Geographic Indexing of Documents. *Journal of the American Society for Information Science and Technology*, 45(9):645–655, 1994.

[WPF+99] Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. KEA: Practical Automatic Keyphrase Extraction. In *Proceedings of the 4th ACM Conference on Digital Libraries, DL 1999, Berkeley, CA, USA*, pages 254–255, 1999.

[YBH+12] Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, and Gerhard Weikum. HYENA: Hierarchical Type Classification for Entity Names. In *Proceedings of the 24th International Conference on Computational Linguistics, Coling 2012, Mumbai, India*, pages 1361–1370, 2012.

[YHB+11] Mohamed Amir Yosef, Johannes Hoffart, Ilaria Bordino, Marc Spaniol, and Gerhard Weikum. AIDA: An Online Tool for Accurate Disambiguation of

Named Entities in Text and Tables. In *Proceedings of the 37th International Conference on Very Large Databases, VLDB 2011, Seattle, WA, USA*, pages 1450–1453, 2011.

[ZMG08] Torsten Zesch, Christof Müller, and Iryna Gurevych. Using Wiktionary for Computing Semantic Relatedness. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, IL, USA*, pages 861–867, 2008.

A. Keyphrase Part-of-Speech Patterns

When extracting keyphrases from natural language texts, as done in Chapter 5, keyphrases candidates are proper noun phrases and technical terminology. We use the following regular expressions on part-of-speech patterns (Penn Treebank tag set) to extract them.

Proper Noun Phrases The patterns for extracting proper noun phrases are taken from [NTW11].

- (NNP\s)+(NNPS\s)*(NNS\s)*(NN\s)*(NNP\s)*(NNPS\s)*(NNS\s?)*
- (NNP\s)+(IN\s)(NNP\s?)+
- (JJ\s)(NNP\s?)+
- ((?=[^A-Z])DT\s)(NNP\s?)+

Technical Terms The pattern to extract technical terms is taken from [JK95].

- ((JJ\s|NNS?\s)+|((JJ\s|NNS?\s)*(NNS?\s IN\s)?)(JJ\s|NNS?\s)+)NNS?