Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

FAKULTÄT
FÜR MATHEMATIK, INFORMATIK
UND NATURWISSENSCHAFTEN

DISSERTATION

# Entity Analysis: Disambiguation and Typing with Graphs

Özge Sevgili Ergüven

Language Technology
Department of Informatics
Faculty of Mathematics, Informatics and Natural Sciences

Universität Hamburg
Hamburg, Germany

A thesis submitted for the degree of

*Doctor rerum naturalium (Dr. rer. nat.)*

2025

Entity Analysis: Disambiguation and Typing with Graphs

Dissertation submitted by: Özge Sevgili Ergüven

Date of Submission: 29.03.2025
Date of Disputation: 25.06.2025

Supervisors:
Prof. Dr. Chris Biemann, Universität Hamburg
Prof. Dr. Alexander Panchenko, Skolkovo Institute of Science and Technology

Committee:
Prof. Dr. Chris Biemann, Universität Hamburg
Prof. Dr. Sören Laue, Universität Hamburg
Prof. Dr. Ralf Möller, Universität Hamburg

Universität Hamburg, Hamburg, Germany
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics

Language Technology

# Affidavit

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Sofern im Zuge der Erstellung der vorliegenden Dissertationsschrift generative Künstliche Intelligenz (gKI) basierte elektronische Hilfsmittel verwendet wurden, versichere ich, dass meine eigene Leistung im Vordergrund stand und dass eine vollständige Dokumentation aller verwendeten Hilfsmittel gemäß der Guten wissenschaftlichen Praxis vorliegt. Ich trage die Verantwortung für eventuell durch die gKI generierte fehlerhafte oder verzerrte Inhalte, fehlerhafte Referenzen, Verstöße gegen das Datenschutz- und Urheberrecht oder Plagiate.

29.03.2025

Date

Signature

(Özge Sevgili Ergüven)

Dedicated to my baby...

# Acknowledgements

First and foremost, I would like to convey my sincere appreciation to my supervisors, Chris and Alex. This thesis could not have been completed without their invaluable support. I am grateful to Chris for believing in me and providing me this opportunity. Throughout this journey, there were extremely challenging moments, and especially during those times, he provided unwavering mental support and consistent encouragement. I am thankful to Alex for his constant support. Particularly in the early stages, he played a crucial role in guiding me through the challenges I faced. His support greatly eased my adjustment to this new academic environment, allowing me to focus more effectively on tangible outcomes.

I would also like to thank Deutscher Akademischer Austauschdienst (DAAD) for providing me this opportunity. I am thankful to the University of Hamburg and Language Technology group for the computing infrastructure.

I would like to express my appreciation to my colleagues in my research group, Language Technology, Debayan, Abhik, Steffen, Seid, Saba, Hans, Florian, and all other group members. We had many valuable discussions, and collaborations, along with enjoyable coffee breaks, especially with Kikka (Fussboll) playing. I would also like to thank Irina and Martin for our collaborations in the later stages of my journey. I only wish I had met you earlier! :) I am grateful to Hans and Martin for proofreading and cross-checking the Abstract part, and to Irina for proofreading the Introduction and Conclusion chapters. I would also like to express my appreciation to Artem for his collaboration; I have learned a great deal from him.

I would like to express my deepest gratitude to my family, each of whom has played an essential role in my journey. Without their unwavering support, I would have been lost along the way. To my mom, Gülseren, for supporting me to stay stronger; to my brother, Burak, for supporting me in following my dreams and encouraging me to step out of my comfort zone; to my father, Ali Ihsan, for his humorous conversations; to my nephew, Cinar, for his boundless, infectious energy that brightened even my toughest days, to my sister, Alev, for her understanding of the challenges I faced living in another country.

I would like to express my heartfelt thanks to my husband, Fatih. There are so many things I could say, but to start with, I am deeply grateful for his unwavering support in embarking on this journey with me—leaving behind everything familiar and adapting to a completely new way of life. He spent many weekends in coffee shops, working alongside me; stayed awake with me during long hours of work, always there to support me. These are just a few of the many ways he has supported me, making this journey not only easier but more meaningful. Finally, I want to express my heartfelt thanks to my child, even before your arrival. You have already brought immeasurable hope into my life. You are already the greatest gift I could have ever imagined, and I eagerly anticipate the beautiful journey that lies ahead.

# Use of Third-Party Software

I used Grammarly (https://www.grammarly.com/) to check grammar issues, in this thesis. I used draw io (https://app.diagrams.net/) to draw the figures. I used ChatGPT (https://chatgpt.com/) in the Introduction chapter and in the Acknowledgements, to rephrase sentences, to link sentences, to check grammar issues, and to find the synonyms for phrases. I also used it to find the famous quote, which is relevant to this thesis and to translate the Abstract from English to German, which was cross-checked by my colleagues later on. To find synonyms, I also used thesaurus (https://www.thesaurus.com/) throughout the thesis. As a dictionary, I used Cambridge (https://dictionary.cambridge.org/) and Tureng (https://tureng.com/). I used Overleaf (https://www.overleaf.com/), to write the thesis, with the template from the following link, https://github.com/uhh-lt/thesis-template-uhh-lt-latex.

*The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.*

— Claude Elwood Shannon, 1948

# Abstract

Information has been ubiquitous on the web in various styles, e.g. news articles, blog posts, encyclopedic contents, etc. Wikipedia, for example, consists of articles, containing encyclopedic information on entities, e.g. locations, organizations, persons, events, animals, cars, films, etc., and has grown into a valuable resource. Yet, finding the relevant information is still an issue: consider the scenario one wants to know about "What is the most critical step in Michael Jordan's career after joining the Chicago Bulls?". A search engine might retrieve the documents related to "Michael Jordan (basketball player)", "Michael Jordan (scientist)", and "Michael Jordan (football player)". The key challenge here is the lexical ambiguity of entities. Thus, resolving this ambiguity is a crucial process and task in natural language processing to enable several applications, e.g. retrieving the relevant information as examplified here.

In this thesis, we discuss the annotation of the entities in two levels: 1) the process of assigning labels to entity mentions, e.g. "Michael Jordan" in a textual context, with the ultra-fine type information, e.g. `basketball player`, `scientist`, `football player` – entity typing, and 2) the process of matching them to a knowledge graph record e.g. providing the link (https://en.wikipedia.org/wiki/Michael_Jordan) – entity disambiguation.

We study on these tasks in several aspects. We systemize design features of entity disambiguation and linking systems, developed since 2015 as a result of the "deep learning revolution" in natural language processing. This work distills a generic architecture and discusses its prominent components as well as the vast variety of modifications of this general architecture. In a similar manner, the summarization of the ultra-fine entity typing models, which address a lack of the annotated data issue is presented. There are numerous types in the type vocabulary of ultra-fine entity typing task. This results in difficulties for human to label them and thus, it is a crucial challenge in this task. To address this issue, we explore an unsupervised way, which requires no labeled data for training. This study relies on the information from a graph of terms (known as distributional thesaurus) that carries crucial information about terms and their relations. We explore the leverage of distributionally induced word/term senses through such a graph in an ultra-fine entity typing task. More specifically, we analyze the use of the labels information of the appropriate induced word/term sense to the entity mention. The graph is constructed through the features of terms, in this study, nonetheless, there could be many different graphs built in various ways. For example, a graph constructed through page links in Wikipedia information. We leverage such a graph to transform it to graph embeddings and utilize it as a vector representation of entities. One of the essential components of the generic architecture (presented in our first study) is entity representations with the goal to capture semantic meaning of the entities and/or semantic relatedness between entities in various aspects. We investigate the role of entity embeddings, constructed through this graph information. We present

a simple technique for the integration of the structured information of entities into an entity disambiguation model with graph embeddings.

All in all, we have analyzed recently proposed neural entity disambiguation and linking models, which generally show better performance than the classical solutions. With the guidance of this analysis, new researchers to this field would understand the task and this might help shaping future works in this field. We have investigated the leverage of information from the graph structures in entity disambiguation and ultra-fine entity typing tasks. We observe that use of information coming from graph structure helps on these tasks. This might lead to use such kind of information in different tasks/scenarios, for future studies.

# Zusammenfassung

Informationen sind im Web allgegenwärtig und in verschiedenen Formaten verfügbar, z. B. in Nachrichtenartikeln, Blogbeiträgen oder enzyklopädischen Inhalten. Wikipedia besteht beispielsweise aus Artikeln, die enzyklopädische Informationen zu Entitäten wie Orten, Organisationen, Personen, Ereignissen, Tieren, Autos, Filmen usw. enthalten, und hat sich zu einer wertvollen Ressource entwickelt. Dennoch bleibt die Suche nach relevanten Informationen eine Herausforderung: Stellen wir uns das Szenario vor, in dem jemand wissen möchte: „Was war der entscheidendste Schritt in Michael Jordans Karriere nach seinem Wechsel zu den Chicago Bulls?“ Eine Suchmaschine könnte Dokumente zu „Michael Jordan (Basketballspieler)“, „Michael Jordan (Wissenschaftler)“ und „Michael Jordan (Fußballspieler)“ abrufen. Die zentrale Herausforderung hierbei ist die lexikalische Mehrdeutigkeit von Entitäten. Daher ist die Auflösung dieser Mehrdeutigkeit ein wesentlicher Prozess und eine zentrale Aufgabe in der Verarbeitung natürlicher Sprache, um verschiedene Anwendungen zu ermöglichen, z. B. das Abrufen relevanter Informationen, wie es hier veranschaulicht wurde.

In dieser Arbeit diskutieren wir die Annotation von Entitäten auf zwei Ebenen: 1) den Prozess der Zuweisung von Labels zu Entitätsnennungen, z. B. „Michael Jordan“ in einem Textkontext, mit ultra-feinen Typinformationen, z. B. `Basketballspieler`, `Wissenschaftler`, `Fußballspieler` – Entitätstypisierung, und 2) den Prozess des Abgleichs mit einem Eintrag in einem Wissensgraphen, z. B. durch Bereitstellung eines Links (https://de.wikipedia.org/wiki/Michael_Jordan) – Entitätsdisambiguierung.

Wir untersuchen diese Aufgaben aus verschiedenen Perspektiven. Wir systematisieren die Designmerkmale von Entitätsdisambiguierungs- und Verknüpfungssystemen, die seit 2015 als Folge der „Deep-Learning-Revolution“ in der Verarbeitung natürlicher Sprache entwickelt wurden. Diese Arbeit destilliert eine allgemeine Architektur und diskutiert ihre wichtigsten Komponenten sowie die Vielzahl an Modifikationen dieser allgemeinen Struktur. In ähnlicher Weise wird eine Zusammenfassung der Modelle zur ultra-feinen Entitätstypisierung präsentiert, die das Problem des Mangels an annotierten Daten adressieren. Das Vokabular der ultra-feinen Entitätstypisierung umfasst zahlreiche Typen. Dies führt zu Herausforderungen für Menschen bei der manuellen Annotation und stellt somit eine zentrale Schwierigkeit in dieser Aufgabe dar. Um dieses Problem zu lösen, untersuchen wir einen unüberwachten Ansatz, der keine annotierten Trainingsdaten erfordert. Diese Studie basiert auf Informationen aus einem Graphen von Begriffen (bekannt als distributionelles Thesaurus), der wichtige Informationen über Begriffe und deren Relationen enthält. Wir analysieren, wie sich distributionell induzierte Wort- bzw. Begriffs-Sinnesinformationen durch einen solchen Graphen für die ultra-feine Entitätstypisierung nutzen lassen. Konkret untersuchen wir, wie die Label-Informationen des passenden induzierten Wort- bzw. Begriffssinns auf die Entitätsnennung angewendet werden können. Der Graph wird in dieser Studie anhand von Merkmalsinformationen von Begriffen erstellt. Dennoch gibt es verschiedene Möglichkeiten, Graphen zu konstruieren, z. B. durch die Nutzung von Seitenverlinkungen aus Wikipedia. Wir

verwenden einen solchen Graphen, um ihn in Graph-Embeddings zu transformieren und als Vektorrepräsentation von Entitäten zu nutzen. Eine der zentralen Komponenten der allgemeinen Architektur (präsentiert in unserer ersten Untersuchung) ist die Entitätsrepräsentation, mit dem Ziel, die semantische Bedeutung von Entitäten und/oder ihre semantische Verwandtschaft in verschiedenen Kontexten zu erfassen. Wir untersuchen die Rolle von Entitäts-Embeddings, die durch diese Graphinformationen konstruiert werden, und präsentieren eine einfache Technik zur Integration der strukturierten Entitätsinformationen in ein Entitätsdisambiguierungsmodell mit Graph-Embeddings.

Zusammenfassend haben wir kürzlich vorgeschlagene neuronale Modelle zur Entitätsdisambiguierung und -verknüpfung analysiert, die in der Regel eine bessere Leistung als klassische Lösungen zeigen. Diese Analyse kann neuen Forschenden in diesem Bereich helfen, die Aufgabe besser zu verstehen und zur Entwicklung zukünftiger Arbeiten beitragen. Darüber hinaus haben wir den Einfluss von Graphstrukturen auf die Entitätsdisambiguierung und ultra-feine Entitätstypisierung untersucht. Unsere Ergebnisse zeigen, dass die Nutzung von Informationen aus Graphstrukturen diese Aufgaben positiv beeinflusst. Dies könnte dazu führen, dass solche Informationen in zukünftigen Studien für verschiedene Aufgaben und Szenarien genutzt werden.

# Contents

# List of Publications

## Dissertation-Related Publications

**Özge Sevgili**, Alexander Panchenko, and Chris Biemann. 2019. Improving Neural Entity Disambiguation with Graph Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop,* 315–322. Florence, Italy: Association for Computational Linguistics.

**Özge Sevgili**, Steffen Remus, Abhik Jana, Alexander Panchenko, and Chris Biemann. 2024b. Unsupervised Ultra-Fine Entity Typing with Distributionally Induced Word Senses. In *Analysis of Images, Social Networks and Texts,* 126–140. Yerevan, Armenia, Cham: Springer Nature Switzerland.

**Özge Sevgili**, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. 2022. Neural entity linking: A survey of models based on deep learning. *Semantic Web* 13 (3): 527–570.

### About the contributions of each publication

- **Improving Neural Entity Disambiguation with Graph Embeddings (Sevgili et al., 2019)**: I implemented the experiments and wrote the paper. Alexander Panchenko closely supervised me on designing experiments, writing the paper. Chris Biemann provided comments on the research, experiments, and writing.

- **Unsupervised Ultra-Fine Entity Typing with Distributionally Induced Word Senses (Sevgili et al., 2024)**: I implemented the experiments and wrote the paper. Steffen Remus provided a code review, helped to develop ideas, design experiments, write the paper. Alexander Panchenko, Abhik Jana, and Chris Biemann also helped to develop ideas, design experiments, write the paper. Chris Biemann and all provided comments on the research, experiments, and writing.

- **A survey of models based on deep learning (Sevgili et al., 2022)**: This work was done collaboratively. I mainly researched and wrote the section or subsections; Previous Surveys, Candidate Generations, Entity Encoding, Global Context Architectures, Entity Relatedness, the table in the Appendix. Artem Shelmanov mainly researched and wrote the section or subsections; Terminological Aspects, Context-mention Encoding, Entity Ranking, Joint Entity Mention Detection and Disambiguation, Unlinkable Mention Prediction, Applications of Entity Linking. Mikhail Arkhipov mainly researched and wrote the parts; Domain-Independent Architectures, Cross-lingual Architectures. Alexander Panchenko wrote Introduction, and Knowledge Graph in Formal Definition. Other parts were written quite collaboratively, and difficult to discriminate, e.g. Summary, Goal and Scope of this Survey, etc. For instance, in Entity Linking in the Evaluation, I wrote most of the

parts of Experimental Setup, while Artem Shelmanov wrote most of the parts of Discussion of Results, yet it is still difficult to split since some parts of them were written collaboratively. Chris Biemann and all commented on my research, and writing.

I included the following sections or subsections to this thesis: Introduction (Section 1 with all subsections), Task Description including Terminological Aspects and Knowledge Graph (Section 2 with all subsections), Neural Entity Linking (Section 3 with some subsections), General Architecture (Section 3.1 with some subsubsections), Candidate Generation (Section 3.1.1), Entity Encoding (Section 3.1.3), Modifications of the General Architecture (Section 3.2 with some subsections), Global Context Architectures (Section 3.2.2), Methods that do not Fit the General Architecture (Section 3.3), Summary (Section 3.4), Evaluation (Section 4 with all subsections), Conclusion (Section 6), and the table in the Appendix. (Section numbers follow the original survey numbers.)

For Context-mention Encoding (Section 3.1.2), Entity Ranking (Section 3.1.4), Unlinkable Mention Prediction (Section 3.1.5), Joint Entity Mention Detection and Disambiguation (Section 3.2.1), Domain-Independent Architectures (Section 3.2.3), Cross-lingual Architectures (Section 3.2.4): the first paragraph, the formula (if available), sometimes the following sentence are taken to the thesis, for completeness and reference to the original work is provided.

# Other Publications

Jingyuan Feng, **Özge Sevgili**, Steffen Remus, Eugen Ruppert, and Chris Biemann. 2020. Supervised Pun Detection and Location with Feature Engineering and Logistic Regression. In *Proceedings of the 5th Swiss Text Analytics Conference and the 16th Conference on Natural Language Processing,* Zurich, Switzerland.

Robert Günzler, **Özge Sevgili**, Steffen Remus, Chris Biemann, and Irina Nikishina. 2024. Sövereign at The Perspective Argument Retrieval Shared Task 2024: Using LLMs with Argument Mining. In *Proceedings of the 11th Workshop on Argument Mining (ArgMining 2024),* 150–158. Bangkok, Thailand: Association for Computational Linguistics.

Irina Nikishina, **Özge Sevgili**, Mahei Manhai Li, Chris Biemann, and Martin Semmann. 2025. Creating a Taxonomy for Retrieval Augmented Generation Applications. arXiv: 2408.02854 [cs.IR].

**Özge Sevgili**, Irina Nikishina, Seid Muhie Yimam, Martin Semmann, and Chris Biemann. 2024a. UHH at AVeriTeC: RAG for Fact-Checking with Real-World Claims. In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER),* 55–63. Miami, FL, USA: Association for Computational Linguistics.

Gopalakrishnan Venkatesh, Abhik Jana, Steffen Remus, **Özge Sevgili**, Gopalakrishnan Srinivasaraghavan, and Chris Biemann. 2022. Using distributional thesaurus to enhance transformer-based contextualized representations for low resource languages. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing (ACM SAC), Special Track on Knowledge and Natural Language Processing (KNLP),* 845–852. Online.

# List of Figures

# List of Tables

# List of Abbreviations

**BERT** . . . . . Bidirectional Encoder Representations from Transformers.

**CBOW** . . . . . Continuous Bag-of-Word.

**CNN** . . . . . Convolutional Neural Network.

**DT** . . . . . Distributional Thesaurus.

**ED** . . . . . Entity Disambiguation.

**EL** . . . . . Entity Linking.

**ET** . . . . . Entity Typing.

**FET** . . . . . Fine Grained Entity Typing.

**FFNN** . . . . . Feedforward Neural Network.

**GRU** . . . . . Gated Recurrent Unit.

**IE** . . . . . Information Extraction.

**KB - KG** . . . . . Knowledge Base - Knowledge Graph (used interchangeably).

**LM** . . . . . Language Modeling.

**LLM** . . . . . Large Language Model.

**LSTM** . . . . . Long Short-Term Memory.

**MD** . . . . . Mention Detection.

**MLM** . . . . . Masked Language Modeling.

**NER** . . . . . Named Entity Recognition.

**NLI** . . . . . Natural Language Inference.

**NLP** . . . . . Natural Language Processing.

**NLU** . . . . . Natural Language Understanding.

**PLM** . . . . . Pre-trained Language Modeling.

**RAG** . . . . . Retrieval Augmented Generation.

**ReLU** . . . . . Rectified Linear Unit.

**RDF** . . . . . Resource Description Framework.

**RNN** . . . . . Recurrent Neural Network.

**SBERT** . . . . . Sentence BERT.

**seq2seq** . . . . . Sequence-to-Sequence.

**SGD** . . . . . Stochastic Gradient Descent.

**UFET** . . . . . Ultra-Fine (Grained) Entity Typing.

**WSD** . . . . . Word Sense Disambiguation.

# 1 Introduction

**In this chapter, our main tasks of Entity Disambiguation and Ultra-Fine Entity Typing are introduced. We discuss their similarities, differences, and interactions. We explain the motivations for our approaches to the tasks. Finally, we detail the contributions of the research conducted for the thesis and lay out the organization of the thesis.**

## Contents

## 1.1 Introduction

People are incorporating technological advancements in artificial intelligence into their daily lives. Consider, traveling in a foreign country without knowing its local language, one can rely on machine translation application to communicate. Similarly, artificial-intelligence-based recommendation systems can suggest movies and books based on personal preferences. Such applications have been developed through different research fields, such as natural language processing (NLP), a subfield of computer science, focusing on an automatic human language processing. In NLP, there exist various tasks, e.g. machine translation, spam classification, sentiment analysis, and many more, relying on artificial intelligence algorithms. Such algorithms require knowledge, which can be acquired from a large amount of data available on the web through e.g. blog posts, social media posts, encyclopedic contents, etc.

Yet, considering that high amount of available text, finding the related information is not always straightforward. For example, if one wants to reach the population information about `Ottawa`, it should be clear that `Ottawa` refers to either the capital city[1] of Canada or a city[2] in United States. Thus, the exact reference for `Ottawa` is required to reach the right information.

For this specific challenge, we focus on two NLP tasks, i.e. *Entity Disambiguation*: aiming to provide a knowledge base reference for the mentioned text, e.g. for `Ottawa`, providing a link reference https://en.wikipedia.org/wiki/Ottawa, and *Entity Typing*: assigning a list of type labels, for instance `capital`, `city`, or `small_town` for `Ottawa` that would be helpful to distinguish it. In the scope of this thesis, we study on these tasks in several aspects.

We review recent entity disambiguation (and linking) models to systematically analyze these techniques by providing a generic solution applicable to most of such models, covering their components, modifications and features. Similarly, we present a summary for entity typing models and discuss the challenges with the focus on fine/ultra-fine grained labels, e.g. `capital`, since they are more informative as in the `Ottawa` example that helps to clarify the reference. One of the key challenges in this task is a lack of annotated data. For this challenge, we automatically generate labels based on a graph constructed from the features of terms. Similarly, for the entity disambiguation task, we utilize graph information, this time built from page links in Wikipedia, by transforming it into another form to effectively leverage structural information.

Overall, in this thesis, we focus on the annotation of entities with two NLP tasks, which helps a clarification of the references for the mentioned text and thus, enables several applications, e.g. finding a related information from a large amount of data. In our survey, we provide a systematization of various features for recent neural entity disambiguation and linking techniques with categorizations. In our entity typing work, we present an unsupervised solution with the help of graph information. Finally, we describe a method to integrate the graph information into an entity disambiguation model. In the next sections, we will describe the tasks and discuss our motivations while conducting each study.

1. https://en.wikipedia.org/wiki/Ottawa
2. https://en.wikipedia.org/wiki/Ottawa,_Illinois

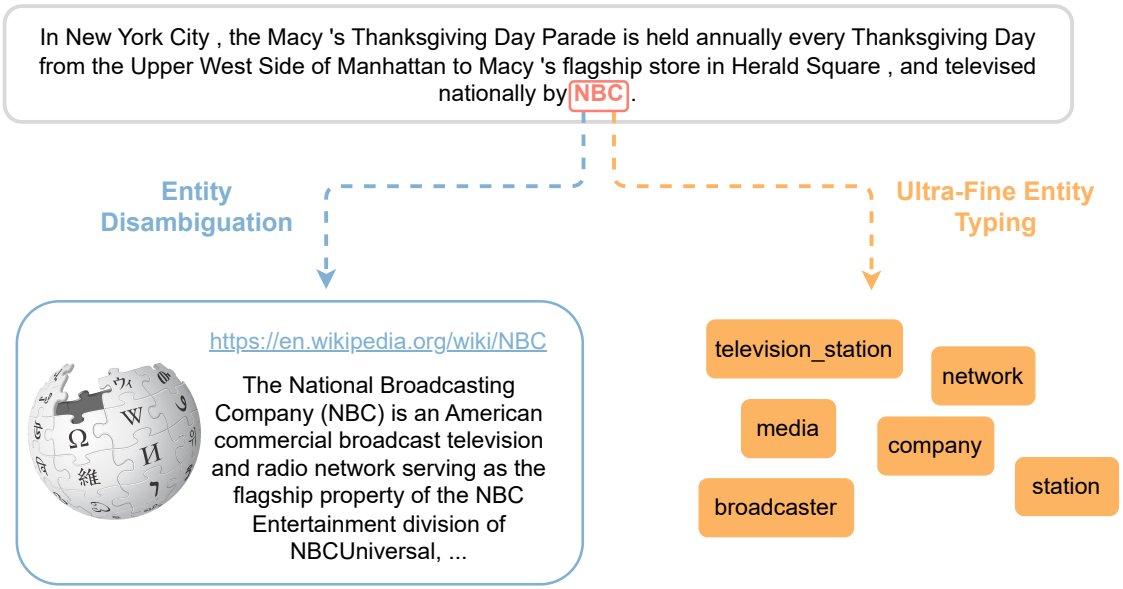


**Figure 1.1: Entity Disambiguation and Ultra-Fine Entity Typing tasks**: matching the mention NBC to the right knowledge base record, i.e. "National Broadcasting Company" rather than e.g. "National Baseball Congress" – entity disambiguation, and assigning free-form type labels for the mention – ultra-fine entity typing.

## 1.2 Entity Disambiguation and Ultra-Fine Entity Typing

Human languages are ambiguous (Navigli, 2009) in several aspects, e.g. syntactic ambiguity, semantic ambiguity, etc. Consider the following examples for syntactic and semantic ambiguities. The example by Bailey et al. (2015), "eat spaghetti with chopsticks." has two interpretations; either chopsticks modifying the verb, *eat* (correct) or chopsticks modifying the noun, *spaghetti*, where the interpretion of "spaghetti with chopsticks" is as a meal (incorrect), therefore it is syntactically ambiguous. Consider the following two sentences for semantic ambiguity by Navigli (2009) "I can hear *bass* sounds." and "They like grilled *bass*.", where the meaning of the *bass* differ in each sentence. Automatically resolving such ambiguities has been a challenging issue, as it requires an understanding of text. There exist several tasks dedicated to help these problems, e.g. dependency parsing (Jurafsky and Martin, 2024; Chen and Manning, 2014), word sense disambiguation (Navigli, 2009), entity disambiguation/linking (Jurafsky and Martin, 2024), etc. Similar to words, entities can be ambiguous, i.e. same entity mention might refer different entities (Balog, 2018). The focus of this thesis is on this issue and, specifically, entity disambiguation and typing tasks.

**Entity Linking (EL)** *"is the task of associating a mention in text with the representation of some real-world entity in an ontology or knowledge base."* as described by Jurafsky and Martin (2024). EL encompasses two tasks: mention detection, which detects a mention in text and **Entity Disambiguation (ED)**, which maps a mention to an entity in a knowledge base. Throughout this thesis, we mostly focus on entity disambiguation. Consider the example in Figure 1.1, here NBC is a mention and it is ambiguous, as it can be abbreviation of various diverse entities, e.g. "National Baseball Congress"[3], "National

3. https://en.wikipedia.org/wiki/National_Baseball_Congress

Bus Company (UK)"[4], "Nürnberger Basketball Club"[5], "National Bank of Cambodia"[6], "National Business Center"[7], etc. (see disambiguation page[8] for more references). The goal of entity disambiguation is finding out the right reference among all knowledge base entries, i.e. `NBC`[9] - National Broadcasting Company, in the running example with the help of contextual information.

**Ultra-Fine Entity Typing (UFET)** "given a sentence and an entity mention $e$ within it, the task is to predict a set of natural-language phrases $T$ that describe the type of $e$" as described by Choi et al. (2018). For the example mention `NBC`, it assigns types of `media, network, company, station, broadcaster, television_station` among numerous types in the type vocabulary, which contains over 10k types (Choi et al., 2018). One can infer that `NBC` is for "National Broadcasting Company" rather than e.g. "Nürnberger Basketball Club" or "National Baseball Congress" with these annotations. It seems that the finer the types become, the more information is delivered about the mention. For instance, `television_station` or `broadcaster` are helpful annotations to understand what the mention refers to.

There exist similarities, differences and interactions between these two tasks. We will discuss them in the following paragraphs. The details of the tasks are discussed, individually, in Chapters 3 and 4.

### Similarities

- Both tasks help solving ambiguity at entity level and increase text understanding, as shown in the example.
- Inputs are the same for both of the tasks: context and mention. That means they do not deal with determining mention boundaries and assume these boundaries are given already as inputs.

### Differences

- The output for entity disambiguation is single, i.e. one knowledge base entry (sometimes NIL) per mention. In ultra-fine entity typing, the output can be multiple for each mention, with distinct type labels (Murty et al., 2018), as also shown in Figure 1.1.
- Label set in UFET is predefined vocabulary of types, instead the label set or annotation set is all knowledge base entries in entity disambiguation.
- Commonly, ED is treated as ranking problem, although UFET is mostly considered as multi-label multi-class classification. Note that these are common behaviours, there are some models resolving the tasks, in different ways.
- In UFET, labels are expected to be context-dependent, that is the ability to assign, e.g. "philanthropist" to "Bill Gates" based on the context. Hence, the type label

4. https://en.wikipedia.org/wiki/National_Bus_Company_(UK)
5. https://en.wikipedia.org/wiki/N%C3%BCrnberger_Basketball_Club
6. https://en.wikipedia.org/wiki/National_Bank_of_Cambodia
7. https://en.wikipedia.org/wiki/National_Business_Center
8. https://en.wikipedia.org/wiki/NBC_(disambiguation)
9. https://en.wikipedia.org/wiki/NBC

should be appropriate for the role the target entity plays in the sentence (Choi et al., 2018). The main concentration of ED is to map this mention to a knowledge base entry, e.g. "Bill Gates"[10] rather than "Bill Gates (footballer)"[11] and solve the ambiguity.

- Mentions can be in different forms in UFET, i.e. named entities, nominals, and pronouns. For more discussion, see Chapters 3 and 4, for ED and UFET, respectively.

**Interactions**

- The tasks help each other, more specifically, entity types are helpful in entity linking/disambiguation (e.g. Sui et al. (2022)). Likewise, entity linking/disambiguation is also beneficial for entity typing (e.g. Dai et al. (2019)). There are some models trying to resolve such tasks jointly (e.g. Durrett and Klein (2014)). In the following items, these interactions are discussed.

  1. **Entity Typing in Entity Linking**
     In the literature, entity type information is found helpful for the entity linking/disambiguation task and leveraged in various ways to solve the entity linking/disambiguation task, e.g. (Sui et al., 2022; Hou et al., 2020; Onoe and Durrett, 2020; Gupta et al., 2017; Raiman and Raiman, 2018; S Chen et al., 2020; Bhargav et al., 2022; Tianran et al., 2021), interalia.
     Some of them utilize entity typing in zero-shot entity linking, and/or domain-independent setting, e.g. (Sui et al., 2022; Bhargav et al., 2022; Onoe and Durrett, 2020).
     Some studies leverage entity type information in entity representations, e.g. (Hou et al., 2020; Tianran et al., 2021; S Chen et al., 2020; Gupta et al., 2017), etc.
     Other than them, there are different solutions to use entity type information in EL, e.g. (Raiman and Raiman, 2018).
  2. **Entity Linking in Entity Typing**
     Similarly, there exist several works that leverage entity linking in an entity typing model. For instance, Dai et al. (2019) utilize the type information of the knowledge base obtained through entity linking in a fine-grained entity typing model.
  3. **Joint Entity Linking and Typing**
     Some authors aim to solve these two (or more) tasks jointly. Durrett and Klein (2014) provide a joint model of coreference resolution, entity linking and coarse semantic (entity) typing using the interactions of these tasks. Their motivation is that these tasks are heavily interdependent, and one task can help to solve other tasks. For example, knowing the Wikipedia reference through entity linking can help to resolve ambiguities in the entity types and coreference resolution.
     Note that in some other studies training of entity linking and entity typing can be done simultaneously, by taking e.g. entity linking as a primary task and entity typing as an auxiliary task, e.g. (Bhargav et al., 2022).

10. https://en.wikipedia.org/wiki/Bill_Gates
11. https://en.wikipedia.org/wiki/Bill_Gates_(footballer)

4. **Integration Knowledge Graph (KG) information into Language Models (LMs)** Recent language models, including pre-trained and large ones, have shown promising performance in many tasks (WX Zhao et al., 2024). Furthermore, there are many works to incorporate knowledge information into such models for better language understanding and to recover factual knowledge (Z Zhang et al., 2019; Peters et al., 2019). There are various methods to integrate knowledge information, e.g. proposed by Z Zhang et al. (2019), Peters et al. (2019), Ruize Wang et al. (2021), Févry et al. (2020), T Sun et al. (2020), X Wang et al. (2021), Yamada, Asai, Shindo, et al. (2020), and J Wang et al. (2022), interalia. Indeed, these works do not directly involve interactions between entity linking and typing, however some leverage entity linking to integrate knowledge and most use entity typing for evaluation. Thus, we think it is worth to mention them briefly.

   For example, KnowBERT (Peters et al., 2019) incorporates knowledge graph information into pre-trained BERT architecture (Devlin et al., 2019) via entity linker component. More recently, J Wang et al. (2022) propose a model, which leverage knowledge information through prompts.

   The majority of such knowledge-enhanced (pre-trained) language models have evaluations with downstream-tasks, including entity typing, however mostly with less types.

## 1.3 Motivation

**Natural Language Processing (NLP)** is, as defined by Eisenstein (2018), "*the set of methods for making human language accessible to computers*". There are numerous NLP tasks, each aiming to tackle different challenges, e.g. part-of-speech tagging, sentiment analysis, machine translation, question answering, summarization, just to name a few. Some of these tasks are increasingly integrated into a daily life; for instance machine translation is widely employed for language translations and recently ChatGPT[12] – a chatbot built by the OpenAI team based on Large Language Models (LLMs)– is commonly used for various tasks.

Many diverse techniques have been proposed to solve various NLP tasks, including hand-crafted feature based models, shallow architecture based methods, neural architecture based techniques, etc. Recent neural-networks-based models have shown very promising performance, achieving state-of-the-art results on many NLP problems (Goldberg, 2016; **Sevgili et al.**, 2022; J Li et al., 2022). In contrast to shallow models, neural network and/or deep learning contains non-linear layers, and this non-linearity allows models to learn complex features (J Li et al., 2022).

The analogous trend of neural models succeeding in comparison to classical models also exists in the entity linking and disambiguation task, as shown in Figure 1.2. In this Figure, we show entity disambiguation performances of classical and neural models based on one of the common dataset (more discussion in Chapter 3). Based on these results, although the classical models (orange squares) have shown good performance, the neural ones (blue circles) have outperformed them.

12. https://openai.com/blog/chatgpt/

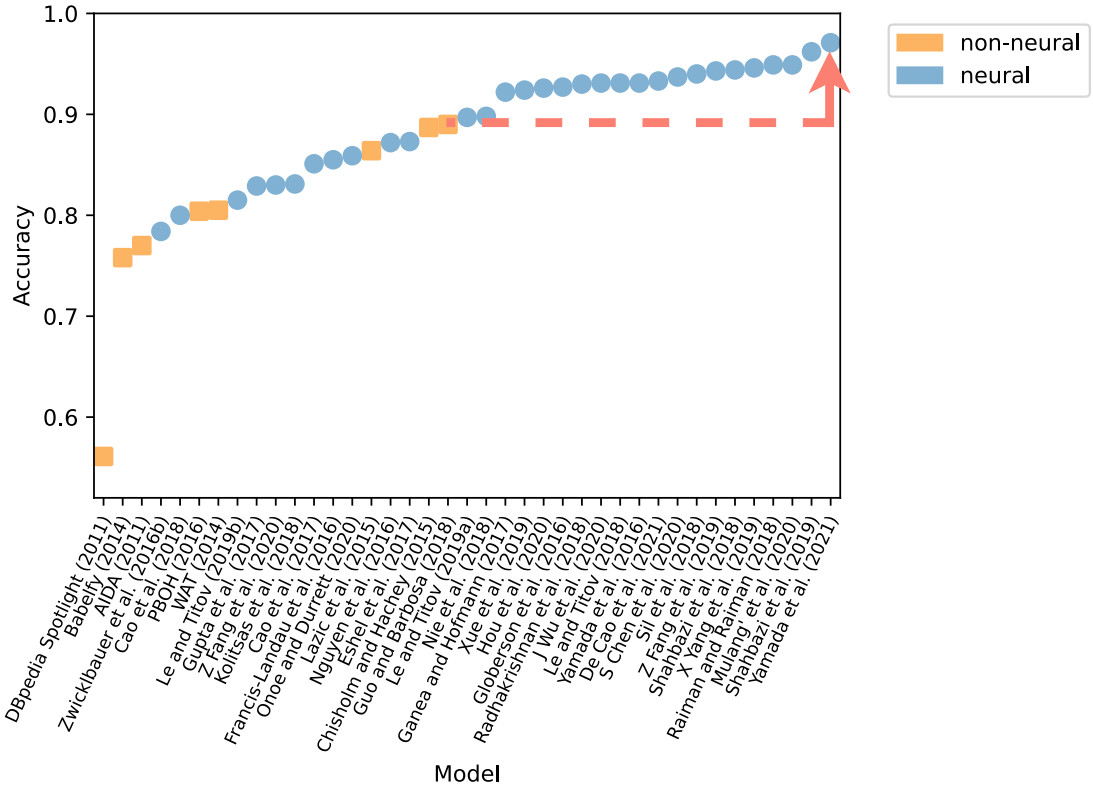


**Figure 1.2: Entity disambiguation: classic and neural models**. Performance of the classic entity linking models ( orange squares) with the more recent neural models ( blue circles) on one of the common dataset (i.e. AIDA test set) shows an improvement, highlighted by arrow. (For more discussion, see Figure 3.7, in Chapter 3.)

Due to this success of neural networks, many researchers have turned their focus on entity linking and disambiguation into neural model-driven solutions. We are motivated by this to provide a comprehensive review of neural models that have emerged with this recent wave with the following research question.

**Research Question**

How can neural entity linking and disambiguation models be analyzed and their features systematized?

Chapter 3 handles this question with a survey of neural based models for entity linking and disambiguation.

In a similar vein, numerous techniques have been provided in the ultra-fine entity typing task. In Chapter 4, we present a brief summary of these methods. Instead of neural-based solutions, we focus on the models providing solutions to data scarcity issue in UFET, in this overview. There are a large number of types in the ultra-fine entity typing task, namely over 10k type set (Choi et al., 2018). Thus, a manual creation of labeled data is quite difficult for human due to that large number of types (Dai et al., 2021). Therefore, a lack of human-annotated data is a critical challenge, in this task.

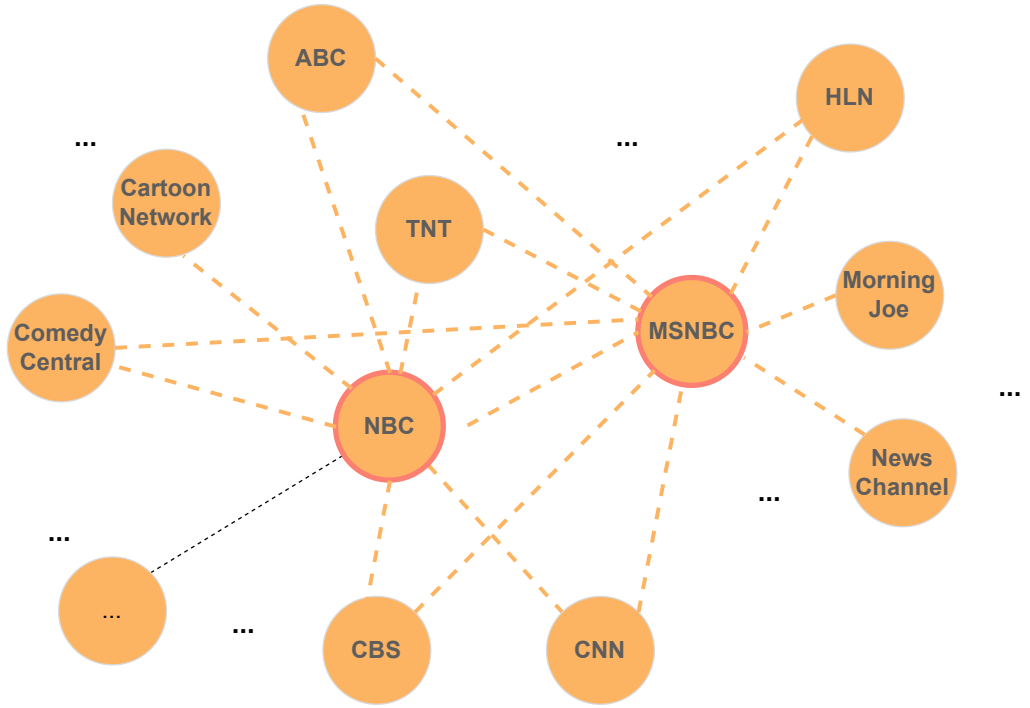


**Figure 1.3: Example part of a JoBimText graph.** Nodes are some similar terms of NBC, and MSNBC. The information is taken via JoBimText API (Ruppert et al., 2015). Note that there are much more nodes and edges related to searched terms NBC, and MSNBC, only a few are shown for illustrative purposes.

In the context of this particular challenge, we conduct an exploration for an unsupervised solution that requires no annotated samples, yet relying on information from a graph.

Graphs are, as in general, applied in various ways within NLP to resolve many tasks, including word sense disambiguation, part-of-speech tagging, dependency parsing, summarization, and many more (Nastase et al., 2015). Graphs can be constructed in different ways based on the requirements, e.g. nodes as word, sentence, document, etc. or edges as co-occurrence, similarity, etc. (Nastase et al., 2015). For instance, a distributional thesaurus consists of terms as nodes, as shown in Figure 1.3, where information is from distributional thesaurus built by Biemann and Riedl (2013). In this example, some similar terms of NBC and MSNBC are shown.

We are motivated by such graph information and their applications to investigate usefulness of this information in UFET task with the following research question.

**Research Question**

Is the information from a graph constructed through the terms (i.e. distributional thesaurus) beneficial to solve ultra-fine entity typing in an unsupervised way?

In Chapter 5, we carry out research on unsupervised UFET leveraging graph information. In this work, we utilize the word/term sense information that is distributionally induced from the distributional thesaurus in JoBimText framework (Biemann and Riedl,

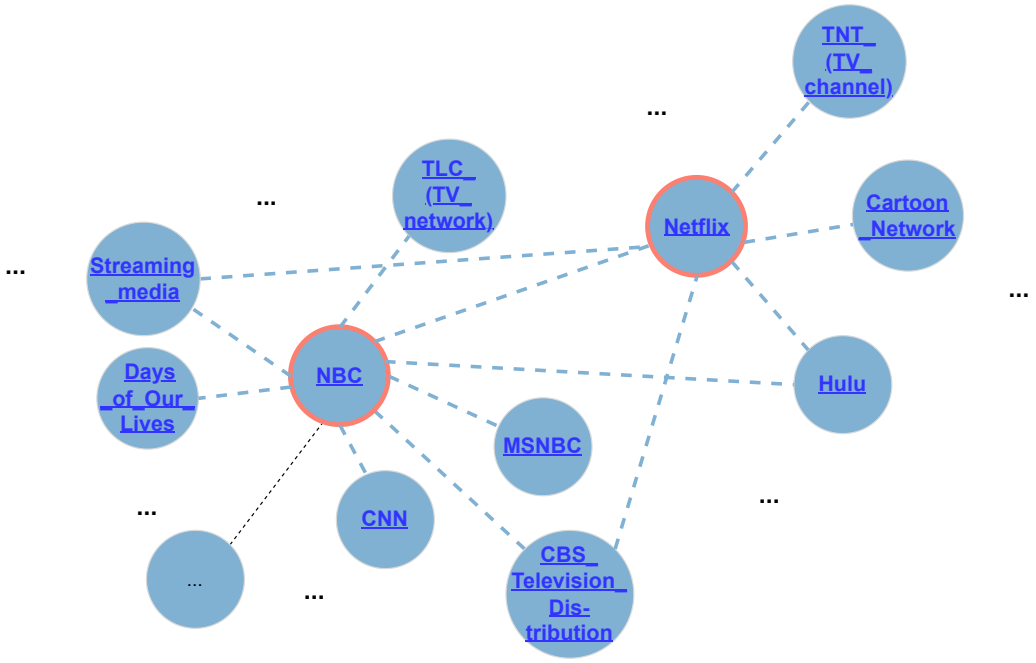


**Figure 1.4: Example part of a DBpedia graph built by page links.** Nodes are some similar entities of NBC entity, and Netflix entity. The information is taken from DBpedia page links [13]. Note that there are much more nodes and edges related to searched entities of NBC, and Netflix, only a few are shown for illustrative purposes.

2013), and the labels (which are also provided through JoBimText) of the appropriate induced word/term sense to the entity mention.

As discussed, nodes and/or edges can differ based on the application scenarios. It is also worth noting that in Chapter 3, we present a generic architecture for neural based EL. One of its core components is entity embeddings. Some strategies have been proposed to create entity embeddings for this task.

Motivated by the significance of entity embeddings in entity disambiguation and linking, and the flexibility of constructing graph through various ways, we conduct a research to investigate the impact of the entity embeddings created by graph information to deal with the following research question.

**Research Question**

Is structural information in the form of graph embeddings helpful for entity disambiguation?

In Chapter 6, we construct a graph through page link information between entities, i.e. nodes are entities, edges refer to links between them based on DBpedia (Lehmann et al., 2015) information, as shown in Figure 1.4. In this figure, we show some similar entities of NBC entity and Netflix entity, as an example. This graph is transformed into vector representations using a graph embedding algorithm. We explore the role of such graph embeddings in entity disambiguation task, in this work.

13. http://downloads.dbpedia.org/2016-10/core-i18n/en/page_links_en.ttl.bz2

## 1.4 Contributions and Organization of the Thesis

In the following items, we list the contributions of this work, under their respective chapters. We summarize contents of each chapter. This also covers an organization of the thesis.

**Chapter 2 - Background of Underlying Technologies**

- **Summary**: In this chapter, we present a brief summary of the technologies that are utilized in the conducted studies. For instance, neural networks including feedforward neural networks, recurrent neural networks etc. is summarized. In the scope of this thesis, we leverage several types of representations, thus a summary of representation learning also presented. Furthermore, we discuss the information sources, e.g. DBpedia (Lehmann et al., 2015), JoBimText (Biemann and Riedl, 2013).

**Chapter 3 - Survey of Entity Linking Methods**

- **Summary**: We comprehensively describe recent neural entity linking (EL) and disambiguation (ED) models. We discuss design features of these models. A generic architecture that is valid for most of the models is presented. Its essential components and modifications are discussed. We provide a summary of prominent methods for each essential component.
- A survey of state-of-the-art neural entity linking and disambiguation models.
- A systematization of various features of neural techniques for EL and ED with categorizations.
- A discussion for evaluation results of recent EL and ED methods on popular benchmarks.
- A summary of different entity embedding techniques.

**Chapter 4 - Survey of Entity Typing Methods**

- **Summary**: We discuss the challenges of ultra-fine entity typing task, for example, different mention forms, context dependency, lack of data, etc. Promising research lines are discussed with the references. We summarize the models especially for addressing data scarcity issue.
- An introduction to ultra-fine entity typing task with the discussion of different challenges.
- A summary of proposed models that deals with lack of the data issue in ultra-fine entity typing.

**Chapter 5 - Unsupervised Ultra-Fine Entity Typing**

- **Summary**: We present an unsupervised way of solution to ultra-fine entity typing for the lack of annotated data issue. This solution is based on distributionally induced word senses. Experimental set-up is explained and results are shown.
- An investigation of an unsupervised ultra-fine entity typing technique.
- A use of label information from distributional thesaurus graph in ultra-fine entity typing task.

**Chapter 6 - Supervised Entity Disambiguation with Graph Embeddings**

- **Summary**: We describe a method that leverages graph embeddings to integrate structured graph information from the knowledge base with unstructured information from text-based representations. Experimental configuration is described and results are discussed.
- A creation of a simple technique for an integration of a structured graph information into an ED model with graph embeddings.
- A straightforward solution to leverage a large structured graph information based on graph embeddings.

**Chapter 7 - Conclusion**

- **Summary**: In this chapter, we discuss the possible influences of each study. We explain the limitations of the work described in this thesis. Furthermore, future directions are outlined.

# 2

# Background of Underlying Technologies

**In this chapter, we summarize background information for underlying technologies as a backbone of the following chapters. We shortly introduce relevant parts of neural networks that are important for other chapters. We also present a summary of representation learning. We discuss knowledge resources that are leveraged in the experiments.**

## Contents

## 2.1 Introduction

The key challenge of artificial intelligence, as in general, is to attempt to solve tasks that people solve intuitively (Goodfellow et al., 2016). Machine learning algorithms aim to solve such tasks using data with the ability of learning (Goodfellow et al., 2016).

Learning algorithms can be distinguished into several categories, supervised, unsupervised zero-shot, distant learning. Note that there exist several more learning techniques, yet in the scope of this thesis, we focus on the mentioned ones.

**Supervised Learning** The algorithms aim at learning input-output pairs from data that are in the form of input examples and their corresponding outputs (Goodfellow et al., 2016). For such algorithms, examples should be annotated with outputs labels in advance.

**Unsupervised Learning** The goal of the algorithms is to learn useful properties of the structure of data (Goodfellow et al., 2016). The data examples are not annotated, in this type of learning.

**Zero-Shot Learning** The purpose of this learning is to observe classes that might not be seen at training time (Xian et al., 2019). For some classes, no training data would be available, yet the descriptions are available (Larochelle et al., 2008).

**Distant Supervision/Learning** The goal is to automatically create a training data relying on some heuristics (Hoffmann et al., 2011; Mintz et al., 2009; Le and Titov, 2019a) and use it for supervision. Some authors use **weak** and distant supervision interchangeably, e.g. (Le and Titov, 2019b).

## 2.2 Neural Networks

Traditional machine learning models have shown good performance on many tasks, however they have some specific limitations. For example, consider a linear model, e.g. linear regression, which are able to model linear functions, however, not every function can be represented with linear models, such as the XOR function that outputs 1 if one of two inputs is 1, or outputs 0 otherwise (Goodfellow et al., 2016). Neural network models are able to cope with such cases with non-linear (activation) functions and learned parameters.

In this section, most of the information, equations, and figures are taken and/or adapted from Goodfellow et al. (2016), Nielsen (2015), and Goldberg (2016), lectures (lecture slides and notes)[1]. We shortly summarize some relevant parts of neural network, for better understanding of next sections, and we give a name for several design choices (like e.g. activation, loss, optimization functions) just for a reference.

1. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1214/, and https://cs231n.github.io/neural-networks-1/

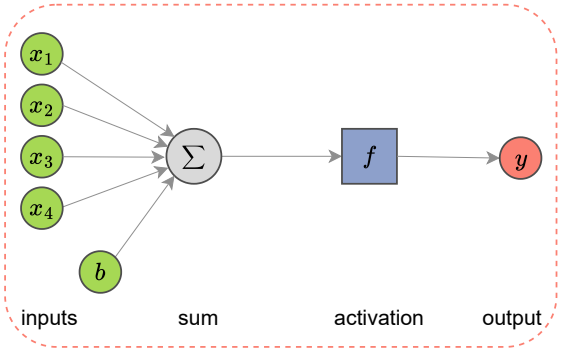


**Figure 2.1**: A single neuron that computes $y$ by multiplying inputs with weights, summing them, and applying an activation (non-linear) function.

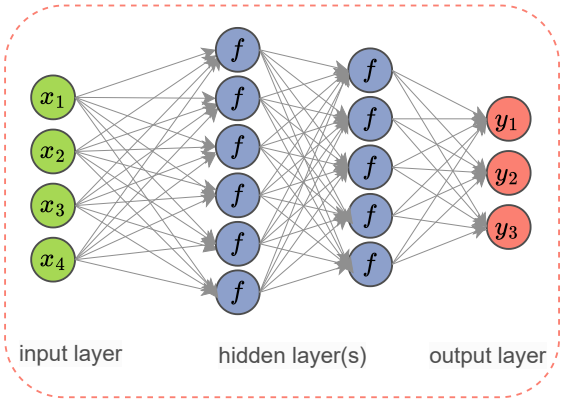


**Figure 2.2**: An example feedforward neural network with two hidden layers. Figure is from Goldberg (2016).

### 2.2.1 Feedforward Neural Networks

An essential component of neural networks is a neuron, which computes an output $y$ from the input $x$ by multiplying with weights $W$, summing them, and applying a non-linear activation function $f$[2], as shown in Figure 2.1.

$$y = f(xW + b). \tag{2.1}$$

**Activation Functions** There are various activation functions, e.g. commonly, **sigmoid**: $\sigma(x) = 1/(1 + exp(-x))$, **tanh**: $tanh(x) = (exp(x) - exp(-x))/(exp(x) + exp(-x))$. Another popular choice is the ReLU (rectified linear unit) function $relu(x) = max(x, 0)$ despite its simplicity.

Feedforward Neural Network (FFNN) contains neurons in a way that there are no feedback connections (Goodfellow et al., 2016). Neurons in the layers do not have any connections within the layer, as shown in Figure 2.2, where the number of hidden layers can be increased or the number of neurons in the hidden layer(s) can be changed, and

2. Notation is adapted from Goldberg (2016)

the number of outputs can be more or less depending on the problem. Usually, the layers are fully connected. Consider the weights between the input $x$ and the hidden layer as $W^1$, $W^2$ between the hidden layers, $W^3$ between the last hidden layer and the output $y$[3].

$$y = f^2(f^1(xW^1 + b^1)W^2 + b^2)W^3. \tag{2.2}$$

where $b^1, b^2$ are biases, and $f^1, f^2$ are activation functions. Note that the non-linear activation function is important, since if it is not applied, the neural network can only represent linear functions of the input (Goldberg, 2016).

**Loss Functions** The prediction $y$ can be calculated as discussed. The requirement is to evaluate how close it is to the actual output, which is achieved by loss functions (also called cost or objective functions) (Nielsen, 2015). There are many different types of loss functions, e.g. mean squared error, cross-entropy, inter alia.

**Stochastic Gradient Descent and Back-propagation** The network has completed its forward pass and computed the loss between its prediction $y$ and the actual output. The goal is to approximate the prediction to the output, which means to minimize the loss function. For this purpose, good parameters that lead to a small loss should be found (Nielsen, 2015). One way to do it is to update the weights and biases (and sometimes inputs) with small changes and repeat it many times (Nielsen, 2015). Mostly, this is achieved with the gradient descent (commonly stochastic gradient descent (SGD) (Robbins and Monro, 1951) algorithm – extension of gradient descent), with the following equation[4].

$$\theta \leftarrow \theta - \epsilon g, g = \nabla_\theta J(\theta). \tag{2.3}$$

where, $\epsilon$ is the learning rate, $\theta$ are the parameters, and $J(\theta)$ is the loss. $\nabla_\theta J(\theta)$ is a gradient function of the loss function with respect to the parameters, which is calculated using the back-propagation algorithm (Goodfellow et al., 2016). The back-propagation algorithm computes the chain rule recursively.

For instance, the gradient of $z$ function with respect to $x_i$, $y = g(x), z = f(y)$ with chain rule (example is from Goodfellow et al. (2016)):

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j}\frac{\partial y_j}{\partial x_i}. \tag{2.4}$$

There are some adaptive optimization algorithms, in which the learning rates can change per parameter, such as Adam (Kingma and Ba, 2014), Adagrad (Duchi et al., 2011), etc.

Overall, replacing the linear models with a feedforward neural network in some NLP tasks, like e.g., syntactic parsing, sentiment classification, has shown good results (Goldberg, 2016).

3. Notation is fromGoldberg (2016)
4. Notation is from Goodfellow et al. (2016)

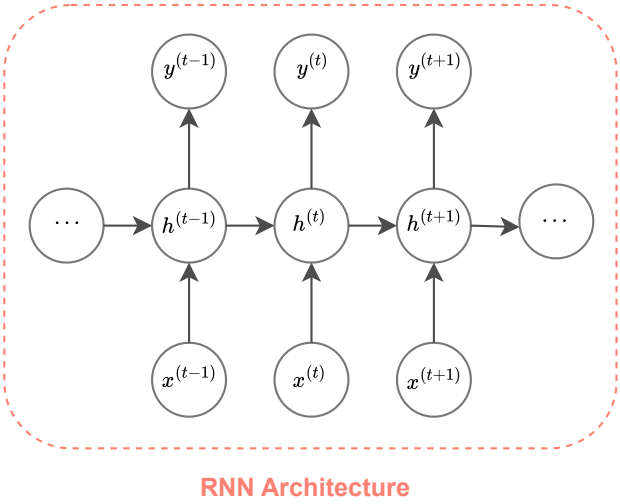


**Figure 2.3**: RNN architecture. Inputs and outputs of RNN, where each input $x^{(t)}$ corresponds to one sequence (e.g. word), $h^{(t)}$ is for the hidden state, and $y^{(t)}$ refers to the prediction. The figure is from Goodfellow et al. (2016).

### 2.2.2 Recurrent Neural Networks

Although feedforward networks can achieve good scores on some NLP tasks, there are still some limitations for sequential data (e.g. words, sentences, etc.). FFNNs allow for a fixed size window of inputs, and thus cannot scale for longer sequences. Furthermore, the position/order of sequences is not considered in such networks (Goodfellow et al., 2016; Goldberg, 2016). Recurrent Neural Networks (RNNs) are specialized to process sequential data. The idea is to share parameters, which allows generalization to different sequence lengths and obtain position information (Goodfellow et al., 2016). RNNs have the recurrent connections between hidden units $h$, as shown in Figure 2.3, where each $x^{(t)}$ is one sequence (e.g. word), $h^{(t)}$ is hidden state, and $y^{(t)}$ is prediction, at any time $t$.

Recurrent Neural Networks have achieved high scores in numerous NLP tasks (Goldberg, 2016). For instance, Mikolov et al. (2010) present experiments with RNN-based language model. As in general, language modeling (LM) is a task to predict the probabilities of future or missing words or tokens (WX Zhao et al., 2024).

**Long Short-Term Memory (LSTM)** RNNs face the problem of vanishing gradient. Gradients in the later steps for long sequences fade away during the back-propagation phases until reaching the earlier steps (Goldberg, 2016). LSTM (Hochreiter and Schmidhuber, 1997; Gers et al., 2000) overcomes the obstacle with the "memory cells" mechanism to preserve information, which are controlled with three gates to decide what to write (input), forget, and output (Goldberg, 2016). With these controls, the gradients can stay in the cells longer.

As an alternative to LSTM, the Gated Recurrent Unit (GRU) is designed to reduce complexity a bit to allow for easy analysis with fewer gates and no memory component (Goldberg, 2016).

Several architectures are designed by combining RNNs, e.g. bidirectional RNNs and sequence-to-sequence models, as discussed below.

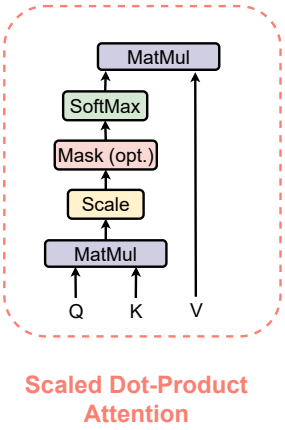


**Figure 2.4**: A scaled dot-product attention part in the transformer architecture, from Vaswani et al. (2017).

**Bidirectional RNNs** The recurrent models explained until now consider the information of the left sequence. Yet, the whole sequence information should be useful for a better prediction if we have an access to it (Goodfellow et al., 2016). Bidirectional RNNs address it with a simple idea: combining or concatenating two RNNs, one of which is the same as shown here, starting from beginning to end, and the other one starting the sequence backward, from the end to beginning (Goodfellow et al., 2016).

**Sequence-to-sequence models** There are challenging NLP problems, requiring a sequential output from the sequential input, where the lengths of input and output may vary like machine translation or question answering (Goodfellow et al., 2016). Sequence-to-sequence (seq2seq or encoder-decoder) model was introduced by Sutskever et al. (2014), consisting of two RNNs (or LSTMs), i.e. the encoder RNN and the decoder RNN. The encoder RNN reads the input sequence and produces a fixed-dimensional representation for it; decoder RNN conditions on this vector to generate the target sequence (Goodfellow et al., 2016). However, the vector produced by the encoder should carry all the information of the source sequence, which causes a bottleneck problem (Bahdanau et al., 2015). To solve it, the attention mechanism (Bahdanau et al., 2015) is applied between encoder and decoder.

### 2.2.3 Transformer

Despite the successful applications of RNNs in NLP, sequential computation prevents it from parallel operations[5]. The transformer architecture (Vaswani et al., 2017) is designed to provide significantly more parallelization without relying on recurrence instead completely depending on an attention mechanism.

The idea of the transformer architecture is to replace recurrence with attention to grasp global dependencies. This attention is not always from decoder to encoder, as in seq2seq, but also inside a single sentence, i.e. self-attention. Attention is in general a function that takes the vectors of the query, keys, values, and produces the output vector by computing the weighted sum of the values, where, the weights are the scores between the query and each key (Vaswani et al., 2017).

5. https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

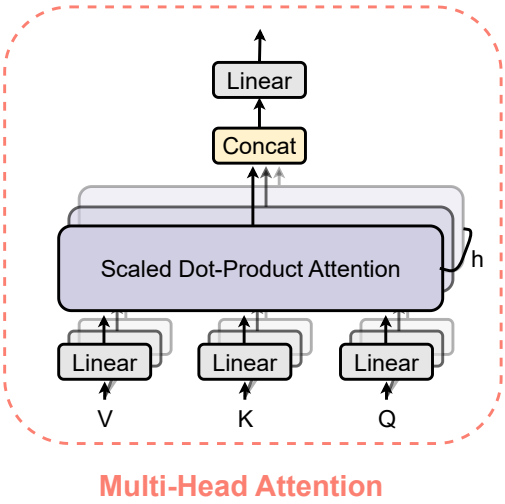


**Figure 2.5:** A multi-head attention part in the transformer architecture, from Vaswani et al. (2017).

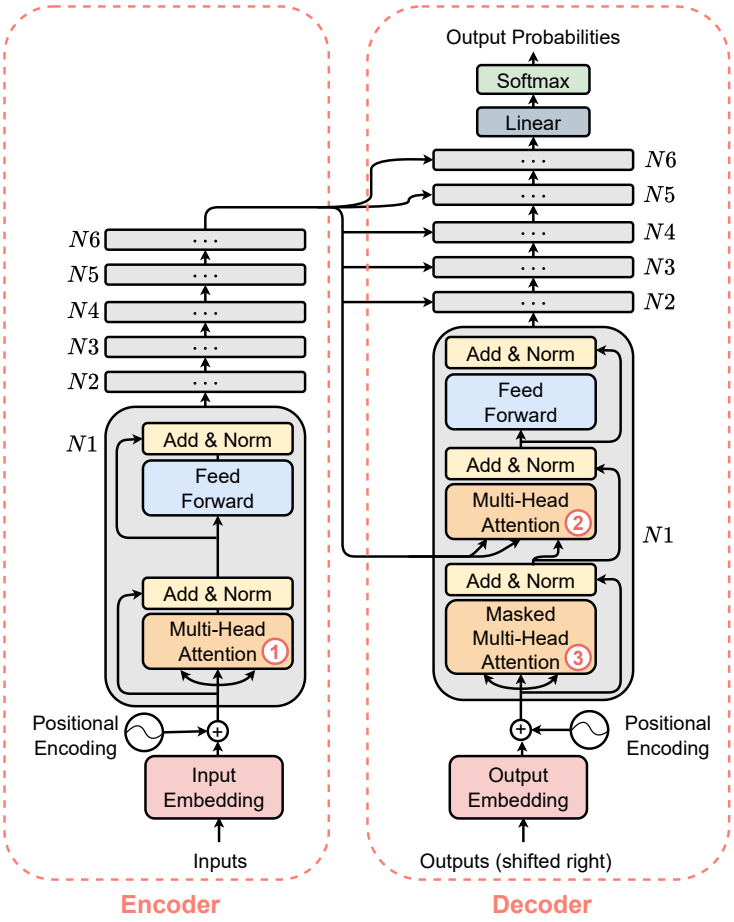


**Figure 2.6:** Transformer architecture, from Vaswani et al. (2017). Each block of encoder $Nx$ is the same as the encoder $N1$, as each decoder block $Nx$ is the same as the decoder $N1$.

In self-attention, query, keys, and values are from the same source and are the output of the previous step so that each word can attend to each other in the previous layer. Additionally, there are $K, V, Q$ matrices for scaled dot-product attention, as shown in Figure 2.4. To be able to focus more than one place in the sentence, multi-head attention mechanism is designed, which consists of multiple $K, V, Q$ matrices[6], as in Figure 2.5. This attention is used in the encoder, as (1), in Figure 2.6. In the attention of the decoder shown as (2), the keys and values are from the encoder, and the queries are from the decoder, as in the seq2seq models (Bahdanau et al., 2015). In the third type of attention, the idea is to prevent attention from looking at future when predicting the sequence (e.g. language modeling) in the decoder, and thus future words are masked out, as (3) in Figure 2.6.

The input for the first blocks contains position embeddings to catch the position information. After attention(s), point-wise FFNN is applied to each word separately and identically for non-linearity in each block. Residual connections and normalizations are applied in the blocks to train the overall model in a better and faster way. All in all, it contains 6 identical blocks of encoder and decoder.

### 2.2.4 BERT

Pre-training means to train the model over some tasks, like LM, using a large amount of data in an unsupervised way, and the trained parameters will be used as a starting point for other tasks to fine-tune the model[7] (Dai and Le, 2015).

BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) is one of the popular techniques for pre-training. BERT is based on the transformer encoder. As discussed earlier, transformer encoders use all the words in the context, i.e. both left and right contexts, however, in language modeling, there is no access to the future, i.e. right context. The key idea in BERT is to insert [MASK] tokens in place of some words and try to predict them based on other words in the sentence to pre-train the model known as a masked language modeling (MLM). Additionally, there is one more pre-training task to capture the relationship between two sentences, called next sentence prediction, where the goal is to predict whether the second sentence is actually the next one after the first sentence. BERT input can be a single sentence or a pair of sentences. There should be a special [CLS] token as the first token of a sentence and [SEP] token between two sentences to differentiate them.

### 2.2.5 Other Common Neural Networks

In this chapter, we have discussed networks that are prevailing in the NLP domain. Yet, there are several other networks, which have also been applied in this domain and shown improvements, like convolutional neural networks (CNNs).

CNN is a neural network that focuses especially on processing a grid of values, like an image (Goodfellow et al., 2016). It is composed of convolution and pooling layers.

6. https://jalammar.github.io/illustrated-transformer/
7. https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html

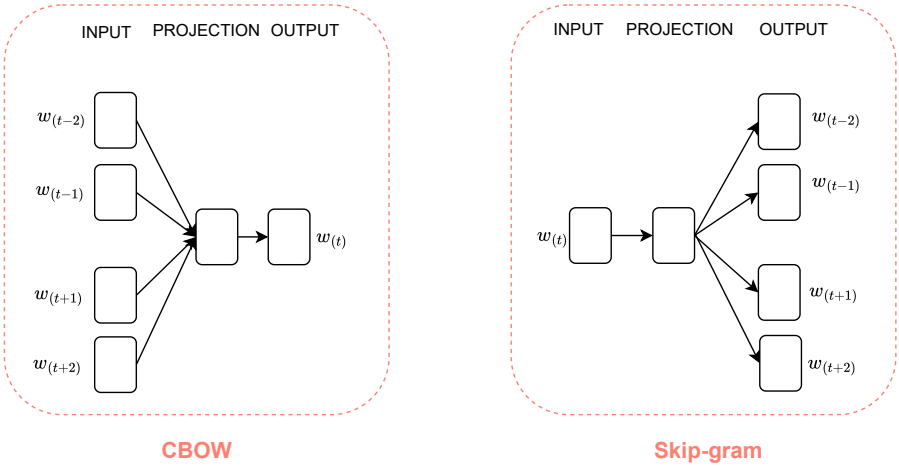


**Figure 2.7**: Word2vec architecture are from Mikolov, Chen, et al. (2013).

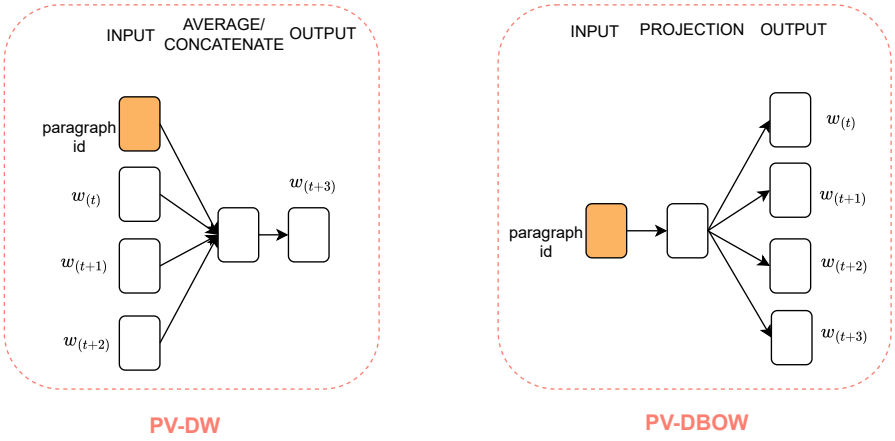


**Figure 2.8**: Doc2vec architectures are from Le and Mikolov (2014).

## 2.3 Representation Learning

Representation learning or feature learning is to learn transformation of the data to be able extract the beneficial information more easily for other downstream tasks, e.g. classification (Bengio et al., 2013). The data might be in various form, e.g. speech, image, text, graph, etc. In NLP, commonly applications utilize text. In the scope of this thesis, we summarize several representation learning techniques on text and graph data types. While creating text embeddings, the granularity of text might differ, e.g., word, sentence, document. We summarize some of them in the following sections.

### 2.3.1 Word2Vec and Doc2Vec

Word2vec (Mikolov, Chen, et al., 2013) is a neural model to learn a word representation, which has been a robust technique and has shown successful results (Goldberg, 2016). The word2vec architecture is quite similar to a feedforward network, however, the non-linear hidden layer is replaced by a projection layer to either predict the middle word from context words (known as Continuous Bag-of-Words or CBOW) or predict context words from the middle word (called Continuous Skip-gram), as shown in Figure

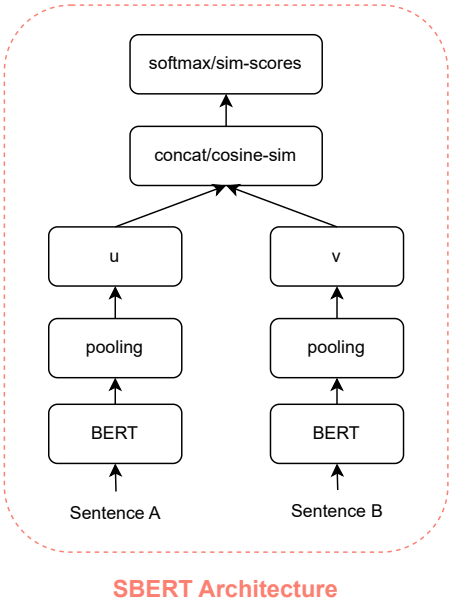


**Figure 2.9**: Siamese architecture, the figure is from Reimers and Gurevych (2019). If cosine similarity is applied to the u and v vectors, the similarity score is the output. If concatenation is applied, the output is the softmax classifier output.

2.7. The context corresponds to the fixed length sliding window of sequential words (Goldberg, 2016). Several extensions to these models have been proposed, like Mikolov, Sutskever, et al. (2013) introduce negative sampling. For example, the doc2vec (Le and Mikolov, 2014) model is designed to learn document/paragraph vectors as well as word vectors, where the paragraph token might be considered as another word. As word2vec, it has two similar variations, as shown in the Figure 2.8. At the prediction time, an inference step should be applied to generate a vector for new document/paragraph.

### 2.3.2 Sentence BERT (SBERT)

The common ways to create sentence embeddings for the individual sentences from pre-trained BERT are either directly using the output of the [CLS] token or averaging all the outputs of BERT. SBERT (Reimers and Gurevych, 2019) proposes a new method to derive sentence embeddings, which is based on fine-tuning BERT.

SBERT uses a Siamese network architecture to fine-tune BERT, which in general contains two identical networks with different inputs and joined with an objective function. The parameters are tied in the Siamese architecture. As shown in Figure 2.9, the twin BERT networks are connected with an objective function, which can optionally be either a classification objective with softmax, or a regression objective with cosine similarities, or a triplet objective that aims at reducing the distance between anchor and positive than the distance between anchor and negative (the last one is not depicted in the figure). To obtain sentence embeddings, u, v, there is a pooling operation applied to the output of BERT just after feeding sentences to BERT.

SBERT shows promising performance for some sentence embedding tasks as compared to other sentence embeddings techniques and BERT-based ones, i.e. [CLS] token

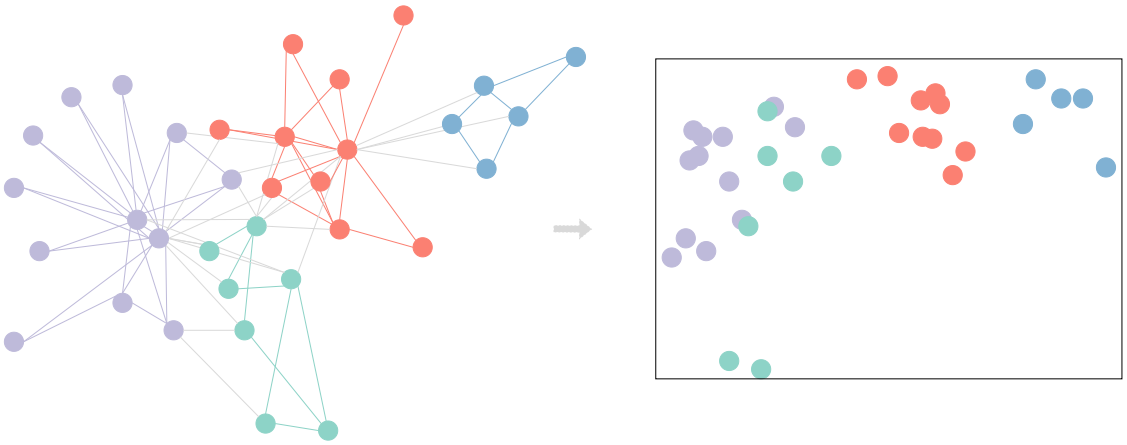

**Figure 2.10**: The input is the graph (here, network of Zachary (1977)) and the output is representations produced by DeepWalk. Figure is from Perozzi et al. (2014).

or average of outputs, as reported in their paper. So far, they provide various pre-trained models[8] based on several pre-training models rather than only BERT.

Furthermore, the authors later propose a method to extend monolingual sentence embeddings to multilingual language with the idea to map the translated sentence to the same location as the original sentence (Reimers and Gurevych, 2020).

### 2.3.3 Graph Embeddings

Recently, (knowledge) graph embedding has become a prominent technique and facilitated solving various NLP and data mining tasks (Q Wang et al., 2017) from knowledge graph completion (Nayyeri et al., 2019; Bordes et al., 2013; Z Wang et al., 2014) to entity classification (Nickel et al., 2011). Information about Knowledge Graph (KG) is summarized in Section 2.4.1. There are many models proposing a graph embedding algorithm, for instance, DeepWalk (Perozzi et al., 2014) and TransE (Bordes et al., 2013) are among well-known techniques.

The goal of the DeepWalk (Perozzi et al., 2014) algorithm is to produce embeddings of vertices that preserve their proximity in a graph (Goyal and Ferrara, 2018). The input is the graph and the output is the representations, as shown in Figure 2.10. It first generates several random walks for each vertex in a graph. The generated walks are used as training data for the skip-gram algorithm. Like in word2vec for language modeling, given a vertex, the algorithm maximizes the probabilities of its neighbors in the generated walks.

The goal of the TransE (Bordes et al., 2013) algorithm is to construct embeddings of both vertices and relations in such a way that they are compatible with the facts in a KG (Q Wang et al., 2017). Consider the facts in a KG are represented in the form of triples (i.e. head entity, relation, tail entity). If a fact is contained in a KG, the TransE margin-based ranking criterion facilitates the presence of the following correspondence between embeddings: $head + relation \approx tail$. This means that the relationship in a KG should be a linear translation in the embedding space of entities. At the same time, if there is no such fact in a KG, this functional relationship should not hold. The TransE-based entity representations constructed from Wikidata (Vrandečić and Krötzsch, 2014) and Freebase (Bollacker et al., 2008) have been used for entity representation in language

8. https://www.sbert.net/docs/pretrained_models.html

modeling (Z Zhang et al., 2019) and in several works on entity linking (Banerjee et al., 2020; Sorokin and Gurevych, 2018; Nedelchev et al., 2020).

There are many other techniques for graph embedding: (Grover and Leskovec, 2016; Z Wang et al., 2014; Nickel et al., 2011; Trouillon et al., 2016; B Yang et al., 2015; Dettmers et al., 2018) inter alia and very recent 5*E (Nayyeri et al., 2021), which is designed to preserve complex graph structures in the embedding space. A detailed overview of all graph embedding algorithms is out of the scope of the current work. We refer the reader to the surveys on this topic (Goyal and Ferrara, 2018; Cai et al., 2018; Q Wang et al., 2017; Ruffinelli et al., 2020).

## 2.4 Knowledge Resources

In the scope this thesis, we take advantage from two sources of knowledge: DBpedia and distributional thesaurus. Both are in the form of graph structure, and we leverage DBpedia throughout graph embeddings, while the distributional thesaurus is utilized via the JoBimText API in a different scenario.

### 2.4.1 Knowledge Graphs and DBpedia

**Knowledge Graphs** Knowledge Graphs (KGs), such as Freebase (Bollacker et al., 2008), DBpedia (Lehmann et al., 2015), and Wikidata (Vrandečić and Krötzsch, 2014), contain rich and precise information about entities of all kinds, such as persons, locations, organizations, movies, and scientific theories, just to name a few. Each entity has a set of carefully defined relations and attributes, e.g. “was born in” or “play for”.

A KG contains entities, relations, and facts, where facts are denoted as triples (i.e. head entity, relation, tail entity) as defined by S Ji et al. (2022). Formally, as defined by Färber et al. (2018), a KG is a set of Resource Description Framework (RDF) triples where each triple $(s, p, o)$ is an ordered set of the following terms: a subject $s$, a predicate $p$, and an object $o$.

Modern widely-used knowledge bases organize information in the form of a graph (Lehmann et al., 2015; Bollacker et al., 2008; Vrandečić and Krötzsch, 2014). Hence, in this thesis, the terms Knowledge Graph (KG) and Knowledge Base (KB) are used interchangeably.

**DBpedia** Wikipedia articles contain unstructured free text, as well as structured data, e.g. infoboxes, tables, categorization information, etc. Yet, one might have difficulties to find the answer for some queries e.g. “all basketball players that played for NBA.”, since the search provided by Wikipedia is limited (Lehmann et al., 2015). DBpedia extracts the structured data and turns it to knowledge graph that can be utilized for such difficult queries (Lehmann et al., 2015). The overview of DBpedia extraction framework is shown in Figure 2.11, as drawn by Lehmann et al. (2015). There are various types of extractors in DBpedia. A few examples are as following items.

- **abstract**: extracting the first lines of the Wikipedia article.
- **disambiguation**: extracting disambiguation links.
- **interlanguage**: extracting interwiki links.

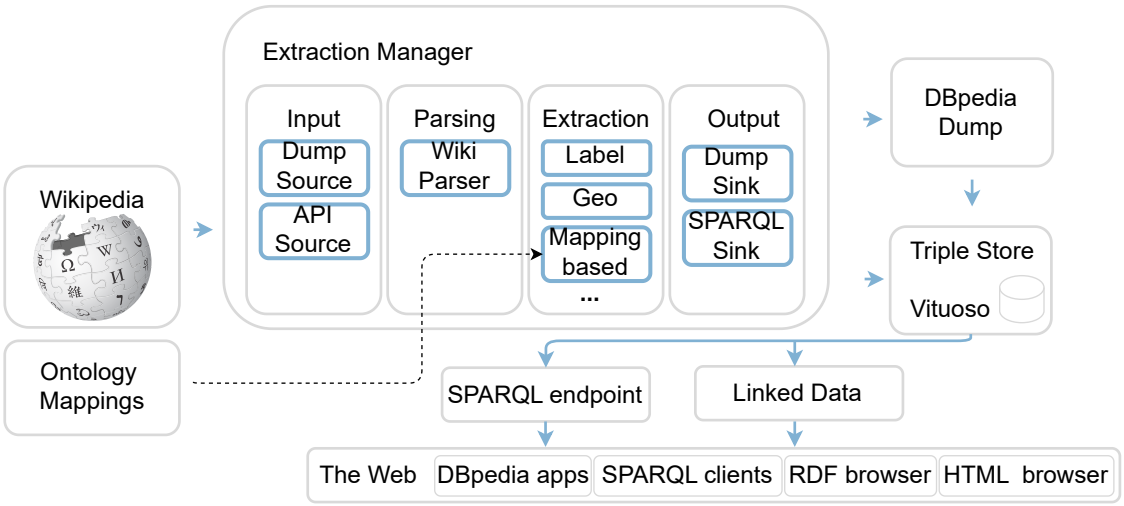


**Figure 2.11**: Overview of extraction framework in DBpedia, figure is from Lehmann et al. (2015).

- **label**: extracting the article title as label.
- **page ID**: extracting page ids of articles.
- **page links**: extracting all links between Wikipedia articles.
- **redirects**: extracting redirect links between articles in Wikipedia.

These examples are by Lehmann et al. (2015), for more extractors and more information, see (Lehmann et al., 2015).

### 2.4.2 Distributional Thesaurus - JoBimText

There are various types of graphs, consisting of different node types. For instance, a distributional thesaurus (DT) is a graph of terms. One such graph construction is proposed by the JoBimText framework (Biemann and Riedl, 2013). The underlying technology of this framework involves a holing operation as the first step, which performs the split of a term (Jo) and a contextual feature (Bim) based on structural observations of text (e.g., dependency parsing). Next, by pruning these terms and features (based, e.g., on some significance scores, like LMI (Kilgarriff et al., 2004)) and by aggregating terms based on their overlapping contextual features, a distributional thesaurus, a graph of terms, is constructed. An example workflow is shown in Figure 2.12 as drawn by Biemann and Riedl (2013). Furthermore, they cluster an ego/neighboring graph (i.e., a sub-graph containing similar terms to a particular term) of a DT entry (i.e., term) using the Chinese Whispers algorithm (Biemann, 2006) to get sense information of an entry in terms of its similar entries. Each induced sense is labeled based on the information of IS-A relationship between terms with their frequencies, collected by applying IS-A (hypernym) patterns (Hearst, 1992) on a text collection. The API (Ruppert et al., 2015) allows to access the information of the JoBimText framework (for more information, see (Biemann et al., 2013; Ruppert et al., 2015; Biemann and Riedl, 2013; Riedl and Biemann, 2013)).

So far, in this chapter, we have summarized the underlying technologies that are discussed in the following chapters, e.g. feedforward neural networks, knowledge graphs,

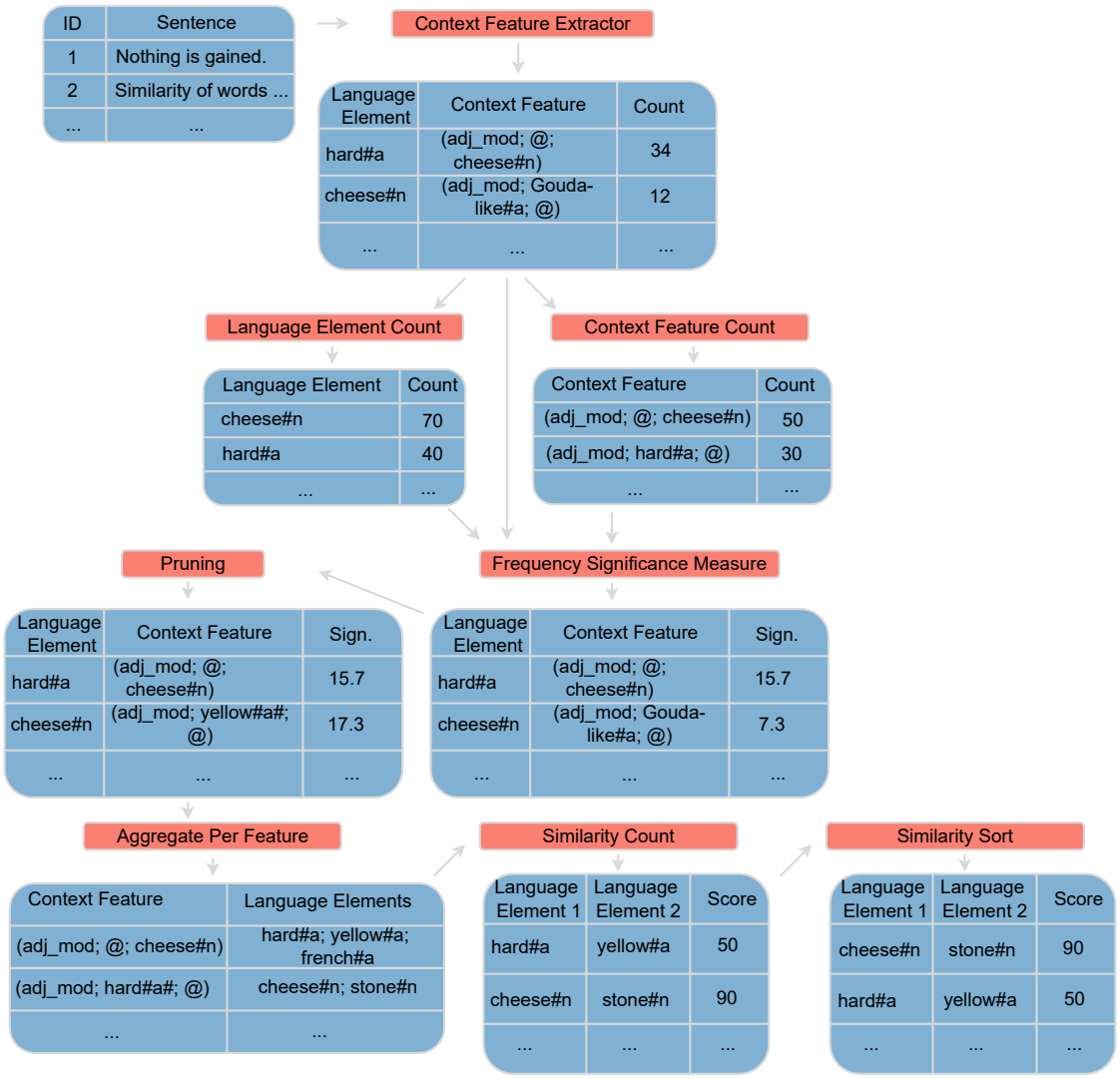


**Figure 2.12:** An example workflow of data processing in JoBimText. Figure is from Biemann and Riedl (2013).

etc. For instance, in the next chapter, we will refer knowledge graphs while describing entity linking and/or several architectures to present different choices of recent neural entity linking and disambiguation models. As already mentioned, we will discuss recent neural entity linking and disambiguation models, systematically, in the next chapter.

# 3

# Survey of Entity Linking Methods

**In this chapter, we present a work to comprehensively describe deep learning based neural entity linking and disambiguation systems developed since 2015 up until and including 2021, with the goal to systemize design features of neural EL. A generic architecture is distilled and its essential components are discussed, prominent methods for each of them are summarized. We group the vast variety of modifications of this general architecture by several common themes. The content of this chapter was published in (Sevgili et al., 2022), edited to fit in the thesis, e.g. some contents are moved to some other chapter, or excluded, corrected language issues, etc.**

## Contents

## 3.1 Introduction

There exist unstructured and structured information available through the web in various formats, for instance, KGs provide information about entities, e.g. a person, location, organization, etc., along with their relations e.g. "was born in", as discussed in Section 2.4.1, in Chapter 2. This wealth of structured information gives rise to and facilitates the development of semantic processing algorithms as they can directly operate on and benefit from such entity representations. For instance, imagine a search engine that is able to retrieve mentions in the news during the last month of all retired NBA players with a net income of more than 1 billion US dollars. The list of players together with their income and retirement information may be available in a knowledge graph. Equipped with this information, it appears to be straightforward to look up mentions of retired basketball players in the newswire. However, the main obstacle in this setup is the lexical ambiguity of entities. In the context of this application, one would want to only retrieve all mentions of "Michael Jordan (basketball player)"[1] and exclude mentions of other persons with the same name such as "Michael Jordan (mathematician)"[2].

This is why Entity Linking (EL) – the process of matching a mention, e.g. "Michael Jordan", in a textual context to a KG record (e.g. "basketball player" or "mathematician") fitting the context – is the key technology enabling various semantic applications. Thus, EL is the task of identifying an entity mention in the (unstructured) text and establishing a link to an entry in a (structured) knowledge graph.

Entity linking is an essential component of many information extraction (IE) and natural language understanding (NLU) pipelines since it resolves the lexical ambiguity of entity mentions and determines their meanings in context. A link between a textual mention and an entity in a knowledge graph also allows us to take advantage of the information encompassed in a semantic graph, which is shown to be useful in such NLU tasks as information extraction, biomedical text processing, or semantic parsing and question answering (see Section 5, in the article (**Sevgili et al.**, 2022)). This wide range of direct applications is the reason why entity linking is enjoying great interest from both academy and industry for more than two decades.

### 3.1.1 Goal and Scope

Recently, a new generation of approaches for entity linking based on neural models and deep learning emerged, pushing the state-of-the-art performance in this task to a new level. The goal of this chapter is to provide an overview of this wave of models, emerging from 2015 up until and including 2021.

Models based on neural networks have managed to excel in EL as in many other natural language processing tasks due to their ability to learn useful distributed semantic representations of linguistic data (Collobert et al., 2011; Young et al., 2018; Bengio et al., 2003). These current state-of-the-art neural entity linking models have shown significant improvements over "classical"[3] machine learning approaches (Lazic et al., 2015; Ratinov

1. https://en.wikipedia.org/wiki/Michael_Jordan
2. https://en.wikipedia.org/wiki/Michael_I._Jordan
3. On classical ML vs deep learning: https://towardsdatascience.com/deep-learning-vs-classical-machine-learning-9a42c6d48aa

et al., 2011; Chisholm and Hachey, 2015) to name a few that are based on shallow architectures, e.g. Support Vector Machines, and/or depend mostly on hand-crafted features. Such models often cannot capture all relevant statistical dependencies and interactions (Ganea and Hofmann, 2017). In contrast, deep neural networks are able to learn sophisticated representations within their deep layered architectures. This reduces the burden of manual feature engineering and enables significant improvements in EL and other tasks.

In this study, recently proposed neural models are systemized, distilling one generic architecture used by the majority of neural EL models (illustrated in Figures, e.g., 3.2). We describe the models used in each component of this architecture, e.g. candidate generation, mention-context encoding, entity ranking. Prominent variations of this generic architecture, e.g. end-to-end EL or global models, are also discussed. To better structure the sheer amount of available models, various types of methods are illustrated in taxonomies (Figures 3.3 and 3.5), while notable features of each model are carefully assembled in a tabular form (Table 3.2). We discuss the performance of the models on commonly used entity linking/disambiguation benchmarks and an entity relatedness dataset. Because of the sheer amount of work, it was not possible for us to try available software and to compare approaches on further parameters, such as computational complexity, run-time, and memory requirements. Nevertheless, we created a comprehensive collection of references to publicly available official implementations of EL models and systems discussed in this chapter (see Table 3.3).

### 3.1.2 Article Collection Methodology

In this work, there is no strict article collection algorithm for the review like e.g., the one conducted by Oliveira et al. (2021). Our main goal is to provide and describe a conceptual framework that can be applied to the majority of recently presented neural approaches to EL. Nevertheless, as with all surveys, we had to draw the line somewhere. The main criteria for including papers into this chapter was that they had been published during or after 2015 up until and including 2021, and they primarily address the task of EL/ED, i.e. resolving textual mentions to entries in KGs. We explicitly exclude related work e.g., on (fine-grained) entity typing (see (Aly et al., 2021; Choi et al., 2018)), which also encompasses a disambiguation task, and work that employs KGs for other tasks than EL. This study also does not try to cover all EL methods designed for specific domains like biomedical texts or messages in social media. For the general-purpose EL models evaluated on well-established benchmarks, we try to be as comprehensive as possible with respect to recent-enough papers that fit into the conceptual framework, no matter where they have appeared (however, with a focus on top conferences and journals in the fields of natural language processing and Semantic Web).

### 3.1.3 Previous Surveys

One of the first surveys on EL was prepared by Shen et al. (2015) in 2015. They cover the main approaches to entity linking (within the modules, e.g. candidate generation, ranking), its applications, evaluation methods, and future directions. In the same year, Ling et al. (2015) presented a work that aims to provide (1) a standard problem definition to reduce confusion that appears due to the existence of variant similar tasks related to

EL (e.g., Wikification (Milne and Witten, 2008) and named entity linking (Hoffart et al., 2011)), and (2) a clear comparison of models and their various aspects.

There are also other surveys that address a wider scope. The work of Martínez-Rodríguez et al. (2020), published in 2020, involves information extraction models and semantic web technologies. Namely, they consider many tasks, like named entity recognition, entity linking, terminology extraction, keyphrase extraction, topic modeling, topic labeling, relation extraction tasks. In a similar vein, the work of Al-Moslmi et al. (2020), released in 2020, overviews the research in named entity recognition, named entity disambiguation, and entity linking published between 2014 and 2019.

Another recent survey paper by Oliveira et al. (2021), published in 2020, analyzes and summarizes EL approaches that exhibit some holism. This viewpoint limits the survey to the works that exploit various peculiarities of the EL task: additional metadata stored in specific input like microblogs, specific features that can be extracted from this input like geographic coordinates in tweets, timestamps, interests of users posted these tweets, and specific disambiguation methods that take advantage of these additional features. In the concurrent work, Möller et al. (2022) overview models developed specifically for linking English entities to the Wikidata (Vrandečić and Krötzsch, 2014) and discuss features of this KG that can be exploited for increasing the linking performance.

Previous surveys on similar topics (a) do not cover many recent publications (Ling et al., 2015; Shen et al., 2015), (b) broadly cover numerous topics (Martínez-Rodríguez et al., 2020; Al-Moslmi et al., 2020), or (c) are focused on the specific types of methods (Oliveira et al., 2021) or a knowledge graph (Möller et al., 2022). There is not yet, to our knowledge, a detailed survey specifically devoted to recent neural entity linking models. The previous surveys also do not address the topics of entity and context/mention encoding.

The structure of this chapter is the following. It is started with defining the EL task in Section 3.2. In Section 3.3.1, the general architecture of neural entity linking systems is presented. Modifications and variations of this basic pipeline are discussed in Section 3.3.2. In Section 3.4, we summarize the performance of EL models on standard benchmarks and present results of the entity relatedness evaluation. Finally, Section 3.5 concludes the chapter.

## 3.2 Task Description

### 3.2.1 Informal Definition

Consider the example presented in Figure 3.1 with an entity mention *Scott Young* in a soccer-game-related context. Literally, this common name can refer to at least three different people: the *American football player*, the *Welsh football player*, or the *writer*. The EL task is to (1) correctly detect the entity mention in the text, (2) resolve its ambiguity and ultimately provide a link to a corresponding entity entry in a KG, e.g. provide for the *Scott Young* mention in this context a link to the *Welsh footballer*[4] instead of the *writer*[5]. To achieve this goal, the task is usually decomposed into two sub-tasks, as illustrated in Figure 3.1: Mention Detection (MD) and Entity Disambiguation (ED).

4. https://en.wikipedia.org/wiki/Scott_Young_(Welsh_footballer)
5. https://en.wikipedia.org/wiki/Scott_Young_(writer)

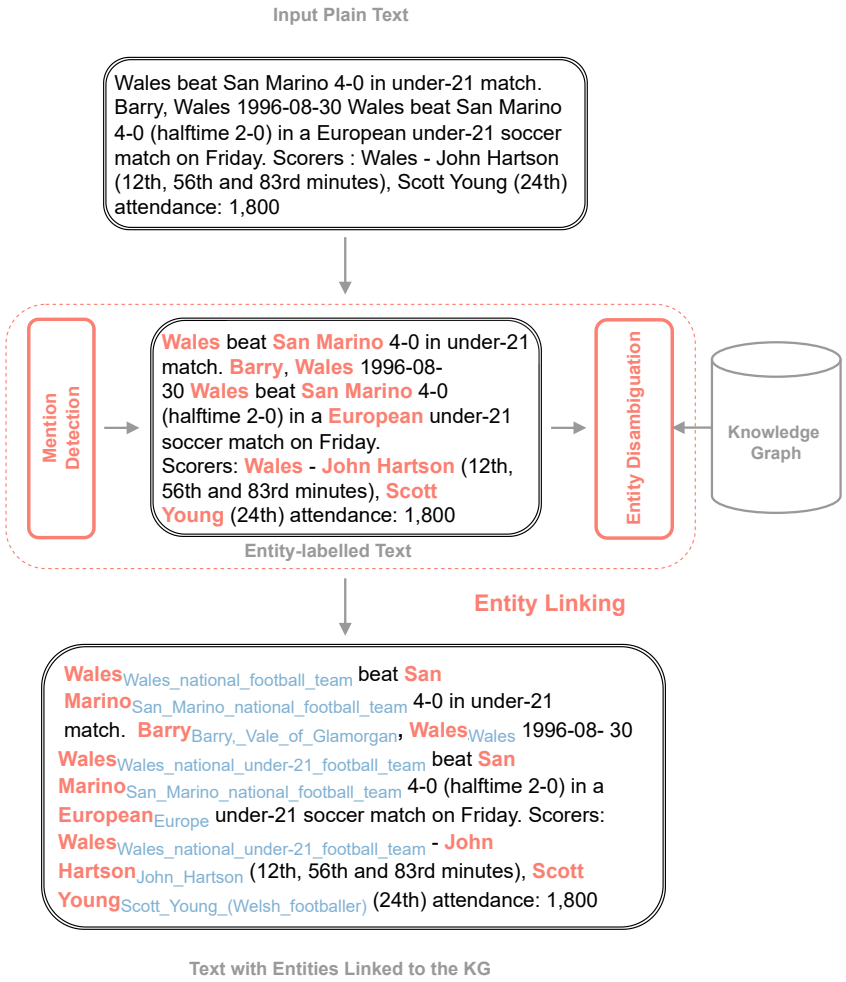


**Figure 3.1: The entity linking task.** An Entity Linking (EL) model takes a raw textual input and enriches it with entity mentions linked to nodes in a Knowledge Graph (KG). The task is commonly split into entity mention detection and entity disambiguation sub-tasks.

## 3.2.2 Formal Definition

### 3.2.2.1 Knowledge Graph (KG)

A KG contains entities, relations, and facts, where facts are denoted as triples (i.e. head entity, relation, tail entity) as defined by S Ji et al. (2022). As discussed, in Section 2.4.1, a KG is a set of Resource Description Framework (RDF) triples (Färber et al., 2018).

This RDF representation can be considered as a multi-relational graph $G = (E, \mathbb{A} = \{A_0, A_1, ..., A_m \subseteq (E \times E)\})$, where $E$ is a set of all entities of a KG, and $\mathbb{A}$ is a family of typed edge sets of length $m$. For example, $A_0$ is the "occupation" predicate adjacency matrix, $A_1$ is the "founded" predicate adjacency matrix, etc.

There is also an equivalent three-way tensor representation of a KG $\mathcal{A} \in \{0, 1\}^{n \times m \times n}$, where

$$\mathcal{A}_{i,k,j} = \begin{cases} 1 & \text{if } (i, j) \in A_k \ : \ k \leq m \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

#### 3.2.2.2 Mention Detection (MD)

The goal of mention detection is to identify an entity mention span, while entity disambiguation performs linking of found mentions to entries of a KG. We can consider this task as determining an MD function that takes as input a textual context $c_i \in C$ (e.g. a document in a document collection) and outputs a sequence of $n$ mentions $(m_1, \dots m_n)$ in this context $m_i \in M$, where $M$ is a set of all possible text spans in the context:

$$\mathsf{MD} : C \rightarrow M^n. \tag{3.2}$$

In the majority of works on EL, it is assumed that the mentions are already given or detected, for example, using a named entity recognition (NER) system (sometimes called named entity recognition and classification (NERC) (Aly et al., 2021; Nadeau and Sekine, 2007)). We should note that, usually, in addition to MD, NER systems also tag/classify mentions with a predefined types (J Li et al., 2022; van Hulst et al., 2020; Oliveira et al., 2021; Martins et al., 2019) that also can be leveraged for disambiguation (Martins et al., 2019).

#### 3.2.2.3 Entity Disambiguation (ED)

The entity disambiguation task can be considered as determining a function ED that, given a sequence of $n$ mentions in a document and their contexts $(c_1, \dots, c_n)$, outputs an entity assignment $(e_1, \dots, e_n), e_i \in E$, where $E$ is a set of entities in a KG:

$$\mathsf{ED} : (M, C)^n \rightarrow E^n. \tag{3.3}$$

To learn a mapping from entity mentions in a context to entity entries in a KG, EL models use supervision signals like manually annotated mention-entity pairs. The size of KGs varies; they can contain hundreds of thousands or even millions of entities. Due to their large size, training data for EL would be extremely unbalanced; training sets can lack even a single example for a particular entity or mention, e.g. as in the popular AIDA corpus (Hoffart et al., 2011). To deal with this problem, EL models should have wide generalization capabilities.

Despite KGs being usually large, they are incomplete. Therefore, some mentions in a text cannot be correctly mapped to any KG entry. Determining such unlinkable mentions, which usually is designated as linking to a NIL entry, is one of the current EL challenges. Methods that address this problem provide a separate function for it or extend the set of entities in the disambiguation function with this special entry:

$$\mathsf{ED} : (M, C)^n \rightarrow (E \cup \mathsf{NIL})^n. \tag{3.4}$$

### 3.2.3 Terminological Aspects

More or less, the same technologies and models are sometimes called differently in the literature. Namely, Wikification (Cheng and Roth, 2013) and entity disambiguation are considered as subtypes of EL (Moro et al., 2014). To be comprehensive, we assume that the entity linking task encompasses both entity mention detection and entity disambiguation. However, only a few studies suggest models that perform MD and ED jointly, while the majority of papers on EL focus exclusively on ED and assume that mention boundaries are given by an external entity recognizer (Rizzo et al., 2014) (which

may lead to some terminological confusions). Numerous techniques that perform MD (e.g. in the NER task) without entity disambiguation are considered in many previous surveys (Nadeau and Sekine, 2007; Sharnagat, 2014; Goyal et al., 2018; Yadav and Bethard, 2018; J Li et al., 2022) inter alia and are out of the scope of this work.

Entity linking in the general case is not restricted to linking mentions to graph nodes but rather to concepts in a knowledge base. However, most of the modern widely-used knowledge bases organize information in the form of a graph (Lehmann et al., 2015; Bollacker et al., 2008; Vrandečić and Krötzsch, 2014), even in particular domains, like e.g. the scholarly domain (Dessì et al., 2021). A basic statement in a data/knowledge base usually can be represented as a subject-predicate-object tuple $(s, p, o)$, e.g. (John_Lennon, occupation, singer) or (New_York_City, founded, 1624), and a set of such tuples can be represented as a multi-relational graph. This formalism helps to efficiently organize knowledge for many applications ranging from search engines to question answering and recommendation systems (Hogan et al., 2021; S Ji et al., 2022). Therefore, as discussed earlier, in this thesis, the terms Knowledge Graph (KG) and Knowledge Base (KB) are used interchangeably.

## 3.3 Neural Entity Linking

The discussion of neural entity linking approaches will start with the most general architecture of EL pipelines and continue with various specific modifications like joint entity mention detection and linking, disambiguation techniques that leverage global context, domain-independent EL approaches including zero-shot methods, and cross-lingual models.

### 3.3.1 General Architecture

Some of the attempts to EL based on neural networks treat it as a multi-class classification task in which entities correspond to classes. However, the straightforward approach results in a large number of classes, which leads to suboptimal performance without task-sharing (Kar et al., 2018). The streamlined approach to EL is to treat it as a ranking problem. The generalized EL architecture is presented in Figure 3.2, which is applicable to the majority of neural approaches. Here, the mention detection model identifies the mention boundaries in text. The next step is to produce a shortlist of possible entities (candidates) for the mention, e.g. producing Scott_Young_(writer) as a candidate rather than a completely random entity. Then, the mention encoder produces a semantic vector representation of a mention in a context. The entity encoder produces a set of vector representations of candidates. Finally, the entity ranking model compares mention and entity representations and estimates mention-entity correspondence scores. An optional step is to determine unlinkable mentions, for which a KG does not contain a corresponding entity. The categorization of each step in the general neural EL architecture is summarized in Figure 3.3.

#### 3.3.1.1 Candidate Generation

An essential part of EL is candidate generation. The goal of this step is given an ambiguous entity mention, such as “Scott Young”, to provide a list of its possible “senses”

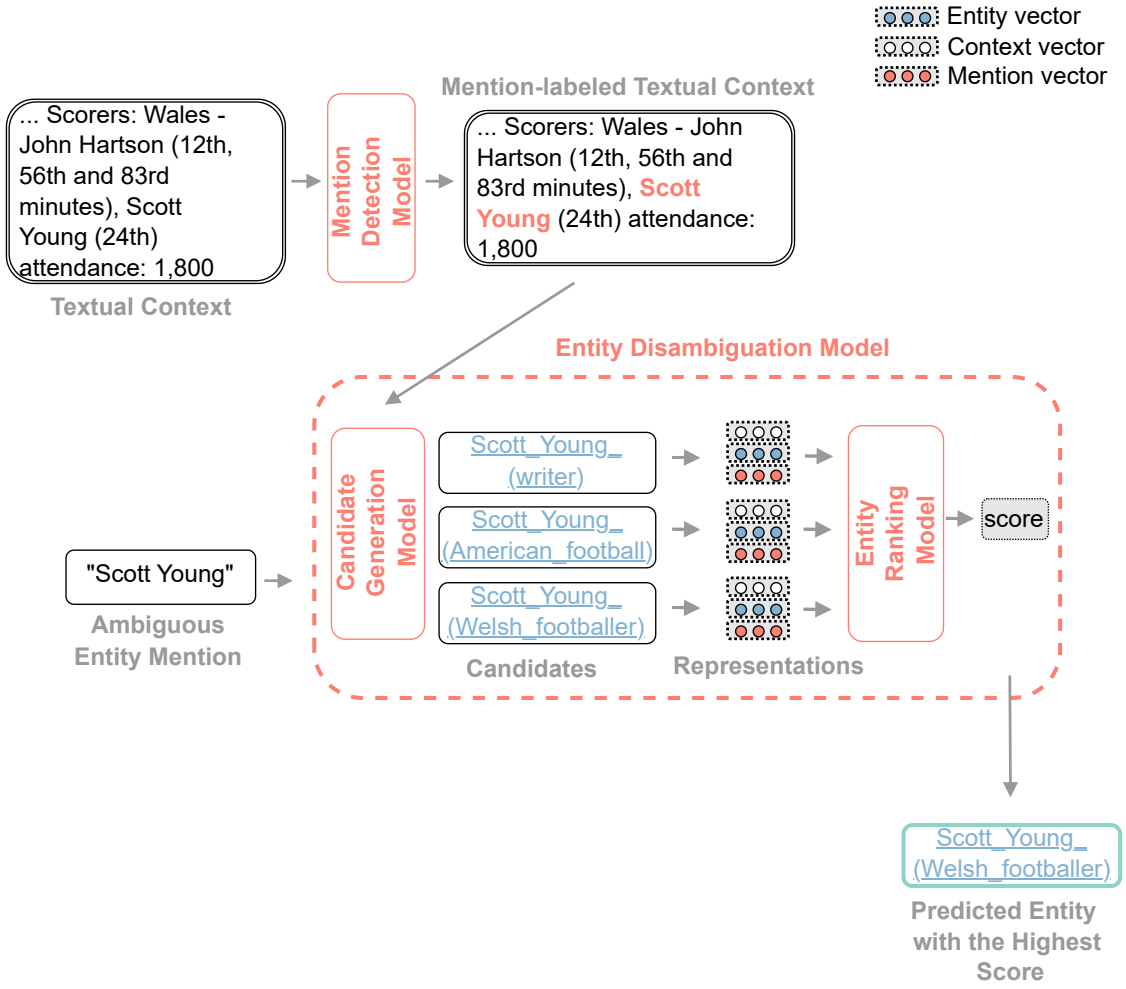


**Figure 3.2: General architecture for neural entity linking.** Entity Linking (EL) consists of two main steps: *Mention Detection (MD)*, when entity mention boundaries in a text are identified, and *Entity Disambiguation (ED)*, when a corresponding entity is predicted for the given mention. Entity disambiguation is further carried out in two steps: *Candidate Generation*, when possible candidate entities are selected for the mention, and *Entity Ranking*, when a correspondence score between context/mention and each candidate is computed through the comparison of their vector representations.

as specified by entities in a KG. EL is analogous to the Word Sense Disambiguation (WSD) task (Moro et al., 2014; Navigli, 2009) as it also resolves lexical ambiguity. Yet in WSD, each sense of a word can be clearly defined by WordNet (*WordNet: An Electronic Lexical Database*, 1998), while in EL, KGs do not provide such an exact mapping between mentions and entities (Moro et al., 2014; Navigli, 2009; Chang et al., 2016). Therefore, a mention potentially can be linked to any entity in a KG, resulting in a large search space, e.g. "Big Blue" referring to IBM. In the candidate generation step, this issue is addressed by performing effective preliminary filtering of the entity list.

Formally, given a mention $m_i$, a candidate generator provides a list of probable entities, $e_1, e_2, ..., e_k$, for each entity mention in a document.

$$\mathsf{CG} : M \rightarrow (e_1, e_2, ..., e_k). \tag{3.5}$$

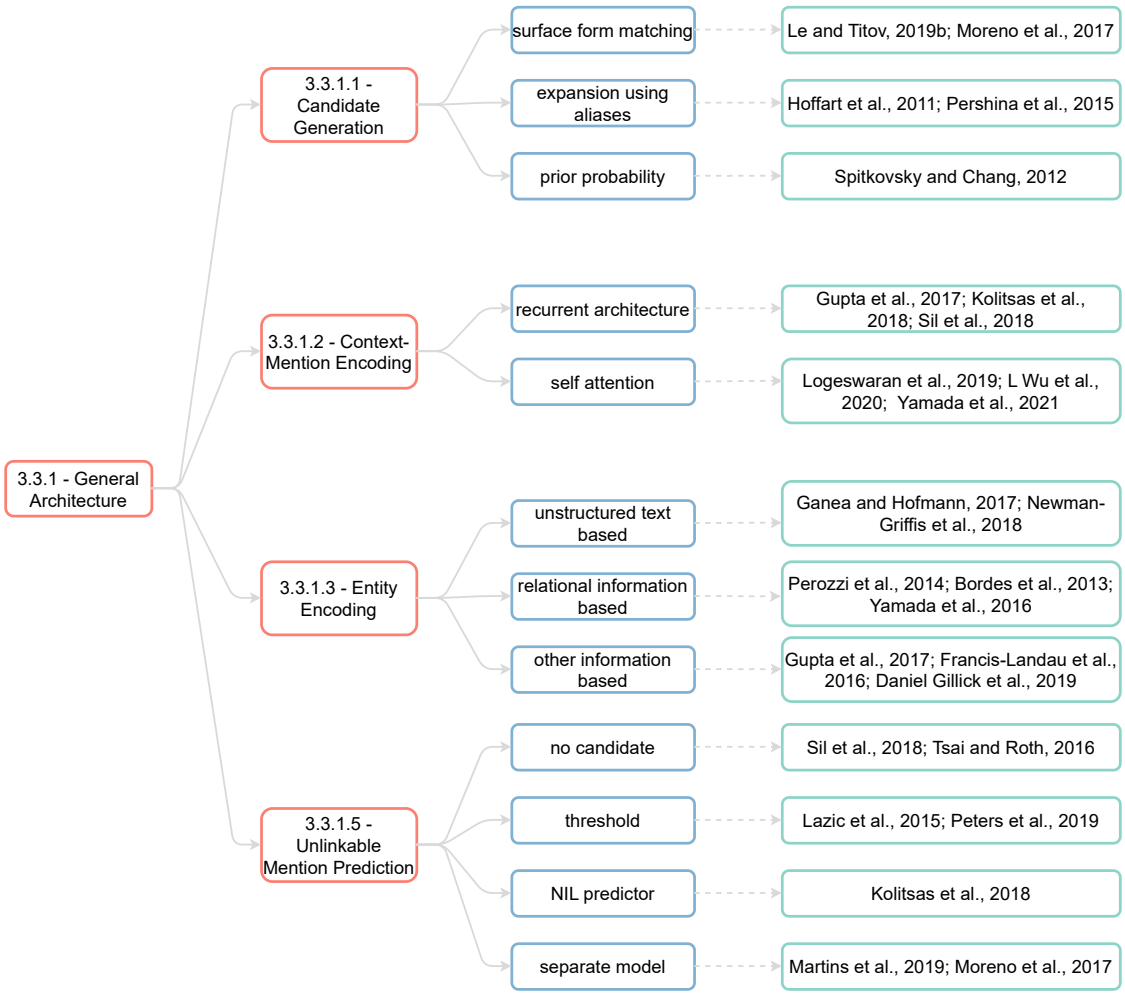


**Figure 3.3: Reference map of the general architecture of neural EL systems.** The categorization of each step in the general neural EL architecture with alternative design choices and a few example references illustrating each of the choices.

Similar to Shen et al. (2015) and Al-Moslmi et al. (2020), we distinguish three common candidate generation methods in neural EL: (1) based on surface form matching, (2) based on expansion with aliases, and (3) based on a prior matching probability computation. In the first approach, a candidate list is composed of entities that match various surface forms of mentions in the text (Zwicklbauer et al., 2016b; Moreno et al., 2017; Le and Titov, 2019b). There are many heuristics for the generation of mention forms and matching criteria like the Levenshtein distance, n-grams, and normalization. For the example mention of "Big Blue", this approach would not work well, as the referent entity "IBM" or its long-form "International Business Machines" does not contain a mention string. Examples of candidate entity sets are presented in Table 3.1, where we searched a name matching of the mention "Big Blue" in the titles of all Wikipedia articles present in DBpedia and presented random 5 matches.

In the second approach, a dictionary of additional aliases is constructed using KG metadata like disambiguation/redirect pages of Wikipedia (Z Fang et al., 2019; Zwicklbauer et al., 2016b) or using a dictionary of aliases and/or synonyms (e.g. "NYC" stands for "New York City"). This helps to improve the candidate generation recall as

**Table 3.1: Candidate generation examples.** Candidate entities for the example mention "Big Blue" obtained using several candidate generation methods. The highlighted candidates are "correct" entities assuming that the given mention refers to the IBM corporation and not a river, e.g. Big_Blue_River_(Kansas).

| Method | 5 candidate entities for the example mention "Big Blue" |
|---|---|
| **surface form matching based**[6] on DBpedia names | Big_Blue_Trail, Big_Bluegrass, Big_Blue_Spring_cave_crayfish, Dexter_Bexley_and_the_Big_Blue_Beastie, IBM_Big_Blue_(X-League) |
| **expansion using aliases**[7] from YAGO-means | Big_Blue_River_(Indiana), Big_Blue_River_(Kansas), Big_Blue_(crane), Big_Red_(drink), **IBM** |
| **probability + expansion using aliases**[8] from (Ganea and Hofmann, 2017): Anchor prob. + CrossWikis + YAGO | **IBM**, Big_Blue_River_(Kansas), The_Big_Blue Big_Blue_River_(Indiana), Big_Blue_(crane) |

the surface form matching usually cannot catch such cases. Pershina et al. (2015) expand the given mention to the longest mention in a context found using coreference resolution. Then, an entity is selected as a candidate if its title matches the longest version of the mention, or it is present in disambiguation/redirect pages of this mention. This resource is used in many EL models, e.g. Yamada et al. (2016), Cao et al. (2017), Newman-Griffis et al. (2018), Radhakrishnan et al. (2018), Martins et al. (2019), Onoe and Durrett (2020), and Sil et al. (2018). Another well-known alternative is YAGO (Suchanek et al., 2007) – an ontology automatically constructed from Wikipedia and WordNet. Among many other relations, it provides *"means"* relations, and this mapping is utilized for candidate generation like in Hoffart et al. (2011), Yamada et al. (2016), Ganea and Hofmann (2017), Sil et al. (2018), and Shahbazi et al. (2018). In this technique, the external information would help to disambiguate "Big Blue" as "IBM". Table 3.1 shows examples of candidates generated with the help of the YAGO-means candidate mapping dataset used in Hoffart et al. (2011).

The third approach to candidate generation is based on pre-calculated prior probabilities of correspondence between certain mentions and entities, $p(e|m)$. Many studies rely on mention-entity priors computed based on Wikipedia entity hyperlinks. A URL of a hyperlink to an entity page of Wikipedia determines a candidate entity, and the anchor text of the hyperlink determines a mention. Another widely-used option is CrossWikis (Spitkovsky and Chang, 2012), which is an extensive resource that leverages the frequency of mention-entity links in web crawl data (Ganea and Hofmann, 2017; Gupta et al., 2017).

6. Random matches from DBpedia labels dataset – http://downloads.dbpedia.org/2016-10/core-i18n/en/labels_en.ttl.bz2

7. YAGO-means dataset of Hoffart et al. (2011) – http://resources.mpi-inf.mpg.de/yago-naga/aida/download/aida_means.tsv.bz2

8. We generated these examples using the source code of Peters et al. (2019) – https://github.com/allenai/kb

It is common to apply multiple approaches to candidate generation at once. For example, the resource constructed by Ganea and Hofmann (2017) and used in many other EL methods (Kolitsas et al., 2018; Peters et al., 2019; Yamada et al., 2021; Shahbazi et al., 2019; Le and Titov, 2019a) relies on prior probabilities obtained from entity hyperlink count statistics of CrossWikis (Spitkovsky and Chang, 2012) and Wikipedia, as well as on entity aliases obtained from the "means" relationship of the YAGO ontology (Hoffart et al., 2011). The illustrative mention "Big Blue" can be linked to its referent entity "IBM" with this method, as shown in Table 3.1. As another example, Z Fang et al. (2020) utilize surface form matching and aliases. They share candidates between abbreviations and their expanded versions in the local context. The aliases are obtained from Wikipedia redirect and disambiguation pages, the Wikipedia search engine, and synonyms from WordNet (*WordNet: An Electronic Lexical Database*, 1998). Additionally, they submit mentions that are misspelled or contain multiple words to Wikipedia and Google search engines and search for the corresponding Wikipedia articles. It is also worth noting that some works also employ a candidate pruning step to reduce the number of candidates.

Recent zero-shot models (Logeswaran et al., 2019; Daniel Gillick et al., 2019; L Wu et al., 2020) perform candidate generation without external resources.

#### 3.3.1.2 Context-mention Encoding

To correctly disambiguate an entity mention, it is crucial to thoroughly capture the information from its context. The current mainstream approach is to construct a dense contextualized vector representation of a mention $\boldsymbol{y}_m$ using an encoder neural network.

$$\mathsf{mENC} : (C, M)^n \rightarrow (\boldsymbol{y}_{m_1}, \boldsymbol{y}_{m_2}, ..., \boldsymbol{y}_{m_n}). \tag{3.6}$$

There are several ways to create such vector representations, and recently mostly based on recurrent networks (e.g., (Gupta et al., 2017; Kolitsas et al., 2018; Sil et al., 2018)) and self-attention (e.g., (Logeswaran et al., 2019; L Wu et al., 2020; Yamada et al., 2021)). For more information, please see the article (**Sevgili et al.**, 2022).

#### 3.3.1.3 Entity Encoding

To make EL systems robust, it is essential to construct distributed vector representations of entity candidates $\boldsymbol{y}_e$ in such a way that they capture semantic relatedness between entities in various aspects.

$$\mathsf{eENC} : E^k \rightarrow (\boldsymbol{y}_{e_1}, \boldsymbol{y}_{e_2}, ..., \boldsymbol{y}_{e_k}). \tag{3.7}$$

For instance, in Figure 3.4, the most similar entities for *Scott Young* in the Scott_Young_(American_football) sense are related to American football, whereas the Scott_Young_(writer) sense is in the proximity of writer-related entities.

There are three common approaches to entity encoding in EL: (1) entity representations learned using unstructured texts and algorithms like word2vec (Mikolov, Sutskever, et al., 2013) based on co-occurrence statistics and developed originally for embedding words; (2) entity representations constructed using relations between entities in KGs and various graph embedding methods; (3) training a full-fledged neural encoder to convert textual descriptions of entities and/or other information into embeddings.

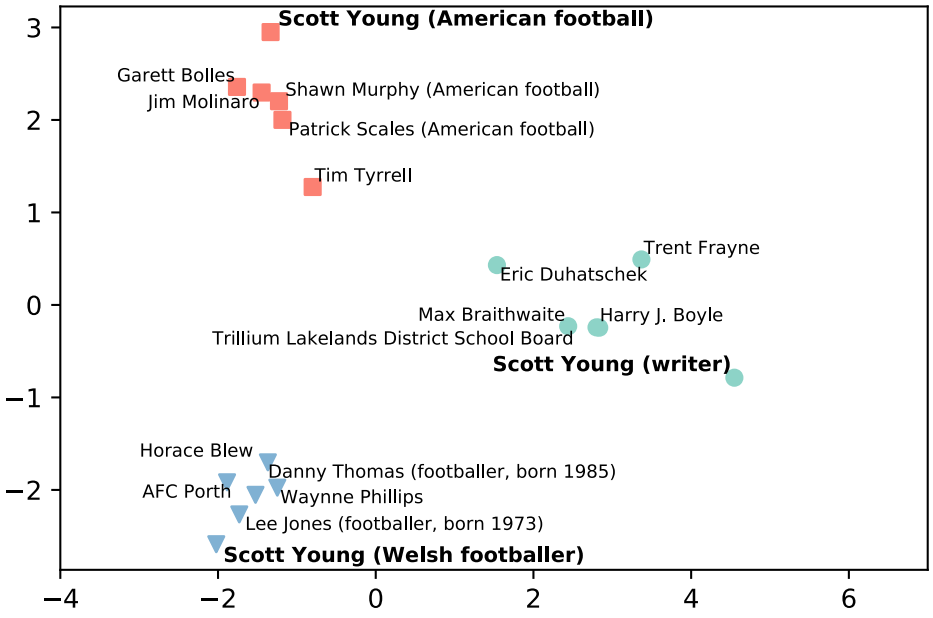


**Figure 3.4: Visualization of entity embeddings.** Entity embedding space for entities related to the ambiguous entity mention "Scott Young". Three candidate entities from Wikipedia are illustrated. For each entity, their most similar 5 entities are shown in the same colors. Entity embeddings are visualized with PCA, which is utilized to reduce dimensionality (in this example, to 2D), using pre-trained embeddings provided by Yamada, Asai, Sakuma, et al. (2020)[9].

In the first category, Ganea and Hofmann (2017) collect entity-word co-occurrences statistics from two sources: entity description pages from Wikipedia; text surrounding anchors of hyperlinks to Wikipedia pages of corresponding entities. They train entity embeddings using the max-margin objective that exploits the negative sampling approach like in the word2vec model, so vectors of co-occurring words and entities lie closer to each other compared to vectors of random words and entities. Some other methods directly replace or extend mention annotations (usually anchor text of a hyperlink) with an entity identifier and straightforwardly train on the modified corpus a word representation model like word2vec (Zwicklbauer et al., 2016b, 2016a; Moreno et al., 2017; Tsai and Roth, 2016; Yamada et al., 2017). In Moreno et al. (2017), Ganea and Hofmann (2017), Tsai and Roth (2016), and Newman-Griffis et al. (2018), entity embeddings are trained in such a way that entities become embedded in the same semantic space as words (or texts i.e., sentences and paragraphs (Yamada et al., 2017)). For example, Newman-Griffis et al. (2018) propose a distantly-supervised method that expands the word2vec objective to jointly learn words and entity representations in the shared space. The authors leverage distant supervision from terminologies that map entities to their surface forms (e.g. Wikipedia page titles and redirects or terminology from UMLS (Bodenreider, 2004)).

9. We used the English 100D embeddings from https://wikipedia2vec.github.io/wikipedia2vec/pretrained

In the second category of entity encoding methods that use relations between entities in a KG, H Huang et al. (2015) train a model that generates dense entity representations from sparse entity features (e.g. entity relations, descriptions) based on the entity relatedness. Several works expand their entity relatedness objective with functions that align words (or mentions) and entities in a unified vector space (W Fang et al., 2016; Yamada et al., 2016; Yamada, Asai, Sakuma, et al., 2020; Cao et al., 2017; W Shi et al., 2020; Radhakrishnan et al., 2018), just like the methods from the first category. For example, Yamada et al. (2016) jointly optimize three objectives to learn word and entity representations: prediction of neighbor words for the given target word, prediction of neighbor entities for the target entity based on the relationships in a KG, and prediction of neighbor words for the given entity.

Knowledge graph embedding has become a leading technique, facilitating many tasks as discussed earlier in Section 2.3.3, in Chapter 2. For entity linking, two major graph embedding algorithms are widely adopted: DeepWalk (Perozzi et al., 2014) and TransE (Bordes et al., 2013).

Parravicini et al. (2019) and **Sevgili et al**. (2019) leverage DeepWalk-based graph embeddings built from DBpedia (Lehmann et al., 2015) for entity linking. Parravicini et al. (2019) use entity embeddings to compute cosine similarity scores of candidate entities in global entity linking. **Sevgili et al**. (2019) show that combining graph and text-based embeddings can slightly improve the performance of neural entity disambiguation when compared to using only text-based embeddings. For more information about this work, see Chapter 6.

Banerjee et al. (2020) and Sorokin and Gurevych (2018) utilize Wikidata-based entity embeddings as an input component of neural models along with other types of information about entities. The ablation study conducted by Banerjee et al. (2020) show that the TransE entity embeddings are the most important features for their entity linking model. They attribute this finding to the fact that graph embeddings contain rich information about the KG structure. Similarly, Sorokin and Gurevych (2018) find that without KG structure information, their entity linker experiences a big performance drop. Nedelchev et al. (2020) integrate knowledge graph embeddings built from Freebase and word embeddings in a single end-to-end model that solves entity and relation linking tasks jointly. The quantitative analysis shows that their KG-embedding-based method helps to pick correct entity candidates. Recently, J Wu et al. (2020) also utilize TransE embeddings with other types of entity embeddings, like Ganea and Hofmann (2017) or dynamic representation, to compute pairwise entity relatedness scores.

In the last category, we place methods that produce entity representations using other types of information like entity descriptions and entity types. Often, an entity encoder is a full-fledged neural network, which is a part of an entity linking architecture. Y Sun et al. (2015) use a neural tensor network to encode interactions between surface forms of entities and their category information from a KG. In the same vein, Francis-Landau et al. (2016) and Nguyen et al. (2016) construct entity representations by encoding titles and entity description pages with convolutional neural networks. In addition to a convolutional encoder for entity descriptions, Gupta et al. (2017) also include an encoder for fine-grained entity types by using the type set of FIGER (Ling and Weld, 2012). Daniel Gillick et al. (2019) construct entity representations by encoding entity page titles, short entity descriptions, and entity category information with feedforward networks. Le and Titov (2019b) use only entity type information from a KG and a simple feedforward network for entity encoding. Hou et al. (2020) also leverage entity

types. However, instead of relying on existing type sets like in (Gupta et al., 2017), they construct custom fine-grained semantic types using words from starting sentences of Wikipedia pages. To represent entities, they first average the word vectors of entity types and then linearly aggregate them with embeddings of Ganea and Hofmann (2017).

Recent works leverage deep language models like BERT (Devlin et al., 2019) or ELMo (Peters et al., 2018) for encoding entities. Nie et al. (2018) use an architecture based on a recurrent network for obtaining entity representations from Wikipedia entity description pages. Subsequently, several models adopt BERT for the same purpose (Logeswaran et al., 2019; L Wu et al., 2020) inter alia. Yamada et al. (2021) propose a masked entity prediction task, where a model based on the BERT architecture learns to predict randomly masked input entities. This task makes the model learn also how to generate entity representations along with standard word representations. Shahbazi et al. (2019) introduce E-ELMo that extends the ELMo model (Peters et al., 2018) with an additional objective. The model is trained in a multi-task fashion: to predict next/previous words, as in a standard bidirectional language model, and to predict the target entity when encountering its mentions. As a result, besides the model for mention encoding, entity representations are obtained. Mulang' et al. (2020) use bidirectional Transformers to jointly encode context of a mention, a candidate entity name, and multiple relationships of a candidate entity from a KG verbalized into textual triples: "[subject] [predicate] [object]". The input sequence of the encoder is composed simply by appending all these types of information delimited by a special separator token.

#### 3.3.1.4 Entity Ranking

The goal of this stage is given a list of entity candidates $(e_1, e_2, ..., e_k)$ from a KG and a context $C$ with a mention $M$ to rank these entities assigning a score to each of them, as in Equation 3.8, where $n$ is a number of entity mentions in a document, $k$ is a number of candidate entities.

$$\mathsf{RNK} : ((e_1, e_2, ..., e_k), C, M)^n \rightarrow \mathbb{R}^{n \times k}. \tag{3.8}$$

There are several approaches have been proposed, for more information please see the article (**Sevgili et al.**, 2022).

#### 3.3.1.5 Unlinkable Mention Prediction

The referent entities of some mentions can be absent in the KGs, e.g. there is no Wikipedia entry about *Scott Young* as a cricket player of the Stenhousemuir cricket club.[10] Therefore, an EL system should be able to predict the absence of a reference if a mention appears in specific contexts, which is known as the NIL prediction task:

$$\mathsf{NILp} : (C, M)^n \rightarrow \{0, 1\}^n. \tag{3.9}$$

Several techniques exist in the literature, more details can be found in the article (**Sevgili et al.**, 2022).

10. Information about *Scott Young* as a cricket player: https://www.stenhousemuircricketclub.com/teams/171906/player/scott-young-1828009

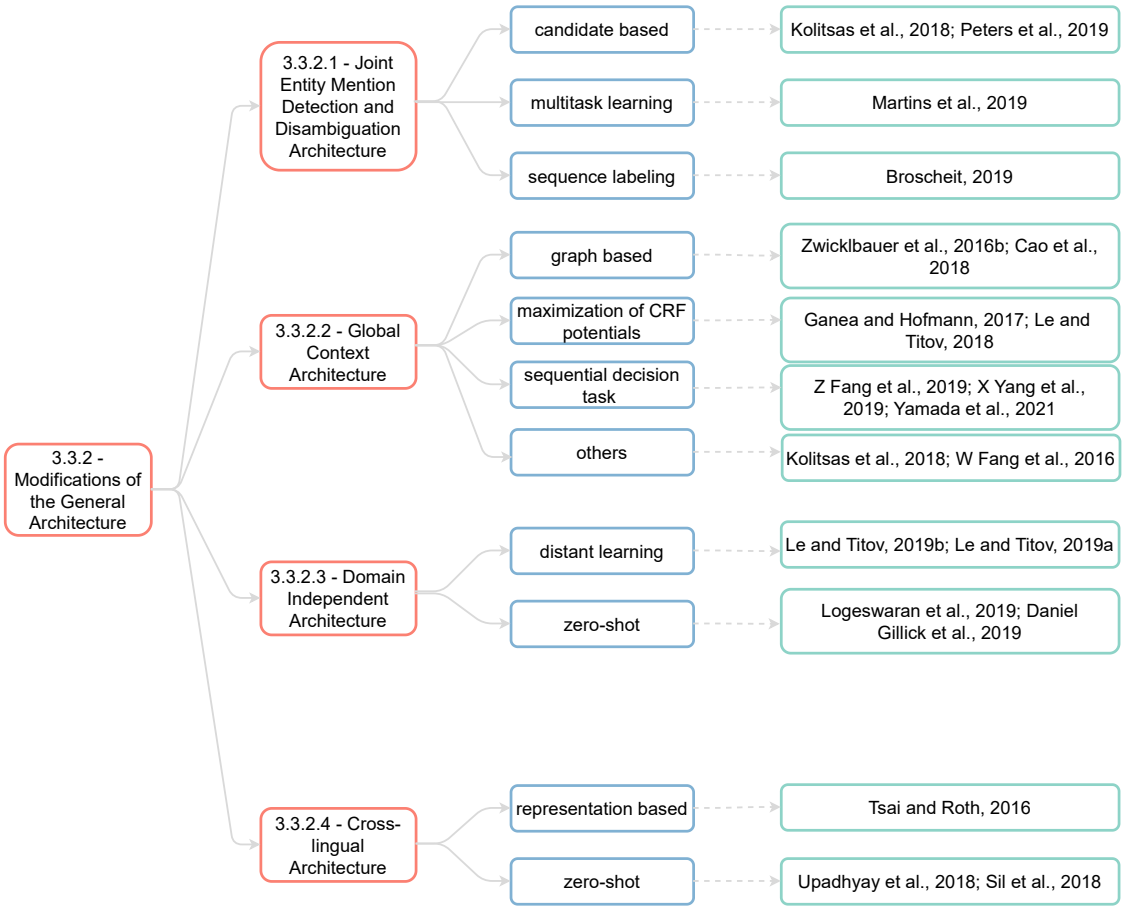


**Figure 3.5: Reference map of the modifications of the general architecture for neural EL.** The categorization of each modification with various design choices and a few example references illustrating each choice. Sections 3.3.2.3 and 3.3.2.4 are categorized based on their EL solutions, here.

## 3.3.2 Modifications of the General Architecture

This section presents the most notable modifications and improvements of the general architecture of neural entity linking models presented in Section 3.3.1 and Figures, e.g., 3.2. The categorization of each modification is summarized in Figure 3.5.

### 3.3.2.1 Joint Entity Mention Detection and Disambiguation

While it is common to separate the mention detection (cf. Equation 3.2) and entity disambiguation stages (cf. Equation 3.3), as illustrated in Figure 3.1, a few systems provide *joint* solutions for entity linking where entity mention detection and disambiguation are done at the same time by the same model. Formally, the task becomes to detect a mention $m_i \in M$ and predict an entity $e_i \in E$ for a given context $c_i \in C$, for all $n$ entity mentions in the context:

$$\mathsf{EL} : C \rightarrow (M, E)^n. \tag{3.10}$$

Undoubtedly, solving these two problems simultaneously makes the task more challenging. However, the interaction between these steps can be beneficial for improving

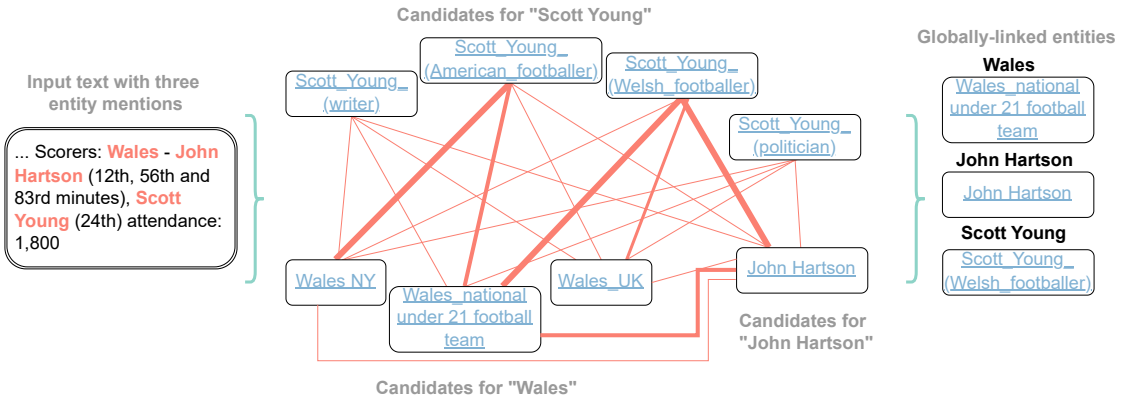


**Figure 3.6: Global entity disambiguation.** The global entity linking resolves all mentions simultaneously based on entity coherence. Bolder lines indicate expected higher degrees of entity-entity similarity.

the quality of the overall pipeline due to their natural mutual dependency. For the explanation of neural models solutions, please see the article (**Sevgili et al.**, 2022).

#### 3.3.2.2 Global Context Architectures

Two kinds of contextual information are available in entity disambiguation: local and global. In local approaches to ED, each mention is disambiguated independently based on the surrounding words, as in the following function:

$$\text{LED} : (M, C) \rightarrow E. \tag{3.11}$$

Global approaches to ED take into account semantic consistency (coherence) across multiple entities in a context. In this case, all $q$ entity mentions in a group are disambiguated interdependently: a disambiguation decision for one entity is affected by decisions made for other entities in a context as illustrated in Figure 3.6 and Equation 3.12.

$$\text{GED} : ((m_1, m_2, ..., m_q), C) \rightarrow E^q. \tag{3.12}$$

In the example presented in Figure 3.6, the consistency score between correct entity candidates: the *national football team* sense of *Wales* and the *Welsh footballer* sense of *Scott Young* and *John Hartson*, is expected to be higher than between incorrect ones.

Besides involving consistency, the considered context of a mention in global methods is usually larger than in local ones or even extends to the whole document. Although modeling consistency between entities and the extra information of the global context improves the disambiguation accuracy, the number of possible entity assignments is combinatorial (Ganea et al., 2016), which results in high time complexity of disambiguation (X Yang et al., 2019; Ganea and Hofmann, 2017). Another difficulty is an attempt to assign an entity its consistency score since this score is not possible to compute in advance due to the simultaneous disambiguation (Yamada et al., 2016).

The typical approach to global disambiguation is to generate a graph including candidate entities of mentions in a context and perform some graph algorithms, like random walk algorithms (e.g. PageRank (Page et al., 1999)) or graph neural networks, over it to select highly consistent entities (Zwicklbauer et al., 2016a, 2016b; Pershina et al., 2015; Guo and Barbosa, 2018). Recently, Xue et al. (2019) propose a neural recurrent

random walk network learning algorithm based on the transition matrix of candidate entities containing relevance scores, which are created from hyperlinks information and cosine similarity of entities. Cao et al. (2018) construct a subgraph from the candidates of neighbor mentions, integrate local and global features of each candidate, and apply a graph convolutional network over this subgraph. In this approach, the graph is static, which would be problematic in such cases that two mentions would co-occur in different documents with different topics, however, the produced graphs will be the same, and so, could not catch the different information (J Wu et al., 2020). To address it, J Wu et al. (2020) propose a dynamic graph convolution architecture, where entity relatedness scores are computed and updated in each layer based on the previous layer information (initialized with some features, including context scores) and entity similarity scores. Globerson et al. (2016) introduce a model with an attention mechanism that takes into account only the subgraph of the target mention, rather than all interactions of all the mentions in a document and restrict the number of mentions with an attention.

Some works approach global ED by maximizing the Conditional Random Field (CRF) potentials, where the first component $\Psi$ represents a local entity-mention score, and the other component $\Phi$ measures coherence among selected candidates (Ganea and Hofmann, 2017; Ganea et al., 2016; Le and Titov, 2018, 2019a), as defined in Ganea and Hofmann (2017):

$$g(e, m, c) = \sum_{i=1}^{n} \Psi(e_i, m_i, c_i) + \sum_{i<j} \Phi(e_i, e_j). \tag{3.13}$$

However, model training and its exact inference are NP-hard. Ganea and Hofmann (2017) utilize truncated fitting of loopy belief propagation (Ganea et al., 2016; Globerson et al., 2016) with differentiable and trainable message passing iterations using pairwise entity scores to reduce the complexity. Le and Titov (2018) expand it in a way that pairwise scores take into account relations of mentions (e.g. located_in, or coreference: the mentions are coreferent if they refer to the same entity) by modeling relations between mentions as latent variables. Shahbazi et al. (2018) develop a greedy beam search strategy, which starts from a locally optimal initial solution and is improved by searching for possible corrections with the focus on the least confident mentions.

Despite the optimizations proposed like in some aforementioned works, taking into account coherence scores among candidates of all mentions at once can be prohibitively slow. It also can be malicious due to erroneous coherence among wrong entities (Z Fang et al., 2019). For example, if two mentions have coherent erroneous candidates, this noisy information may mislead the final global scoring. To resolve this issue, some studies define the global ED problem as a sequential decision task, where the disambiguation of new entities is based on the already disambiguated ones with high confidence. Z Fang et al. (2019) train a policy network for sequential selection of entities using reinforcement learning. The disambiguation of mentions is ordered according to the local score, so the mentions with high confident entities are resolved earlier. The policy network takes advantage of output from the LSTM global encoder that maintains the information about earlier disambiguation decisions. X Yang et al. (2019) also utilize reinforcement learning for mention disambiguation. They use an attention model to leverage knowledge from previously linked entities. The model dynamically selects the most relevant entities for the target mention and calculates the coherence scores. Yamada et al. (2021) iteratively predict entities for yet unresolved mentions with a BERT model, while attending on

the previous most confident entity choices. Similarly, Gu et al. (2021) sort mentions based on their ambiguity degrees produced by their BERT-based local model and update query/context based on the linked entities so that the next prediction can leverage the previous knowledge. They also utilize a gate mechanism to control historical cues – representations of linked entities. Yamada et al. (2016) and Radhakrishnan et al. (2018) measure the similarity first based on unambiguous mentions and then predict entities for complex cases. Nguyen et al. (2016) use an RNN to implicitly store information about previously seen mentions and corresponding entities. They leverage the hidden states of the RNN to reach this information as a feature for the computation of the global score. Tsai and Roth (2016) directly use embeddings of previously linked entities as features for the disambiguation model. Recently, Z Fang et al. (2020) combine sequential approaches with graph based methods, where the model dynamically changes the graph depending on the current state. The graph is constructed with previously resolved entities, current candidate entities, and subsequent mention's candidates. The authors use a graph attention network over this graph to make a global scoring. As explained before, J Wu et al. (2020) also change the entity graph dynamically depending on the outputs from previous layers of a GCN. Zwicklbauer et al. (2016b) include to the candidates graph a topic node created from the set of already disambiguated entities.

Some studies, for example, Kolitsas et al. (2018) model the coherence component as an additional feedforward neural network that uses the similarity score between the target entity and an average embedding of the candidates with a high local score. W Fang et al. (2016) use the similarity score between the target entity and its surrounding entity candidates in a specified window as a feature for the disambiguation model.

Another approach that can be considered as global is to make use of a document-wide context, which usually contains more than one mention and helps to capture the coherence implicitly instead of explicitly designing an entity coherence component (Peters et al., 2019; Gupta et al., 2017; Moreno et al., 2017; Francis-Landau et al., 2016).

#### 3.3.2.3 Domain-Independent Architectures

Domain independence is one of the most desired properties of EL systems. Annotated resources are very limited and exist only for a few domains. Obtaining labeled data in a new domain requires much labor. Earlier, this problem is tackled by few domain-independent approaches based on unsupervised (H Wang et al., 2015; Cao et al., 2017; Newman-Griffis et al., 2018) and semi-supervised models (Lazic et al., 2015). Recent studies provide solutions based on distant learning and zero-shot methods. We refer reader to the article (**Sevgili et al.**, 2022), for more information.

#### 3.3.2.4 Cross-lingual Architectures

An abundance of labeled data for EL in English contrasts with the amount of data available in other languages. The cross-lingual EL (sometimes called XEL) methods (H Ji et al., 2015) aim at overcoming the lack of annotation for resource-poor languages by leveraging supervision coming from their resource-rich counterparts. Many of these methods are feasible due to the presence of a unique source of supervision for EL – Wikipedia, which is available for a variety of languages. The inter-language links in Wikipedia that map pages in one language to equivalent pages in another language

also help to map corresponding entities in different languages. For more information, please see the article (**Sevgili et al.**, 2022).

### 3.3.3 Methods that do not Fit the General Architecture

There are a few works that propose methods not fitting the general architecture presented in Figures, e.g. 3.2. Raiman and Raiman (2018) rely on the intermediate supplementary task of entity typing instead of directly performing entity disambiguation. They learn a type system in a KG and train an intermediate type classifier of mentions that significantly refines the number of candidates for the final linking model. Onoe and Durrett (2020) leverage distant supervision from Wikipedia pages and the Wikipedia category system to train a fine-grained entity typing model. At test time, they use the soft type predictions and the information about candidate types derived from Wikipedia to perform the final disambiguation. The authors claim that such an approach helps to improve the domain independence of their EL system. Kar et al. (2018) consider a classification approach, where each entity is considered as a separate class or a task. They show that the straightforward classification is difficult due to exceeding memory requirements. Therefore, they experiment with multitask learning, where parameter learning is decomposed into solving groups of tasks. Globerson et al. (2016) do not have any encoder components; instead, they rely on contextual and pairwise feature-based scores. They have an attention mechanism for global ED with a non-linear optimization as described in Section 3.3.2.2.

### 3.3.4 Summary

**Table 3.2: Features of neural EL models**. Neural entity linking models compared according to their architectural features. The description of columns is presented in the beginning of Section 3.3.4. The footnotes in the table are enumerated in the end of Section 3.3.4.

| Model | Encoder Type | Global | MD+ED | NIL Pred. | Ent. Encoder Source based on | Candidate Generation | Learning Type for Disam. | Cross-lingual |
|---|---|---|---|---|---|---|---|---|
| Y Sun et al. (2015) | CNN+ Tensor net. | | | | ent. specific info. | surface match +aliases | supervised | |
| Francis-Landau et al. (2016) | CNN | ✘[3] | | ✘ | ent. specific info. | surface match +prior | supervised | |
| W Fang et al. (2016) | word2vec-based | ✘ | | | relational info. | n/a | supervised | |
| Yamada et al. (2016) | word2vec-based | ✘ | | | relational info. | aliases | supervised | |
| Zwicklbauer et al. (2016b) | word2vec-based | ✘ | | ✘ | unstructured text + ent. specific info. | surface match | unsupervised[5] | |
| Tsai and Roth (2016) | word2vec-based | ✘ | | ✘ | unstructured text | prior | supervised | ✘ |
| Nguyen et al. (2016) | CNN | ✘ | | ✘ | ent. specific info. | surface match +prior | supervised | |
| Globerson et al. (2016) | n/a | ✘ | | | n/a | prior+aliases | supervised | |
| Cao et al. (2017) | word2vec-based | ✘ | | | relational info. | aliases | supervised or unsupervised | |
| Eshel et al. (2017) | GRU +Atten. | | | | unstructured text[1] | aliases or surface match | supervised | |
| Ganea and Hofmann (2017) | Atten. | ✘ | | | unstructured text | prior+aliases | supervised | |
| Moreno et al. (2017) | word2vec-based | ✘[3] | | ✘ | unstructured text | surface match +aliases | supervised | |
| Gupta et al. (2017) | LSTM | ✘[3] | | | ent. specific info. | prior | supervised[4] | |
| Nie et al. (2018) | LSTM +CNN | ✘ | | | ent. specific info. | surface match +prior | supervised | |
| Sorokin and Gurevych (2018) | CNN | ✘ | ✘ | | relational info. | surface match | supervised | |
| Shahbazi et al. (2018) | Atten. | ✘ | | | unstructured text | prior+aliases | supervised | |
| Le and Titov (2018) | Atten. | ✘ | | | unstructured text | prior+aliases | supervised | |
| Newman-Griffis et al. (2018) | word2vec-based | | | | unstructured text | aliases | unsupervised | |
| Radhakrishnan et al. (2018) | n/a | ✘ | | | relational info. | aliases | supervised | |
| Kolitsas et al. (2018) | LSTM | ✘ | ✘ | | unstructured text | prior+aliases | supervised | |
| Sil et al. (2018) | LSTM+ Tensor net. | | | ✘ | ent. specific info. | prior or prior+aliases | zero-shot | ✘ |
| Upadhyay et al. (2018) | CNN | ✘[3] | | | ent. specific info. | prior | zero-shot | ✘ |
| Cao et al. (2018) | Atten. | ✘ | | | relational info. | prior+aliases | supervised | |

Continued on next page

Table 3.2 – continued from previous page

| Model | Encoder Type | Global | MD+ED | NIL Pred. | Ent. Encoder Source based on | Candidate Generation | Learning Type for Disam. | Cross-lingual |
|---|---|---|---|---|---|---|---|---|
| Raiman and Raiman (2018) | n/a | ✘ | | | n/a | prior+type classifier | supervised | ✘ |
| Mueller and Durrett (2018) | GRU+ Atten. +CNN | | | | unstructured text[1] | surface match | supervised | |
| Shahbazi et al. (2019) | ELMo | | | | unstructured text | prior+aliases or aliases | supervised | |
| Logeswaran et al. (2019) | BERT | | | | ent. specific info. | BM25 | zero-shot | |
| Daniel Gillick et al. (2019) | FFNN | | | | ent. specific info. | nearest neighbors | supervised[4] | |
| Peters et al. (2019)[2] | BERT | ✘[3] | ✘ | ✘ | unstructured text | prior+aliases | supervised | |
| Le and Titov (2019b) | LSTM | | | | ent. specific info. | surface match | weakly-supervised | |
| Le and Titov (2019a) | Atten. | ✘ | | | unstructured text | prior+aliases | weakly-supervised | |
| Z Fang et al. (2019) | LSTM | ✘ | | | unstructured text + ent. specific info. | aliases | supervised | |
| Martins et al. (2019) | LSTM | | ✘ | ✘ | unstructured text | aliases | supervised | |
| X Yang et al. (2019) | Atten. or CNN | ✘ | | | unstructured text or ent. specific. info. | prior+aliases | supervised | |
| Xue et al. (2019) | CNN | ✘ | | | ent. specific info. | prior+aliases | supervised | |
| S Zhou et al. (2019) | n/a | ✘ | | | unstructured text | prior+char.-level model | zero-shot | ✘ |
| Broscheit (2019) | BERT | | ✘ | ✘ | n/a | n/a | supervised | |
| Hou et al. (2020) | Atten. | ✘ | | | ent. specific info.+ unstructured text | prior+aliases | supervised | |
| Onoe and Durrett (2020) | ELMo+Atten. +CNN+LSTM | | | | n/a | prior or aliases | supervised[4] | |
| H Chen et al. (2020) | BERT | | ✘ | | relational info. | n/a or aliases | supervised | |
| L Wu et al. (2020) | BERT | | | | ent. specific info. | nearest neighbors | zero-shot | |
| Banerjee et al. (2020) | fastText | | ✘ | | relational info. | surface match | supervised | |
| J Wu et al. (2020) | ELMo | ✘ | | | unstructured text+ relational info. | prior+aliases | supervised | |
| Z Fang et al. (2020) | BERT | ✘ | | | ent. specific info. | surface match +aliases+ Google Search | supervised | |
| S Chen et al. (2020) | Atten. +BERT | ✘ | | | unstructured text | prior+aliases | supervised | |
| Botha et al. (2020) | BERT | | | | ent. specific info. | nearest neighbors | zero-shot | ✘ |
| Yao et al. (2020) | BERT | | | | ent. specific info. | BM25 | zero-shot | |
| BZ Li et al. (2020) | BERT | | ✘ | | ent. specific info. | nearest neighbors | zero-shot | |
| Poerner et al. (2020)[2] | BERT | ✘ | ✘ | ✘ | relational info. | prior+aliases | supervised | |
| Fu et al. (2020) | M-BERT | | | | ent. specific info. | Google Search Google Maps | zero-shot | ✘ |
| Mulang' et al. (2020) | Atten. or CNN or BERT | ✘ | | | relational info. | prior+aliases | supervised | |
| Yamada et al. (2021) | BERT | ✘ | | | unstructured text | prior+aliases or aliases | supervised | |
| Gu et al. (2021) | BERT | ✘ | | ✘ | ent. specific info. | surface match +prior or aliases | supervised | |
| Tang et al. (2021) | BERT | | | | ent. specific info. | BM25 | zero-shot | |
| De Cao et al. (2021) | BART | ✘ | ✘ | | n/a | prior+aliases | supervised | |

Design features are summarized for neural EL models in Table 3.2 and also links to their publicly available implementations in Table 3.3. The mention encoders have made a shift to self-attention architectures and started using deep pre-trained models like BERT. The majority of studies still rely on external knowledge for the candidate generation step. There is a surge of models that tackle the domain adaptation problem in a zero-shot fashion. However, the task of zero-shot joint entity mention detection and linking has not been addressed yet. It is shown in several works that the cross-encoder architecture is superior compared to models with separate mention and entity encoders. The global context is widely used, but there are few recent studies that focus only on local EL.

Each column in Table 3.2 corresponds to a model feature. The **encoder type** column presents the architecture of the mention encoder of the neural entity linking model. It contains the following options:

- n/a – a model does not have a neural encoder for mentions / contexts.

- CNN – an encoder based on convolutional layers (usually with pooling), (see Section 2.2.5).
- Tensor net. – an encoder that uses a tensor network.
- Atten. – means that a context-mention encoder leverages an attention mechanism to highlight the part of the context using an entity candidate.
- GRU – an encoder based on a recurrent neural network and gated recurrent units (Chung et al., 2014), (see Section 2.2.2).
- LSTM – an encoder based on a recurrent neural network and long short-term memory cells (Hochreiter and Schmidhuber, 1997) (might be also bidirectional), (see Section 2.2.2).
- FFNN – an encoder based on a simple feedforward neural network, (see Section 2.2.1).
- ELMo – an encoder based on a pre-trained ELMo model (Peters et al., 2018).
- BERT – an encoder based on a pre-trained BERT model (Devlin et al., 2019), (see Section 2.2.4).
- fastText – an encoder based on a pre-trained fastText model (Bojanowski et al., 2017).
- word2vec-based – an encoder that leverages principles of CBOW or skip-gram algorithms (Le and Mikolov, 2014; Mikolov, Sutskever, et al., 2013; Mikolov, Chen, et al., 2013), (see Section 2.3.1).

Note that the theoretical complexity of various types of encoders is different. As discussed by Vaswani et al. (2017), complexity per layer of self-attention is $O(n^2 \cdot d)$, as compared to $O(n \cdot d^2)$ for a recurrent layer, and $O(k \cdot n \cdot d^2)$ for a convolutional layer, where $n$ is the length of an input sequence, $d$ is the dimensionality, and $k$ is the kernel size of convolutions. At the same time, the self-attention allows for a better parallelization than the recurrent networks as the number of sequentially executed operations for self-attention requires a constant number of sequentially executed operations of $O(1)$, while a recurrent layer requires $O(n)$ sequential operations. Overall, estimation of the computational complexity of training and inference of various neural networks is certainly beyond the scope of the goal of this thesis. The interested reader may refer to (Vaswani et al., 2017) and specialized literature on this topic, e.g. (Orponen, 1994; Šíma and Orponen, 2003; Livni et al., 2014).

The **global** column shows whether a system uses a global solution (see Section 3.3.2.2). The **MD+ED** column refers to joint entity mention detection and disambiguation models, where detection and disambiguation of entities are performed collectively (Section 3.3.2.1). The **NIL prediction** column points out models that also label unlinkable mentions. The **entity embedding** column presents which resource is used to train entity representations based on the categorization in Section 3.3.1.3, where

- n/a – a model does not have a neural encoder for entities.
- unstructured text – entity representations are constructed from unstructured text using approaches based on co-occurrence statistics developed originally for word embeddings like word2vec (Mikolov, Sutskever, et al., 2013).
- relational info. – a model uses relations between entities in KGs.
- ent. specific info. – an entity encoder uses other types of information, like entity descriptions, types, or categories.

In the **candidate generation** column, the candidate generation methods are specified (Section 3.3.1.1). It contains the following options:

- n/a – the solution that does not have an explicit candidate generation step (e.g. the method presented by Broscheit (2019)).
- surface match – surface form matching heuristics.
- aliases – a supplementary aliases for entities in a KG.
- prior – filtering candidates with pre-calculated mention-entity prior probabilities or frequency counts.
- type classifier – Raiman and Raiman (2018) filter candidates using a classifier for an automatically learned type system.
- BM25 – a variant of TF-IDF to measure similarity between a mention and a candidate entity based on description pages.
- nearest neighbors – the similarity between mention and entity representations is calculated, and entities that are nearest neighbors of mentions are retrieved as candidates. L Wu et al. (2020) train a supplementary model for this purpose.
- Google search – leveraging Google Search Engine to retrieve entity candidates.
- char.-level model – a neural character-level string matching model.

The **learning type for disambiguation** column shows whether a model is *'supervised', 'unsupervised', 'weakly-supervised', or 'zero-shot'*, (see Section 2.1). The **cross-lingual** column refers to models that provide cross-lingual EL solutions (Section 3.3.2.4).

In addition, the following superscript notations are used to denote specific features of methods shown as a note in the Table 3.2:

1. These works use only entity description pages, however, they are labeled as the first category (unstructured text) since their training method is based on principals from word2vec.
2. The authors provide EL as a subsystem of language modeling.
3. These solutions do not rely on global coherence but are marked as "global" because they use document-wide context or multiple mentions at once for resolving entity ambiguity.
4. These studies are domain-independent as discussed in Section 3.3.2.3.
5. Zwicklbauer et al. (2016b) may not be accepted as purely unsupervised since they have some threshold parameters in the disambiguation algorithm tuned on a labeled set.

## 3.4 Evaluation

In this section, we present evaluation results for the entity linking and entity relatedness tasks on the commonly used datasets.

**Table 3.3**: Publicly available implementations (either provided in the paper or available at PapersWithCode.com) of the neural models presented in Table 3.2.

| Model | Link for Source Code |
|---|---|
| Y Sun et al. (2015) | - |
| Francis-Landau et al. (2016) | https://github.com/matthewfl/nlp-entity-convnet |
| W Fang et al. (2016) | - |
| Yamada et al. (2016) | https://github.com/wikipedia2vec/wikipedia2vec |
| Zwicklbauer et al. (2016b) | https://github.com/quhfus/DoSeR |
| Tsai and Roth (2016) | - |
| Nguyen et al. (2016) | - |
| Globerson et al. (2016) | - |
| Cao et al. (2017) | https://github.com/TaoMiner/bridgeGap |
| Eshel et al. (2017) | https://github.com/yotam-happy/NEDforNoisyText |
| Ganea and Hofmann (2017) | https://github.com/dalab/deep-ed |
| Moreno et al. (2017) | - |
| Gupta et al. (2017) | https://github.com/nitishgupta/neural-el |
| Nie et al. (2018) | - |
| Sorokin and Gurevych (2018) | https://github.com/UKPLab/starsem2018-entity-linking |
| Shahbazi et al. (2018) | - |
| Le and Titov (2018) | https://github.com/lephong/mulrel-nel |
| Newman-Griffis et al. (2018) | https://github.com/OSU-slatelab/JET |
| Radhakrishnan et al. (2018) | https://github.com/priyaradhakrishnan0/ELDEN |
| Kolitsas et al. (2018) | https://github.com/dalab/end2end_neural_el |
| Sil et al. (2018) | - |
| Upadhyay et al. (2018) | https://github.com/shyamupa/xelms |
| Cao et al. (2018) | https://github.com/TaoMiner/NCEL |
| Raiman and Raiman (2018) | https://github.com/openai/deeptype |
| Mueller and Durrett (2018) | https://github.com/davidandym/wikilinks-ned |
| Shahbazi et al. (2019) | - |
| Logeswaran et al. (2019) | https://github.com/lajanugen/zeshel |
| Daniel Gillick et al. (2019) | https://github.com/google-research/google-research/tree/master/dense_representations_for_entity_retrieval |
| Peters et al. (2019) | https://github.com/allenai/kb |
| Le and Titov (2019b) | https://github.com/lephong/dl4el |
| Le and Titov (2019a) | https://github.com/lephong/wnel |
| Z Fang et al. (2019) | - |
| Martins et al. (2019) | - |
| X Yang et al. (2019) | https://github.com/YoungXiyuan/DCA |
| Xue et al. (2019) | https://github.com/DeepLearnXMU/RRWEL |
| S Zhou et al. (2019) | https://github.com/shuyanzhou/burn_xel |
| Broscheit (2019) | https://github.com/samuelbroscheit/entity_knowledge_in_bert |
| Hou et al. (2020) | https://github.com/fhou80/EntEmb |
| Onoe and Durrett (2020) | https://github.com/yasumasaonoe/ET4EL |
| H Chen et al. (2020) | - |
| L Wu et al. (2020) | https://github.com/facebookresearch/BLINK |
| Banerjee et al. (2020) | https://github.com/debayan/pnel |
| J Wu et al. (2020) | https://github.com/wujsAct/DGCN_EL |
| Z Fang et al. (2020) | https://github.com/fangzheng123/SGEL |
| S Chen et al. (2020) | - |
| Botha et al. (2020) | http://goo.gle/mewsli-dataset |
| Yao et al. (2020) | https://github.com/seasonyao/Zero-Shot-Entity-Linking |
| BZ Li et al. (2020) | https://github.com/facebookresearch/BLINK/tree/master/elq |
| Poerner et al. (2020) | https://github.com/npoe/ebert |
| Fu et al. (2020) | http://cogcomp.org/page/publication_view/911 |
| Mulang' et al. (2020) | https://github.com/mulangonando/Impact-of-KG-Context-on-ED |
| Yamada et al. (2021) | https://github.com/studio-ousia/luke |
| Gu et al. (2021) | - |
| Tang et al. (2021) | - |
| De Cao et al. (2021) | https://github.com/facebookresearch/GENRE |

## 3.4.1 Entity Linking

### 3.4.1.1 Experimental Setup

The evaluation results are reported based on two different evaluation settings. The first setup is entity disambiguation (ED) where the systems have access to the mention boundaries. The second setup is entity mention detection and disambiguation (MD+ED) where the input for the systems that perform MD and ED jointly is only plain text. We presented their results in separate tables since the scores for the joint models accumulate the errors made during the mention detection phase.

**Table 3.4: Evaluation datasets.** Descriptive statistics of the evaluation datasets used in this thesis to compare the EL models. The values for MSNBC, AQUAINT, and ACE2004 datasets are based on the update by Guo and Barbosa (2018). The statistics for AIDA-B, MSNBC, AQUAINT, ACE2004, CWEB, and WW is reported according to (Ganea and Hofmann, 2017) (# of mentions takes into account only non-NIL entity references). The TAC KBP dataset statistics is reported according to (H Ji et al., 2010; L Wu et al., 2020; Ellis et al., 2015; H Ji et al., 2015) (# of mentions takes into account also NIL entity references).

| Corpus | Text Genre | # of Documents | # of Mentions |
|---|---|---|---|
| AIDA-B (Hoffart et al., 2011) | News | 231 | 4,485 |
| MSNBC (Cucerzan, 2007) | News | 20 | 656 |
| AQUAINT (Milne and Witten, 2008) | News | 50 | 727 |
| ACE2004 (Ratinov et al., 2011) | News | 36 | 257 |
| CWEB (Guo and Barbosa, 2018; Gabrilovich et al., 2013) | Web & Wikipedia | 320 | 11,154 |
| WW (Guo and Barbosa, 2018) | Web & Wikipedia | 320 | 6,821 |
| TAC KBP 2010 (H Ji et al., 2010) | News & Web | 2,231 | 2,250 |
| TAC KBP 2015 Chinese (H Ji et al., 2015) | News & Forums | 166 | 11,066 |
| TAC KBP 2015 Spanish (H Ji et al., 2015) | News & Forums | 167 | 5,822 |

**Datasets** We report the evaluation results of monolingual EL models on the English datasets widely-used in recent research publications: AIDA (Hoffart et al., 2011), TAC KBP 2010 (H Ji et al., 2010), MSNBC (Cucerzan, 2007), AQUAINT (Milne and Witten, 2008), ACE2004 (Ratinov et al., 2011), CWEB (Guo and Barbosa, 2018; Gabrilovich et al., 2013), and WW (Guo and Barbosa, 2018). AIDA is the most popular dataset for benchmarking EL systems. For AIDA, we report the results calculated for the test set (AIDA-B).

The cross-lingual EL results are reported for the TAC KBP 2015 (H Ji et al., 2015) Spanish (es) and Chinese (zh) datasets. The descriptive statistics of the datasets and their text genres are presented in Table 3.4 according to information reported in (Ganea and Hofmann, 2017; L Wu et al., 2020; H Ji et al., 2010; H Ji et al., 2015; Ellis et al., 2015).

**Evaluation Metrics** For the ED setting, we present micro F1 or accuracy scores reported by model authors. We note that, since mentions are provided as an input, the number of mentions predicted by the model is equal to the number of mentions in the ground truth (Shen et al., 2015), so micro F1, precision, recall, and accuracy scores are equal in this setting as explained in Shen et al. (2015):

$$F1 = Acc = \frac{\#\ correctly\ disamb.\ mentions}{\#\ total\ mentions}. \tag{3.14}$$

For the MD+ED setting, where joint models are evaluated, we report micro F1 scores based on strong annotation matching. The formulas to compute F1 scores are shown below, as described in Shen et al. (2015) and Ganea et al. (2016):

$$P = \frac{\#\ correctly\ detected\ and\ disamb.\ mentions}{\#\ predicted\ mentions\ by\ model}, \tag{3.15}$$

$$R = \frac{\#\ correctly\ detected\ and\ disamb.\ mentions}{\#\ mentions\ in\ ground\ truth}, \tag{3.16}$$

$$F1 = \frac{2 \cdot P \cdot R}{P + R}. \tag{3.17}$$

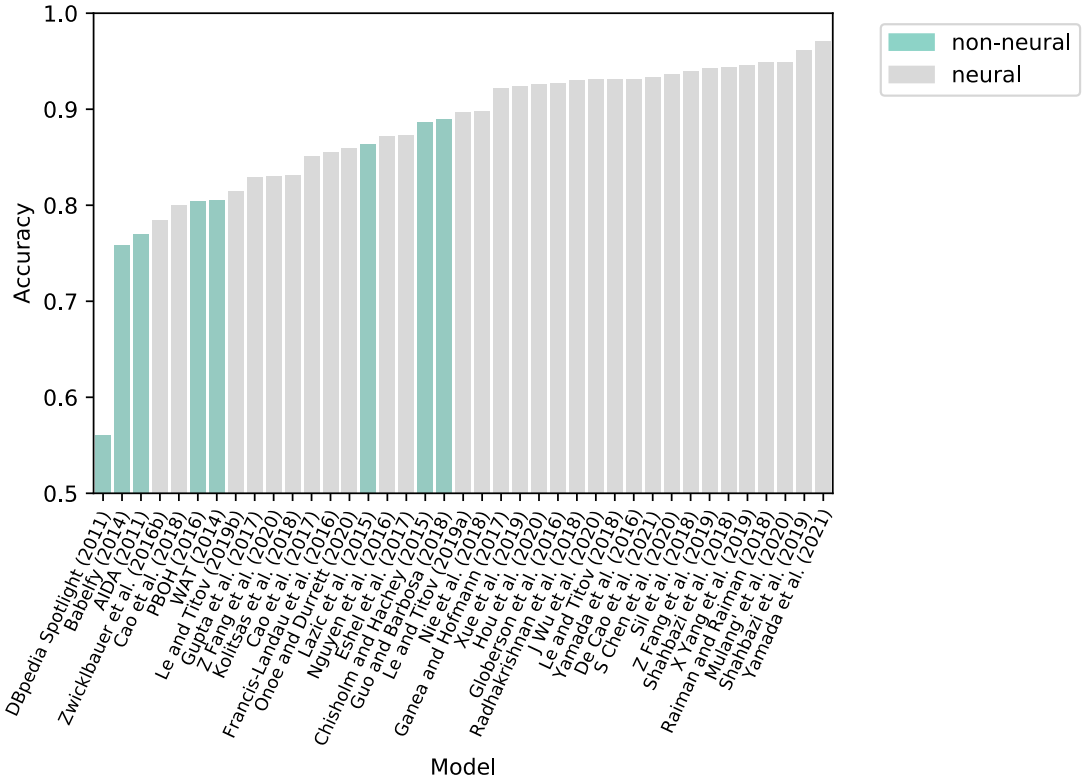


**Figure 3.7: Entity disambiguation progress**. Performance of the classic entity linking models (green) with the more recent neural models (gray) on the AIDA test set shows an improvement (around 10 points of accuracy).

We note that results reported in multiple considered papers are usually obtained using GERBIL (Röder et al., 2018) – a platform for benchmarking EL models. It implements various experimental setups, including entity disambiguation denoted as D2KB and a combination of mention detection and disambiguation denoted as A2KB. GERBIL encompasses many evaluation datasets in a standartized way along with annotations and provides the computation of evaluation metrics, i.e. micro-macro precision, recall, and F-measure.

**Baseline Models** While our goal is to perform a survey of neural EL systems, we also report results of several indicative and prominent classic non-neural systems as baselines to underline the advances yielded by neural models. More specifically, we report results of DBpedia Spotlight (2011) (Mendes et al., 2011), AIDA (2011) (Hoffart et al., 2011), Ratinov et al. (2011), WAT (2014) (Piccinno and Ferragina, 2014), Babelfy (2014) (Moro et al., 2014), Lazic et al. (2015), Chisholm and Hachey (2015) (Chisholm and Hachey, 2015), and PBOH (2016) (Ganea et al., 2016).

For considered neural EL systems, we present the best scores reported by the authors. For the baseline systems, the results are reported according to Kolitsas et al. (2018)[11] and Ganea and Hofmann (2017).

11. Some of the baseline scores are presented in the appendix of Kolitsas et al. (2018), which is available at https://arxiv.org/pdf/1808.07699.pdf

**Table 3.5: Entity disambiguation evaluation.** Micro F1/Accuracy scores of neural entity disambiguation as compared to some classic models on common evaluation datasets.

| Model | AIDA-B | KBP'10 | MSNBC | AQUAINT | ACE-2004 | CWEB | WW | KBP'15 (es) | KBP'15 (zh) |
|---|---|---|---|---|---|---|---|---|---|
| | Accuracy | Accuracy | Micro F1 | Micro F1 | Micro F1 | Micro F1 | Micro F1 | Accuracy | Accuracy |
| **Non-Neural Baseline Models** | | | | | | | | | |
| DBpedia Spotlight (Mendes et al., 2011) | 0.561 | - | 0.421 | 0.518 | 0.539 | - | - | - | - |
| AIDA (Hoffart et al., 2011) | 0.770 | - | 0.746 | 0.571 | 0.798 | - | - | - | - |
| Ratinov et al. (2011) | - | - | 0.750 | 0.830 | 0.820 | 0.562 | 0.672 | - | - |
| WAT (Piccinno and Ferragina, 2014) | 0.805 | - | 0.788 | 0.754 | 0.796 | - | - | - | - |
| Babelfy (Moro et al., 2014) | 0.758 | - | 0.762 | 0.704 | 0.619 | - | - | - | - |
| Lazic et al. (2015) | 0.864 | - | - | - | - | - | - | - | - |
| Chisholm and Hachey (2015) | 0.887 | - | - | - | - | - | - | - | - |
| PBOH (Ganea et al., 2016) | 0.804 | - | 0.861 | 0.841 | 0.832 | - | - | - | - |
| Guo and Barbosa (2018) | 0.890 | - | 0.920 | 0.870 | 0.880 | 0.770 | 0.845 | - | - |
| **Neural Models** | | | | | | | | | |
| Y Sun et al. (2015) | - | 0.839 | - | - | - | - | - | - | - |
| Francis-Landau et al. (2016) | 0.855 | - | - | - | - | - | - | - | - |
| W Fang et al. (2016) | - | 0.889 | 0.755 | 0.852 | 0.808 | - | - | - | - |
| Yamada et al. (2016) | 0.931 | 0.855 | - | - | - | - | - | - | - |
| Zwicklbauer et al. (2016b) | 0.784 | - | 0.911 | 0.842 | 0.907 | - | - | - | - |
| Tsai and Roth (2016) | - | - | - | - | - | - | - | 0.824 | 0.851 |
| Nguyen et al. (2016) | 0.872 | - | - | - | - | - | - | - | - |
| Globerson et al. (2016) | 0.927 | 0.872 | - | - | - | - | - | - | - |
| Cao et al. (2017) | 0.851 | - | - | - | - | - | - | - | - |
| Eshel et al. (2017) | 0.873 | - | - | - | - | - | - | - | - |
| Ganea and Hofmann (2017) | 0.922 | - | 0.937 | 0.885 | 0.885 | 0.779 | 0.775 | - | - |
| Gupta et al. (2017) | 0.829 | - | - | - | 0.907 | - | - | - | - |
| Nie et al. (2018) | 0.898 | 0.891 | - | - | - | - | - | - | - |
| Shahbazi et al. (2018) | 0.944 | 0.879 | - | - | - | - | - | - | - |
| Le and Titov (2018) | 0.931 | - | 0.939 | 0.884 | 0.900 | 0.775 | 0.780 | - | - |
| Radhakrishnan et al. (2018) | 0.930 | 0.896 | - | - | - | - | - | - | - |
| Kolitsas et al. (2018) | 0.831 | - | 0.864 | 0.832 | 0.855 | - | - | - | - |
| Sil et al. (2018) | 0.940 | 0.874 | - | - | - | - | - | 0.823 | 0.844 |
| Upadhyay et al. (2018) | - | - | - | - | - | - | - | **0.844** | **0.860** |
| Cao et al. (2018) | 0.800 | 0.910 | - | 0.870 | 0.880 | - | 0.860 | - | - |
| Raiman and Raiman (2018) | 0.949 | 0.909 | - | - | - | - | - | - | - |
| Shahbazi et al. (2019) | 0.962 | 0.883 | 0.923 | 0.901 | 0.887 | 0.784 | 0.798 | - | - |
| Daniel Gillick et al. (2019) | - | 0.870 | - | - | - | - | - | - | - |
| Le and Titov (2019b) | 0.815 | - | - | - | - | - | - | - | - |
| Le and Titov (2019a) | 0.897 | - | 0.922 | 0.907 | 0.881 | 0.782 | 0.817 | - | - |
| Z Fang et al. (2019) | 0.943 | - | 0.928 | 0.875 | 0.912 | 0.785 | 0.828 | - | - |
| X Yang et al. (2019) | 0.946 | - | 0.946 | 0.885 | 0.901 | 0.756 | 0.788 | - | - |
| Xue et al. (2019) | 0.924 | | 0.944 | 0.919 | 0.911 | 0.801 | 0.855 | - | - |
| S Zhou et al. (2019) | - | - | - | - | - | - | - | 0.829 | 0.855 |
| Hou et al. (2020) | 0.926 | - | 0.943 | 0.912 | 0.907 | 0.785 | 0.819 | - | - |
| Onoe and Durrett (2020) | 0.859 | - | - | - | - | - | - | - | - |
| L Wu et al. (2020) | - | **0.945** | - | - | - | - | - | - | - |
| J Wu et al. (2020) | 0.931 | - | 0.927 | 0.894 | 0.906 | **0.814** | 0.792 | - | - |
| Z Fang et al. (2020) | 0.830 | - | 0.800 | 0.880 | 0.890 | - | - | - | - |
| S Chen et al. (2020) | 0.937 | - | 0.945 | 0.898 | 0.908 | 0.782 | 0.810 | - | - |
| Mulang' et al. (2020) | 0.949 | - | - | - | - | - | - | - | - |
| Yamada et al. (2021) | **0.971** | - | **0.963** | **0.935** | **0.919** | 0.789 | **0.892** | - | - |
| De Cao et al. (2021) | 0.933 | - | 0.943 | 0.909 | 0.911 | 0.773 | 0.879 | - | - |

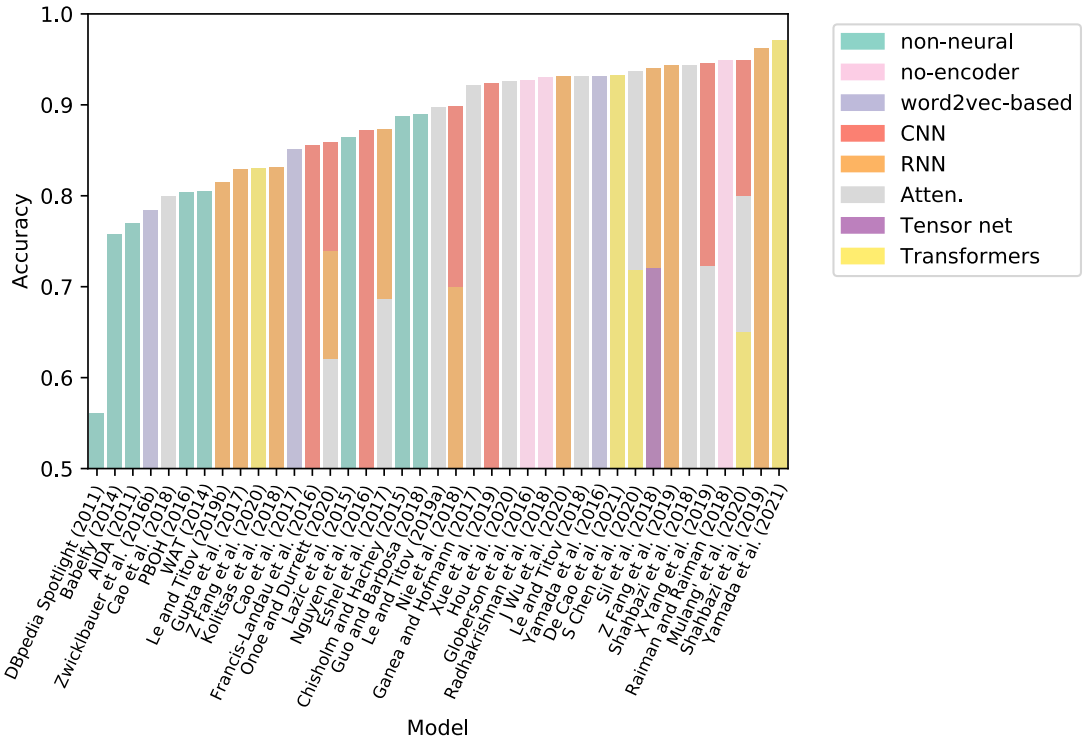


**Figure 3.8: Mention/context encoder type for entity disambiguation**. Performance of the entity disambiguation models on the AIDA test set with mention/context encoder displayed with different colors as defined in Table 3.2. The bars with multiple colors refer to the models that use different types of encoder models; the bars do not reflect any meaning on the percentage. Note: we assigned the "RNN" label for the models LSTM, GRU, and ELMo; the "Transformers" label for BERT and BART models.

#### 3.4.1.2 Discussion of Results

**Entity Disambiguation Results** We start our discussion of the results from the entity disambiguation (ED) models, for which mention boundaries are provided. Figure 3.7 shows how the performance of the entity disambiguation models on the most widely-used dataset AIDA improved during the course of the last decade and how the best disambiguation models based on classical machine learning methods (denoted as "non-neural") correspond to the recent state-of-the-art models based on deep neural networks (denoted as "neural"). As one may observe, the models based on deep learning substantially improve the EL performance pushing the state of the art by around 10 percentage points in terms of accuracy.

Table 3.5 presents the comparison of the ED models in detail on several datasets presented above. The model of Yamada et al. (2021) yields the best result on AIDA and appears to behave robustly across different datasets, getting top scores or near top scores for most of them. Here, we should also mention that none of the non-neural baselines reach the best results on any dataset.

Among local models for disambiguation, the best results are reported by Shahbazi et al. (2019) and L Wu et al. (2020). It is worth noting that the latter model can be used in the zero-shot setting. Shahbazi et al. (2019) have the best score on AIDA among other local models outperforming them by a substantial margin. However, this is due to the

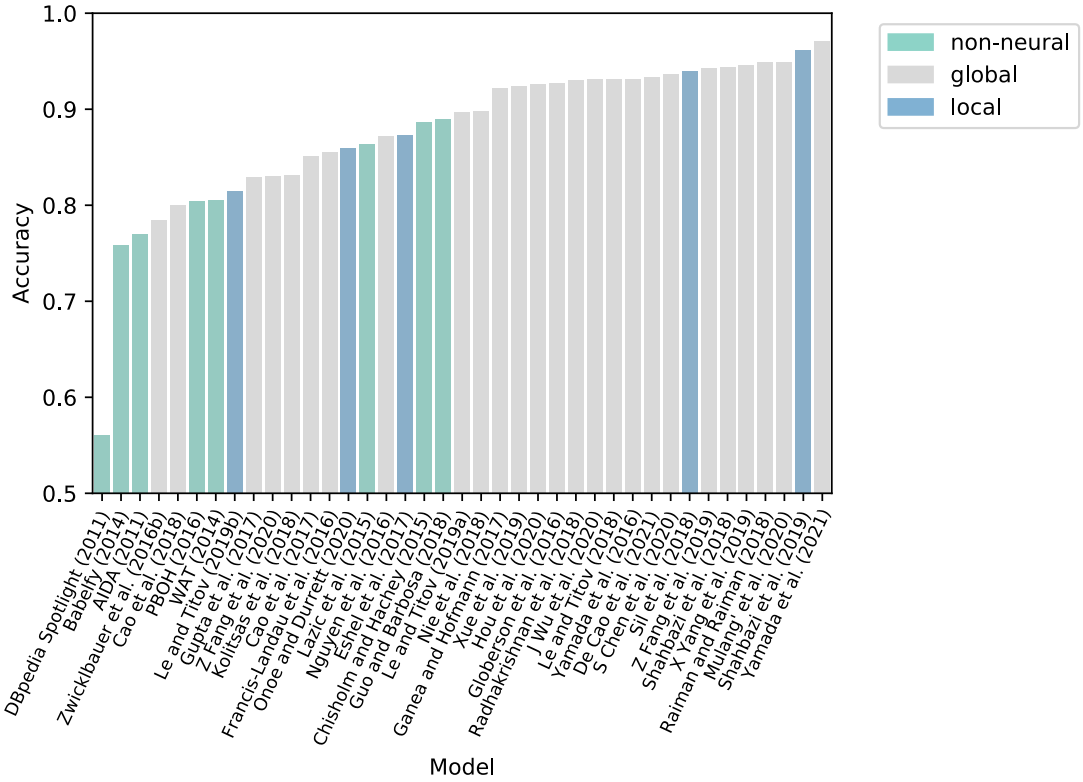


**Figure 3.9: Local-Global entity disambiguation.** Performance of the entity disambiguation models on the AIDA test set with local/global models displayed with different colors as defined in Table 3.2. Note, some models, like Francis-Landau et al. (2016), do not rely on global coherence, but they use document-wide context or multiple mentions at once, as explained in Table 3.2.

use of the less-ambiguous resource of Pershina et al. (2015) for candidate generation, while many other works use the YAGO-based resource provided by Ganea and Hofmann (2017), which typically yields lower results.

The common trend is that the global models (those trying to disambiguate several entity occurrences at once) outperform the local ones (relying on a single mention and its context). The best considered ED model of Yamada et al. (2021) is global. Its performance improvements over competitors are attributed by the authors to the novel masked entity prediction objective that helps to fine-tune pre-trained BERT for producing contextualized entity embeddings and to the multi-step global disambiguation algorithm.

Finally, as one could see from Table 3.5, the least number of experiments is reported on the non-English datasets (TAC KBP datasets for Chinese and Spanish). Among the four reported results, the approach of Upadhyay et al. (2018) provides the best scores, yet outperforming the other three approaches only by a small margin.

**Mention/Context Encoder Type** Figure 3.8 provides further analysis of the performance of entity disambiguation models presented above. The top performing model by Yamada et al. (2021) is based on Transformers. It is followed by the model of Shahbazi et al. (2019), which relies on RNNs: more specifically, it relies on the ELMo encoder that is based on pre-trained bidirectional LSTM cells. Overall, RNN is a popular choice for the mention-context encoder. However, recently, self-attention-based encoders, and especially the ones based on pre-trained Transformer networks, have gained popularity.

**Table 3.6: Entity relatedness evaluation.** Reported results for entity relatedness evaluation on the test set of Ceccarelli et al. (2013) .

| Model | nDCG@1 | nDCG@5 | nDCG@10 | MAP |
|---|---|---|---|---|
| Milne and Witten (2008) | *0.540* | *0.520* | *0.550* | *0.480* |
| H Huang et al. (2015) | **0.810** | 0.730 | 0.740 | **0.680** |
| Yamada et al. (2016) | 0.590 | 0.560 | 0.590 | 0.520 |
| Ganea and Hofmann (2017) | 0.632 | 0.609 | 0.641 | 0.578 |
| Cao et al. (2017) | 0.613 | 0.613 | 0.654 | 0.582 |
| El Vaigh et al. (2019) | *0.690* | *0.640* | *0.580* | - |
| W Shi et al. (2020) | 0.680 | **0.814** | **0.820** | - |

Several approaches, such as Yamada et al. (2016), rely on simpler encoders based on the word2vec models, yet none of them manage to outperform more complex deep architectures.

**Local-global models** Figure 3.9 visualizes the usage of the local and global context in various models for entity disambiguation. As one can observe from the plot, the majority of models perform global entity disambiguation, including the top-performing model by Yamada et al. (2021). Although Shahbazi et al. (2019) provide a local model, they also show a good performance.

### 3.4.2 Entity Relatedness

The quality of entity representations can be measured by how they capture semantic relatedness between entities (H Huang et al., 2015; Ganea and Hofmann, 2017; Yamada et al., 2016; Cao et al., 2017; W Shi et al., 2020). Moreover, the semantic relatedness is an important feature in global EL (El Vaigh et al., 2019; Ceccarelli et al., 2013). In this section, we present results of entity relatedness evaluation, which is different from evaluation of EL pipelines.

#### 3.4.2.1 Experimental Setup

We summarize results from several works obtained on a benchmark of Ceccarelli et al. (2013) for entity relatedness evaluation based on the dataset of Hoffart et al. (2011). Given a target entity and a list of candidate entities, the task is to rank candidates semantically related to the target higher than the others (Ganea and Hofmann, 2017). For the most of the considered works, the relatedness is measured by the cosine similarity of entity representations. For comparison, we also add results for two other approaches: a well-known Wikipedia hyperlink-based measure devised by Milne and Witten (2008) known as WLM and a KG-based measure of El Vaigh et al. (2019).

The evaluation metrics are normalized discounted cumulative gain (nDCG) (Järvelin and Kekäläinen, 2002) and a mean average precision (MAP) (Manning et al., 2008). nDCG is a commonly used metric in information retrieval. It discounts the correct answers, depending on their rank in predictions Manning et al. (2008):

$$nDCG(Q,k) = \frac{1}{|Q|}\sum_{j=1}^{|Q|} Z_{kj} \sum_{m=1}^{k} \frac{2^{R(j,m)}-1}{\log_2(1+m)}, \tag{3.18}$$

where $Q$ is the set of target entities (queries); $Z_{kj}$ is a normalization factor, which corresponds to ideal ranking; $k$ is a number of candidates for each query; $R(j,m) \in \{0,1\}$ is the gold-standard annotation of relatedness between the target entity $j$ and a candidate $m$.

MAP is another common metric in information retrieval (Manning et al., 2008):

$$MAP(Q) = \frac{1}{|Q|}\sum_{j=1}^{|Q|}\frac{1}{m_j}\sum_{k=1}^{m_j} Precision@r_{jk}, \tag{3.19}$$

where $Q$ is a set of target entities (queries); $m_j$ is the number of related candidate entities for the target $j$, and $Precision@r_{jk}$ is a precision at rank $r_{jk}$, where $r_{jk}$ is a rank of each related candidate in the prediction $k = 1..m_j$ (Manning et al., 2008).

#### 3.4.2.2 Discussion of Results

Table 3.6 summarizes the evaluation results in the entity relatedness task reported by the authors of the models. The scores of Milne and Witten (2008) are taken from H Huang et al. (2015).

The highest scores of nDCG@1 and MAP are reported by H Huang et al. (2015), and the best scores of nDCG@5 and nDCG@10 are reported by W Shi et al. (2020). The high scores of H Huang et al. (2015) can be attributed to the usage of different information sources for constructing entity representations, including entity types and entity relations (Ganea and Hofmann, 2017). W Shi et al. (2020) also use various types of data sources for constructing entity representations, including textual and knowledge graph information, like the types provided by a category hierarchy of a knowledge graph.

Note that cosine similarity based measures perform better in terms of nDCG@10 than the methods based on relations in KG (shown as italic in Table 3.6).

## 3.5 Conclusion

In this chapter, neural entity linking models have been analyzed, which generally solve the task with higher accuracy than classical methods. A generic neural entity linking architecture is provided, which is applicable for most of the neural EL systems, including the description of its components, e.g. candidate generation, entity ranking, mention and entity encoding. Various modifications of the general architecture are grouped into four common directions: (1) joint entity mention detection and linking models, (2) global entity linking models, (3) domain-independent approaches, including zero-shot and distant supervision methods, and (4) cross-lingual techniques. Taxonomy figures and feature tables are provided to explain the categorization and to show which prominent features are used in each method.

The majority of studies still rely on external knowledge for the candidate generation step. The mention encoders have made a shift from convolutional and recurrent models to self-attention architectures and start using pre-trained contextual language models like BERT. There is a current surge of methods that tackle the problem of adapting a model trained on one domain to another domain in a zero-shot fashion. These approaches do not need any annotated data in the target domain, but only descriptions of entities from this domain to perform such adaptation. It is shown in several works that the

cross-encoder architecture is superior as compared to models with separate mention and entity encoders. The global context is widely used, but there are few recent studies that focus only on local EL.

Among the solutions that perform mention detection and entity disambiguation jointly, the leadership is owned by the entity-enhanced BERT model (E-BERT) of Poerner et al. (2020) and the autoregressive model of De Cao et al. (2021) based on BART. Among published local models for disambiguation, the best results are reported by Shahbazi et al. (2019) and L Wu et al. (2020). The former solution leverages entity-aware ELMo (E-ELMo) trained to additionally predict entities along with words as in language-modelling task. The latter solution is based on a BERT bi-/cross-encoder and can be used in the zero-shot setting. Yamada et al. (2021) report results that are consistently better in comparison to all other solutions. Their high scores are attributed to the masked entity prediction mechanism for entity embedding and the usage of the pre-trained model based on BERT with a multi-step global scoring function.

In this chapter, we have discussed the recent entity linking models with the focus on systems that are based on the neural architectures. In a similar vein, in the following chapter, we will discuss recent ultra-fine entity typing models, yet now with the focus on the models that deal with the data scarcity challenge.

# 4

# Survey of Entity Typing Methods

**In this chapter, we introduce the ultra-fine entity typing task, and discuss its challenges, e.g. different mention forms, context dependency, or lack of data. Further, we explain promising research lines, give references, and summarize the solutions especially the ones that address data scarcity issue.**

## Contents

## 4.1 Introduction

Resolving the lexical ambiguity of entities enables various applications by clarifying the referred meaning. For instance, consider a sentence contains entity mention "Ant", which might refer to e.g. "Ant (animal)" or "Apache_Ant (software)", depending on contextual information, as discussed by Onoe and Durrett (2020). Entity typing aims to predict appropriate labels with predefined type sets. These sets contain various labels like `person, company, food, animal,` etc. Hence, assigning a `software` type for the considered entity mention "Ant" can help to clarify that the mention refers to "Apache_Ant" rather than an `animal` (Onoe and Durrett, 2020).

Entity Typing (ET) – the process of assigning semantic type labels for a mention – is a critical task, since it contributes to understanding of a text by providing a semantical information in an entity level. The task has been studied with several granularities of type sets, i.e., coarse, fine-grained, and ultra-fine grained. One can assign `animal, insect` or `software` labels for the "Ant" entity mention with the ultra-fine grained type set.

The task enhances various information extraction and natural language processing tasks (Ren, He, Qu, Huang, et al., 2016; Obeidat et al., 2019; Dan Gillick et al., 2016). The semantic type information extracted from the entity typing process has been shown to be useful in the downstream tasks such as entity linking (Onoe and Durrett, 2020), relation extraction (Yaghoobzadeh et al., 2017), question answering (Das et al., 2017).

### 4.1.1 Goal and Scope

Here, we focus on ultra-fine grained entity typing task. Our goal is to introduce the task and summarize similar studies to our work described in Chapter 5. Thus, we specifically focus on the papers, which attempt to address a lack of annotation issue in ultra-fine grained entity typing, however, we give some references for other prominent types of solutions and try to summarize a few samples. We also include works that have evaluation on some fine-grained entity typing benchmarks (i.e. OntoNotes (Dan Gillick et al., 2016), and FIGER (Ling and Weld, 2012)) for more comprehensive summarization, yet with the target of ultra-fine grained entity typing. We refer the reader to the surveys that focus on coarse (e.g. (J Li et al., 2022) in the scope of NER) or fine-grained entity typing (e.g. (Ruili Wang et al., 2023)) for the information about these tasks.

### 4.1.2 Article Collection Methodology

Since we restrict our survey on ultra-fine grained entity typing, we exclude some related works. For example, knowledge graph entity typing (e.g. (Y Zhao et al., 2020; Pan et al., 2021)), or the works that also consider mention detection, (e.g. (Aly et al., 2021)).

The papers considered in this chapter are collected from ACL Anthology[1], with the search string "ultra-fine entity typing", resulting in 139 papers[2]. From the website, we could reach the first 100 papers (sort by selected as *Relevance*). After filtering the duplicates and the links for e.g. authors, we have 77 papers. From these collection, we exclude the papers that have no evaluation on at least one of the common benchmarks UFET (Choi et al., 2018), FIGER (Ling and Weld, 2012), OntoNotes (Dan Gillick et al., 2016) that are summarized in Section 4.4.1. 30 papers are remained, however note that

1. https://aclanthology.org/
2. Search was done on 05.03.2023.

**Table 4.1: Some examples for entity typing with different granularity levels** based on the ultra fine entity typing dataset (Choi et al., 2018) and division of categories by Choi et al. (2018). The mentions in the contexts are marked as red.

| Context | Coarse | Fine | Ultra Fine |
|---|---|---|---|
| For this handpicked group of jewelry savvy Etsy artisans , their passion is The Hunger Games , the first of 3 best selling young adult books by -Suzanne Collins- . | person | author | name, writer |
| " -They- are doing this at the expense of doctors and nurses , which is laying the seeds of a revolt here . " | group, organization | government | club, committee, management, presidency, administration, arrangement, board |
| -Potatoes , tomatoes , rice tobacco , lettuce , safflowers , and other plants- have been genetically engineered to produce insulin and certain vaccines . | object | product | collection, plant, commodity, pile |

we also present a short discussion about some application scenarios, in Section 1.1 in Chapter 1. Further, we include 35 more papers from the related work sections of the collected papers[3].

## 4.2 Task Description

There are three common levels of granularities in types, coarse-grained, fine-grained and ultra-fine (Yuan et al., 2022). Some examples with contexts as classified by Choi et al. (2018) are shown in Table 4.1 and these type classes are summarized as follows:

- Coarse-grained types: the types in this level are generic with a small number of set, like `person, location, organization,` etc.
- Fine-grained types: the types contain more specific labels, like `musician, doctor, park, government,` etc.
- Ultra-fine grained types: the types involve much more specific labels, like `pop_-musician, skate_park, management,` etc.

The goal of fine-grained entity typing (FET) or ultra-fine grained entity typing (UFET) is to assign predefined semantic labels (e.g. person, organization, etc.) to each entity mention in a text (Obeidat et al., 2019; Dan Gillick et al., 2016), as exemplified in Figure 4.1. We can consider the task as determining an (U)FET function that assigns a set of type labels $(t_1, ..., t_k), t_i \in T$ for a given mention $m_i \in M$ with its context $c_i \in C$:

$$\text{(U)FET} : (M, C) \rightarrow T^k. \tag{4.1}$$

3. There are three exceptional works (L Huang et al., 2016; L Huang et al., 2017; L Liu et al., 2021) that do not conduct evaluations on the considered datasets, however, they provide unsupervised solutions. Thus, we include them.

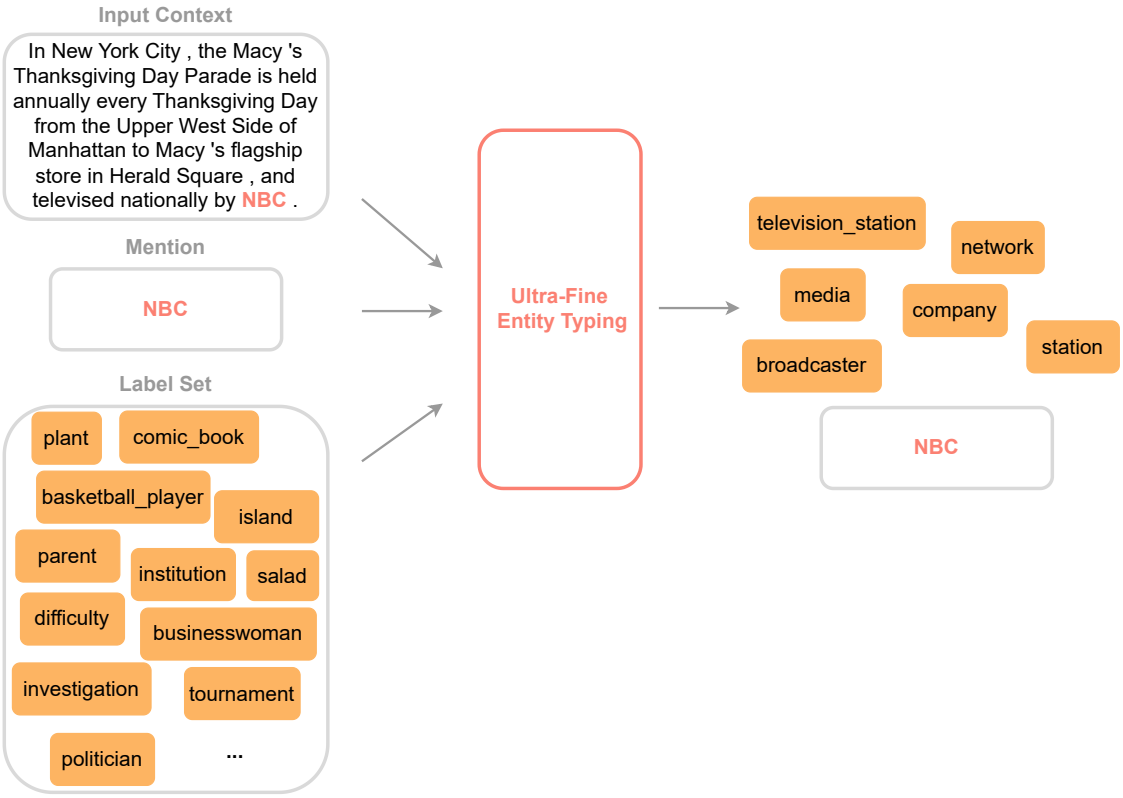


**Figure 4.1: Ultra-fine entity typing task.** An ultra-fine entity typing (UFET) model takes input context, mention, preferably label set and assigns labels to the mention.

Commonly, the task is considered as a multi-label multi-class classification (Obeidat et al., 2019; B Li et al., 2022), since the number of types are predefined. Yet, it might be treated differently, e.g. B Li et al. (2022) re-formulate the task as natural language inference (NLI) with learning-to-rank objective.

### 4.2.1 Challenges

**Hierarchy** In UFET (Choi et al., 2018), the types are free-form noun phrases in an unrestricted setting. Thus, there is no explicit dependency given in the type set, e.g. `location`, `country`, `island`, `park`, `national park`, `skate park`, etc. In OntoNotes (Dan Gillick et al., 2016) and FIGER (Ling and Weld, 2012), the types are in an ontology with a hierarchy, e.g. `/person`, `/person/engineer`, `/location/city` are some examples in FIGER and `person`, `person/artist`, `person/artist/director`, `location/city` are some types in OntoNotes.

**Mentions forms: named entity, pronoun, and nominal** In UFET (Choi et al., 2018), mentions to predict their types can be named entities, nominals, and pronouns. Pronoun mentions is especially challenging, since such mentions do not contain much cue about the labels (Choi et al., 2018). OntoNotes (Dan Gillick et al., 2016) include nominal and named entity mentions (Choi et al., 2018).

**Context dependence** In UFET (Choi et al., 2018) and OntoNotes (Dan Gillick et al., 2016), the predicted types are expected to be context dependent or sensitive. For instance,

the type for "Bill Gates" might be `philanthropist` rather than `inventor` based on the context (Choi et al., 2018).

**Lack of human-annotated data** The number of types is varying depending on the dataset. In OntoNotes (Dan Gillick et al., 2016), there are 89 types, while FIGER (Ling and Weld, 2012) contains 112 types. UFET (Choi et al., 2018) consists of more than 10k various types, making the task relatively quite challenging. Considering that high number of type sets, manual creation of the dataset, i.e. manually matching the types per mention, is remarkably difficult for humans (Dai et al., 2021).

## 4.3 Ultra-Fine Entity Typing

There are several research lines by taking reference of the related work parts of previous studies Zuo et al. (2022) and Q Liu et al. (2021). For instance;

- the studies to investigate the hierarchies or dependencies or correlations among the types.
- methods to address fine grained entity typing in different languages other than English.
- techniques to deal with lack of annotated data issue.

In the scope of this study, we focus on these directions, and specifically, the last one that can further be discussed; data generation models for mostly distant supervision, denosing strategies, zero-shot and unsupervised methods. We give some citations for others. Note that some works can try to resolve several directions in one study, e.g. Ren, He, Qu, Huang, et al. (2016) propose a model to reduce the noise and incorporate type correlation.

### 4.3.1 Scarcity of Annotated Data

#### 4.3.1.1 (Distant Supervision) Data Generation Models

A general trend to cope with the the obstacle of manually created data scarcity is to get help from automatically generated data (Dai et al., 2021). A typical way to generate distant supervision is through the KBs by linking the mention to the entity and leverage the type information of the entity in the KB (Choi et al., 2018). Choi et al. (2018) utilize the head words of mentions as an additional distant supervision source to entity linking based one. For instance, the labels from the head word of mention "national radio station NET" are `radio, station, radio_station`. X Ren et al. (2017) also relies on entity linking to use KB information as distant supervision, however, with the goal to resolve jointly extraction of entity typing and relation. Qian et al. (2021) propose a method to create a distant supervision data without KB access. They use Hearst patterns (Hearst, 1992) to derive mention-type pairs from a large unstructured data. Some mentions in the extracted pairs can refer to more than one type, i.e. "apple" can be `fruit` or `company`, so they cluster the mentions to resolve this ambiguity, and then assign type name for every cluster. For example, the cluster *Apple, Microsoft, Google, Facebook, ...* is assigned to `company` type. Their last step is to match the given mention with this created dictionary, which involves look-up the dictionary and apply some

heuristics. By using this created weak corpus, they apply self-training to increase the generalization ability. Shujian Zhang et al. (2021) investigate uneven label distribution scheme on training samples, which would contain unlabeled, single-label and multi-label examples, and explore that having multiple labels per sample is more effective than labeling many samples. They only modify the training set of UFET and simulate different label distributions without collecting new annotations.

There are some recent techniques that take advantage from pre-trained language models (PLMs). Dai et al. (2021) leverage PLMs to extract the type information by modifying the input context and feed it to masked language model (MLM). They insert several tokens close to the mention for hypernym extraction, using Hearst patterns (Hearst, 1992) with "[MASK]" token to obtain labels in a given sentence. For example, the modified input is e.g. "In late 2015, [MASK] such as *Leonardo DiCaprio* starred in The Revenant.", where "[MASK] such as" is inserted just before the mention "Leonardo DiCaprio". The predictions for "[MASK]" will be used as weak labels. They try to select different hypernym patterns, e.g. "and any other", "such as", "including", "and some other", etc. for different mentions. Xu et al. (2022) analyze the faithfulness and reliability in UFET, define different types of typical model biases, and then, propose technique to augment data for debiasing such biases. They find out six types of biases with PLMs, e.g. named entity mention bias, where if mentions refer frequently to specific entities in corpora, models tend to choose labels of these entities without considering contextual information. They propose the counterfactual data augmentation with different strategies and augment instances, e.g. for named entity bias, they substitute other entities with the same general type and without possessing comprehensive knowledge.

Ding et al. (2022) and B Li et al. (2022) propose models that do not need distant supervision data. B Li et al. (2022) utilize indirect supervision from pre-trained natural language inference (NLI) model, rather than requiring distant data. Each sentence is treated as premise and mention in the sentence is used to create a description for type candidate that is treated as hypothesis. For instance, the sentence, "In fact, Chrysler needs to convince investors...", for mention "Chrysler", the hypotheses are created using all labels, like "Chrysler is a *company*.", "Chrysler is a *corporation*.", "Chrysler is a *sea bird*." etc. The scores for the candidate hypotheses are computed through pre-trained NLI model. Other than "is a", they have several other templates. They also experiment with zero- and few-shot examples discussed in Section 4.3.1.3. Ding et al. (2022) explore the application of prompt-learning to entity typing and leverage PLMs to get type predictions by building prompts, for example "In this sentence, *mention* is [MASK]" is appended to the sentence after replacing with the real mention. This sample is for hard-encoding template, there are several more samples, and also soft-encoding templates. The MLM problem is converted to classification one by mapping labels to the vocabulary words. They only used manually labeled data. They provide solutions for full supervised and few/zero-shot scenarios, later discussed in Section 4.3.1.3.

#### 4.3.1.2 Strategies to Deal with Noise

Automatically generated distant data can contain noisy labels, for example by Onoe and Durrett (2019) "Djokovic lost to Rafael Nadal on Monday, in a rain-delayed U.S. Open final.", for the mention "Rafael Nadal", generated labels are `player, tennis player`, however, types `champion, achiever, winner, contestant, person, athlete` are also appropriate. Hence, distant data can miss the types or generate incorrect types

(Onoe and Durrett, 2019). Even manually annotated samples can contain noise, e.g. by Pan et al. (2022) "After first attempting to write the graphical routines in C, he turned to assembly language.", where "he" is annotated as `person`, and annotators failed to assign also `programmer` type (Pan et al., 2022).

In the literature, many models have been proposed to deal with noisy data (Pang et al., 2022; Ren, He, Qu, Voss, et al., 2016; J Wu et al., 2019; J Xin et al., 2018; H Zhang et al., 2020; B Chen et al., 2019) inter alia. For instance, Onoe and Durrett (2019) propose two functionalities, 1) filter the samples that are not useful – a binary classifier, 2) relabel noisy labels for the retained samples. They are learned by synthetically adding noise to manually annotated data and relying on mention and entity encoders. The augmented denoised data with manually annotated one is used to train typing model. Recently, Pan et al. (2022) work also on relabeling the noisy labels, however, the process is different. They first identify potentially noisy samples based on the observation that model might not separate some labels, in the early training phase. Hence, model is ambiguous on such labels and they select potentially noisy labels by identifying these ambiguous labels. Then, they exclude potentially noisy labels while training typing model and relabel candidate noisy labels with trained model with clean data.

#### 4.3.1.3 Zero-shot and Unsupervised Methodologies

Most of the solutions in the category of zero-shot fine grained entity typing rely on representations of types and mentions (mostly integrated with contexts). The correct types are selected through the scores computed based on these representations. Unseen types are represented with the same technique as the seen ones and so can easily be incorporated. We will discuss representation techniques of each model and the different component(s) if the model has.

The type embeddings of Ma et al. (2016) are created through the multiplication of prototype and hierarchical embeddings. Prototype, here, is the subset of mentions, selected via PMI based scoring per type and prototype embeddings are built by averaging word embeddings. Hierarcial embeddings are sparse representations with 1 if the label is parent, 0 otherwise. For zero-shot setup, the prototypes per unseen types are created manually. For mention/context representation, they use feature vector. In the work by Yuan and Downey (2018), mentions and types are represented as an average of the word embeddings of words in mentions and types, while context embeddings are generated through a bi-LSTM-based model. They also leverage additional features, e.g. syntactic. In a similar vein, Obeidat et al. (2019) generate mention and context embeddings through word embeddings and a bi-LSTM-based model. However, types are represented using their Wikipedia pages with several ways, e.g. averaging the vectors of words appeared in the description of the Wikipedia page, averaging multiple representations generated from the bi-LSTM-based mechanism. In Y Ren et al. (2020), types are represented via sample mentions, similar to Ma et al. (2016). Mentions and contexts are represented as the previous approaches, i.e. an average of word embeddings and a bi-LSTM-based model, respectively, however, additional attention layer added to the context model to incorporate entity types. In T Zhang et al. (2020), mentions are represented by word- and character-level information via bi-LSTM, while context embeddings are based on BERT. Types are represented via BERT using type names by also including the hierarcial embeddings, like in Ma et al. (2016). Different from previous architectures, the architecture contains the memory network to transfer knowledge to unseen types.

JY Huang et al. (2022) leverage RoBERTa (Y Liu et al., 2019) to encode the context and the types. They insert special tokens before and after the mention to highlight it in the context. They also generalize the solution to other different tasks, i.e. event and relation typing, by discriminating the tasks through appending additional task descriptions to contexts, "Describe the type of <mention tokens>" for the entity typing tasks.

Y Chen et al. (2021) rather focus on integration three sources of auxiliary information and investigation of their impacts: 1) context consistency, 2) type hierarchy, and 3) background knowledge. For the first one, they replace entity mention in a context with [MASK] and obtains predictions from BERT. Hierarchical information is modeled using Transformer's self-attention, where a type attends only to itself and its parents, with the loss function of matching with mention, which is represented using weighted sum of ELMo word embeddings (Peters et al., 2018). They use prototypes (mentions for the type) or descriptions (from WordNet) in NLI framework for the background knowledge. B Li et al. (2022) also leverage NLI as explained earlier. They have also experiments using their model on zero- and few-shot predictions by randomly filtering out some training samples. Yuan et al. (2022) propose generative entity typing using PLMs with the curriculum learning strategy. With PLM based such model, they are capable of conceptual reasoning and handling few- and zero-shot dilemma.

While above methods utilize training data, some studies requires no annotated data. B Zhou et al. (2018) propose a model, which relies on Wikipedia entities and Freebase types. Their idea is to find reference type-compatible entities for the mention with the definition of type-compatibility: if two entities share at least one type, then they are type-compatible. After finding the type-compatible entities, they apply their inference algorithm to use the types of the entities. Note that they do not need to find the exact entity as in EL, rather they attempt to ground type-compatible entries, with which they can produce types even though the entity is not in the Wikipedia. In zero-shot scenario of Ding et al. (2022), they provide self-supervised strategy, which inputs mention detected dataset, where the types are not labeled. Their idea is to make the prediction distribution of the same type of mentions similar. They also rely on the idea of same mentions in different sentences have similar types to develop sampling strategy. They perform contrastive learning while training. The studies (L Huang et al., 2016; L Huang et al., 2017) provide unsupervised fine-grained entity typing. Their models rely on mention representations which are created through the knowledge base information, e.g. DBpedia, contextual and semantic information of mention. Semantic information is captured with the embeddings, while contextual information is based on the relation of mention in the context. For knowledge base information, they first build a graph containing entities and their properties and type labels, then apply graph embeddings algorithm to it, and finally leverage this information by linking mention to the entity. The type labels are extracted after applying the clustering algorithm using these mention representations. L Liu et al. (2021) propose an NLP system that supports several functionalities, including an unsupervised solution by applying Hearst patterns and clustering.

### 4.3.2 Relationships in Type Labels

There are correlations between type labels, consider the scenario by Xiong et al. (2019) that a mention is assigned `criminal` label, and so one can infer that `person` is also appropriate label, instead of e.g. `police officer`. Capturing such relations is more

challenging when the type hierarchies are unavailable (Jiang et al., 2022), e.g., in UFET dataset (Choi et al., 2018), no type hierarchy is provided, while in OntoNotes (Dan Gillick et al., 2016) and FIGER (Ling and Weld, 2012), it is available.

There exist many models to address the challenge of dealing with type correlations (Murty et al., 2018; López et al., 2019; Q Liu et al., 2021; Xu and Barbosa, 2018; T Chen et al., 2020; Zuo et al., 2022; Ren, He, Qu, Huang, et al., 2016) to name some. For instance, Jiang et al. (2022) utilize a pairwise conditional random field, while Xiong et al. (2019) leverage a graph propagation layer to capture the label correlations. Onoe et al. (2021) provide a model, where embeddings are situated in n-dimensional hyperrectangles, rather than a usual vector space, to be able to capture the (implicit) hierarchies or relations between types (type-type) and mentions (type-mention). Mention and its context are first encoded by BERT and then projected down to the dimension of the box space. The model learns the parameters based on a conditional probability, which is computed as the intersection between type box and mention-context box.

### 4.3.3 Different Languages

The majority of models on fine-grained entity typing cope with English datasets, yet there are some efforts for different languages, e.g. Chinese (Lee et al., 2020), and cross-lingual solution (Han et al., 2022). In Lee et al. (2020), authors provide fine-grained entity typing dataset with a similar policy to UFET (Choi et al., 2018) but in Chinese. In Han et al. (2022), a cross-lingual model is proposed for low-resource languages by using high-resource languages data.

### 4.3.4 Others

There are some other models (M Chen et al., 2019; Y Chen et al., 2022; Sheng Zhang et al., 2018; Shimaoka et al., 2016, 2017) inter alia that also attempt to handle different kinds of problems in ultra-fine or fine-grained entity typing.

## 4.4 Evaluation

### 4.4.1 Datasets

- FIGER (Ling and Weld, 2012): They provide a set of 112 types that are curated from Freebase types. As explained earlier, there is a hierarchy in the type set and there are two levels in the types. For instance, `/organization/educational_institution`, `/art/film`, `/event/sports_event`, `/location/island`, `/animal`, `/building/hospital`, etc.

- OntoNotes (Dan Gillick et al., 2016): They also derive types in Freebase by additionally organize them into a hierarchy. In this dataset, there are 89 types with three levels, e.g. `location/structure/hospital`, `/location/geography/island`, `other/event/sports event`, `other/living thing/animal`, `person/athlete`, `other/product/camera`, etc.

- UFET (Choi et al., 2018): There is no hierarchy with 10331 free-form phrases as labels. There are various type labels, e.g. `astronaut`, `bass_guitar`, `chocolate_`-

bar, diagnosis, extreme_sport, fashion_designer, green_tea, heart_rate, information_technology, jet_engine, knowledge, literature, monument, natural_science, etc.

Note that there are more datasets, e.g. BBN (Weischedel and Brunstein, 2005), HYENA (Yosef et al., 2012), however, in the scope of this thesis, we focus on above datasets. See the work by Ruili Wang et al. (2023) for more information.

### 4.4.2 Evaluation Metrics

UFET (Choi et al., 2018) report macro-averaged precision, recall, and F1, and the average mean reciprocal rank. FIGER (Ling and Weld, 2012) utilize F1 computed through "strict" precision and recall, "loose macro" precision and recall, "loose micro" precision and recall.

## 4.5 Conclusion

In this chapter, we discuss the fine- and ultra-fine entity typing task. We highlight several directions of research, e.g. models to address the data scarcity issue, or deal with the correlations between labels. We summarize the techniques, especially, for the data scarcity issue, with a focus of data generation models and zero-shot or unsupervised models. Finally, we briefly discuss evaluation datasets and metrics.

All in all, we introduce the task with the survey of the models that address a lack of data challenge, in this chapter. In the next chapter, we will present our study on an unsupervised method for this specific challenge of the data scarcity in ultra-fine entity typing.

# 5

# Unsupervised Ultra-Fine Entity Typing

**The lack of annotated data is one of the challenging issues in an ultra-fine entity typing, as discussed in the previous chapter, Chapter 4. Hence, automatic type generation is receiving increased interest, typically to be used as distant supervision data. In this study, we investigate an unsupervised way based on distributionally induced word senses. The types or labels are obtained by selecting the appropriate sense cluster for a mention. The content of this chapter' version was published as (Sevgili et al., 2024), edited to fit in the thesis, e.g. some contents are moved to some other chapter, or excluded, added several tables, corrected language issues, etc.**

## Contents

## 5.1 Introduction

Ultra-fine entity typing (UFET) is the task of assigning semantic types to an entity mention in context (Choi et al., 2018). There exist numerous diverse types, e.g., consider the sentence – "Olympic National Park came into the national park system in 1938 and has been a favorite destination for naturalists and tourists ever since." – the types for "Olympic National Park" are `geographical_area, national_park, space, region, location, landmark, park, place`. Ultra-fine types can be helpful for natural language understanding tasks, for example, Sui et al. (2022) leverage ultra-fine entity types from entity descriptions in a zero-shot entity linking task. Yet, those large type sets lead to difficulties in annotating mentions for humans (Dai et al., 2021).

As discussed in the previous chapter, this causes a challenge of the scarcity of annotated data. We explore the leverage of distributionally induced word senses from a graph of terms (i.e. distributional thesaurus), since we believe the induced word senses can help to understand and disambiguate the given mention.

In the literature, the studies by Qian et al. (2021) and L Liu et al. (2021) are quite similar to our work as they generate labels under the setting without access to a KB rather based on a large amount of data, and the underlying techniques are quite similar, yet their evaluations are on a fine-grained entity typing (FET) task. We rather focus on UFET task with richer type set described as free-form phrases.

In this work, to produce ultra-fine types, we leverage the API of the JoBimText framework (Ruppert et al., 2015; Biemann and Riedl, 2013), which provides sense clusters with hypernym labels (i.e., IS-As) for a queried term in an unsupervised and knowledge-free way based on a distributional thesaurus. The appropriate sense for a particular mention is selected based on the cosine similarity between vectorial representations of contextual information and each sense cluster information of the mention. The hypernym labels for the selected sense are our final prediction. Our goal in this work is to explore the potential of this approach in UFET task. We experiment a combination of the neural approach predictions by Choi et al. (2018) with JoBimText based predictions to explore their complementarity. We utilize predictions from Choi et al. (2018) here, since they set the baselines while releasing the UFET dataset. With this combination, we observe a slight improvement of the F1 score for ultra-fine types for explicit mentions.

## 5.2 Related Work

### 5.2.1 Ultra-Fine Entity Typing

We refer the reader to the Chapter 4 for more discussion on the related work in the literature. Briefly, there are several lines of research on UFET (Choi et al., 2018) and FET (Ling and Weld, 2012; Dan Gillick et al., 2016). FET contains smaller set of labels, e.g., 112 types in Ling and Weld (2012), which are in an ontology, e.g., `location/city`. UFET is more diverse and finer grained, containing more than 10K labels as free-form noun phrases. Some studies investigate hierarchies/dependencies or correlations in the types in different ways (see Section 4.3.2) While most attention is on English typing, several work on other languages (see Section 4.3.3).

For the challenge of the scarcity of annotated data, many solutions have been provided (see Section 4.3.1). Among all, our study is more relevant to Qian et al. (2021),

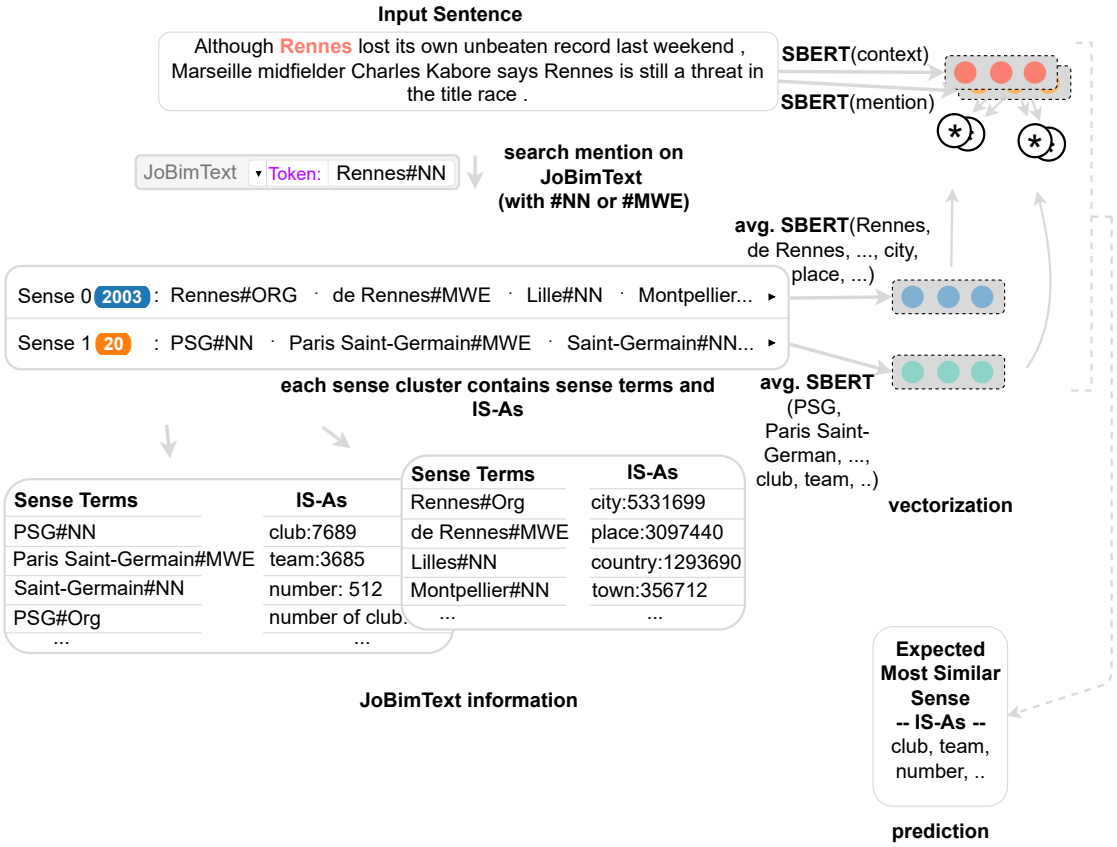


**Figure 5.1: An example prediction process**: search for a mention on JoBimText (mimicked from API) to get sense clusters containing terms and labels (IS-As), vectorize clusters by averaging SBERT vectors of each term (and label), compute cosine similarities between context (and mention) vector and clusters, and obtain IS-As of the most similar cluster as a final prediction.

in terms of applying a Hearst pattern to large data and applying the clustering without accessing the knowledge base. In a similar vein, L Liu et al. (2021) propose an NLP system that supports unsupervised FET by applying a Hearst pattern and clustering. However, both evaluate on FET task, while our focus is on UFET, which contains more fine-grained types. In Dai et al. (2021), the labels are generated automatically from PLMs, and Ding et al. (2022) provide also a zero-shot solution. In comparison, our study investigates particularly the usage of the JoBimText API on this task, which is a simpler scenario than their models. In comparison with B Zhou et al. (2018) (providing zero-shot solution based on unlabeled data), L Huang et al. (2016) and L Huang et al. (2017) (proposing an unsupervised solution), we do not use the knowledge base information.

### 5.2.2 JoBimText Applications

Several works utilize information provided by JoBimText in different tasks (Jana and Goyal, 2018; Anwar et al., 2020; Pelevina et al., 2016), inter alia. Among them, the most similar studies might be unsupervised knowledge-free word sense disambiguation by Panchenko, Marten, et al. (2017) and Panchenko, Ruppert, et al. (2017), in which a word in a context is disambiguated using the induced senses of JoBimText. Inspired by them, we conduct a similar approach to the UFET task.

**Table 5.1: Pre-processing examples**: n-gram tokens and headword token are extracted from the mention (selected in dev. set) and singularization is applied. The first two examples are to show the importance of the beginning tokens, the following two examples are for the end tokens, and last two examples are for the importance of headword. "1-gram first": the first token is taken, while n=1 for the n-gram. "1-gram last": the last token is taken, while n=1 for the n-gram. n=1,2,3 are experimented. "headword": headword for the mention. Note that in n-gram extraction, if the first token is considered with n=1 and the mention starting with a, an, the (or upper case of them), we take the second token.

| Mention | 1-gram first | 1-gram last | 2-gram first | 2-gram last | 3-gram first | 3-gram last | headword |
|---|---|---|---|---|---|---|---|
| education of the medium | education | medium | education of | the medium | education of the | of the medium | education |
| a club playing in the Fourth National division | club | division | a club | National division | a club playing | Fourth National division | club |
| the third largest LDS retail bookstore | third | bookstore | the third | retail bookstore | the third largest | LDS retail bookstore | bookstore |
| The caustic spill – Hungary 's worst ecological disaster – | caustic | disaster | The caustic | ecological disaster | The caustic spill | worst ecological disaster | spill |
| the most lightweight suits which can fit into any trend | most | trend | the most | any trend | the most lightweight | into any trend | suit |
| The eastern and western sections of the city | eastern | city | The eastern | the city | The eastern and | of the city | section |

## 5.3 Method

### 5.3.1 JoBimText Framework

In our work, we generate labels relying on the JoBimText framework (Biemann and Riedl, 2013) as an end-user of the API[1] (Ruppert et al., 2015) provided by this framework. The underlying technology of this framework is explained in Section 2.4.2 in Chapter 2.

### 5.3.2 Method

In our setup, we query the JoBimText API for a mention and obtain sense clusters of this mention. The most appropriate sense cluster is selected based on the vectorial similarity between the context that the mention appeared in, and a sense cluster. To compute this similarity, each sense cluster[2] is vectorized by using the sense terms (and labels) with Sentence BERT (SBERT) (Reimers and Gurevych, 2019) (see Section 2.3.2, for more information about SBERT). Context and mention, separately, are also vectorized using SBERT. The hypernym labels of the most similar sense are the final type predictions, as exemplified in the Figure 5.1.

We query a mention with NN (noun) or MWE (multi-word expression) tag, depending on whether a mention contains a single token or multiple tokens. Other tags, e.g., ADJ, are not considered since the mentions in the UFET dataset (Choi et al., 2018) are pronouns, nominal expressions, and named entity mentions. For named entity mentions, some other tags, like e.g. Organization, Location, would be helpful, however, the goal of entity typing itself to produce such labels, and so, we use only NN/MWE tags. JoBimText might not provide information for all mentions, like long phrases, e.g., "the building , a violation of the Clean Air Act". Thus, we try to shorten the mentions in several ways: extraction of a head word (root word) of the mention or extraction of n-grams located

1. http://ltmaggie.informatik.uni-hamburg.de/jobimviz/#
2. Note that some clusters may not have hypernyms/terms, so we skip them.

**Table 5.2: Post-processing examples**: The labels of random cluster for the searched headword are shown, in "labels" column. Each label is singularized, lowerized, and added underscore if it contains more than one token, as displayed in the next column. Finally, the post-processed labels not in the type vocabulary are filtered.

| Mention | headword | random cluster id | labels | labels-postprocess: singularized lowerized and added underscore | labels-postprocess: filtered based on type vocab. |
|---|---|---|---|---|---|
| the Senate | Senate | 4 | member, leader, Republicans, officer, official, Democrats, position, award, people, legislator, senator, leadership, lawmaker, figure, group, delegation, party, other, politician, Congress | member, leader, republican, officer, official, democrat, position, award, person, legislator, senator, leadership, lawmaker, figure, group, delegation, party, other, politician, congress | member, leader, republican, officer, official, democrat, position, award, person, legislator, senator, leadership, lawmaker, figure, group, delegation, party, other, politician, congress |
| Ferrari | Ferrari | 0 | company, brand, car, manufacturer, automaker, vehicle, name, makes, model, client, competitor, carmaker, marques, car company, car maker, Companies, car manufacturer | company, brand, car, manufacturer, automaker, vehicle, name, makes, model, client, competitor, carmaker, marque, car_company, car_maker, car_manufacturer | company, brand, car, manufacturer, automaker, vehicle, name, model, client, competitor |

close to the beginning/end of the mention due to an observation of some mentions (e.g., "shipments for the month", "the social and economic development"). Note that Choi et al. (2018) also utilize a head word of the mention, however, they directly take it as a weak label, while our goal is to shorten the mention to search later on the JoBimText. Additionally, singularization[3] is applied for the mention as an additional configuration based on the observation that the singular version of some mentions is in the JoBimText, while the plural is not, e.g., "shareholders". Some examples of pre-processing are shown in Table 5.1. JoBimText might still not provide any information for the short mentions, for which we assign a `person` label as it is the most frequent label in the development set. The coverage for our reported results, in Table 5.5, is 87.74.

In a similar vein, we apply some post-processing steps due to some mismatches between predictions (i.e., IS-As) and the type vocabulary of Choi et al. (2018) (e.g., predictions may involve `people`, while the vocabulary contains `person`, not `people`). We first follow Dai et al. (2021) for post-processing: the labels are singularized and filtered if they are not in the type vocabulary. In addition to them, we add underscores for the multi-token labels, e.g., `tennis_player`, and lower-case them. We remove a label `thing` among our predicted labels since we consider it as a noisy label due to its high frequency. Some examples of post-processing steps are shown in Table 5.2. An example overview containing pre-/post-process steps is shown in Figure 5.2.

## 5.4 Experiments

### 5.4.1 Dataset

The experiments are performed on an English UFET dataset provided by Choi et al. (2018). The type vocabulary contains 10331 labels. The dataset consists of a training set, development set, and test set, each with 1998 samples.

3. Only singularization is applied here, not lemmatization.

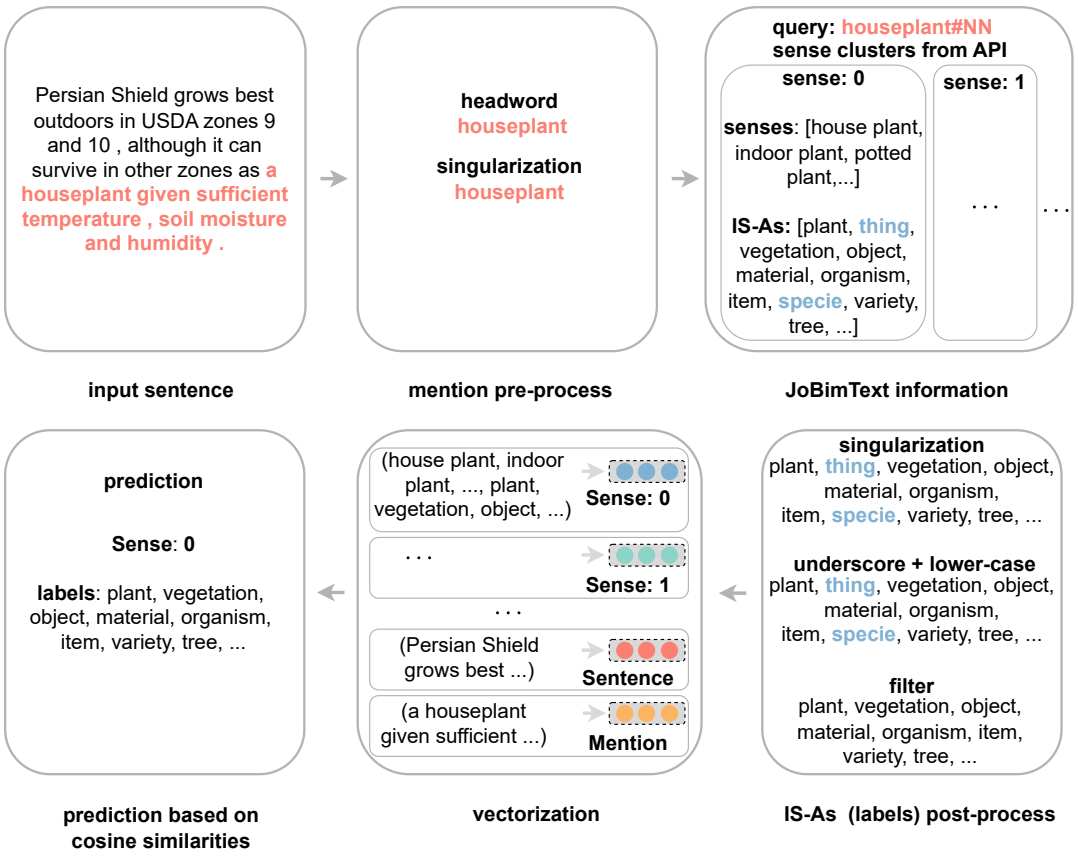


**Figure 5.2: An example overview of the process**: head word is used to shorten mention, and post-process is shown for labels of sense: 0, and IS-A labels are included for vector representation. The prediction is based on the cosine similarities of sentence and sense vectors, and mention and sense vectors.

### 5.4.2 Baselines

- **first cluster or random cluster** There is an order of sense clusters in JoBimText, based on the score of related terms. We either always choose the first cluster with terms and labels, or choose any sense randomly. The same pre- and post-process steps are applied as the configuration, which will be explained in 5.4.3.
- **Choi et al. (2018)** generate representations through the pre-trained word embeddings, bi-directional LSTM, CNN, and train the model with a multitask objective.
- **Dai et al. (2021)** generate labels through the BERT masked language model and leverage the generated labels in the training entity typing model.
- **B Li et al. (2022)** treat each sentence as a premise and generate a hypothesis through the candidate type to formulate the task as NLI. Here, the learning objective is learning-to-rank.

### 5.4.3 Implementation Details

JoBimText provides many DTs, including different languages, from various corpora. In this study, we use the DT constructed from the DepCC corpus (Panchenko et al., 2018). It

**Table 5.3:** Parameters or features, their possible values, and the selection based on our simple manual search on the development set.

| | Parameter/Feature | Possible values | Selection |
|---|---|---|---|
| 1 | mentions shorten options | - head word<br>- n-gram beginning tokens (for n=1 - 2 - 3)<br>- n-gram end tokens (for n=1 - 2 - 3) | head word |
| 2 | apply singularization to mention | true - false | true |
| 3 | cluster types | 200,200 - 200,50 - 50,50 | 50,50 |
| 4 | number of terms for representation | 10 - 20 - 30 | 10 |
| 5 | number of labels for representation | false - 10 - 20 - 30 | 10 |
| 6 | weighting average | false - rank - cos. sim. | false |
| 7 | include mention similarity | true - false | true |
| 8 | include mention | true - false | false |
| 9 | number of predictions | 5 - 10 - 15 - 20 | 10 |

is built from the web-scale data from the Common Crawl, which provides access to large amounts of data. The DepCC model uses a 2016 snapshot of the Common Crawl[4]. As a Sentence BERT model, we utilize "all-mpnet-base-v2", since it is the best-performing one (on average performance) among current models.[5]

We consider several parameters or features and select amongst them based on a simple manual search, in our implementation, as shown in Table 5.3. In Table 5.3, the first row represents the methodology, where we shorten the mention before searching in the JoBimText API, where n-grams are extracted using NLTK (Bird et al., 2009)[6] and head words are extracted with the stanza library/toolkit[7] (Qi et al., 2020). Some mentions start with "a, an, the" (or upper case of them), for which we take the tokens after the first one, if we experiment with the beginning of mention. Punctuation symbols (like, e.g. ".", ",", """, "–", as well as the tokens "-LRB-" and "-RRB-" that refer to round brackets) are removed.

For some mentions (three mentions in the development set and eight mentions in the test set), there can be more than one head word, for which we use the first head word by default. The second item in the table is applying singularization to mentions before the search in API, for which we use inflect library[8], however, it might result in some mistakes for some cases as discussed in Section 5.5. To avoid the case that the term ends with "s" (e.g. "access"), we double-check its morphological property using the stanza toolkit whether it is singular or plural. Here, we cross-check only the last token of the mention and we dismiss the cases of the singularized word that is located in different place. For example, "princess of Brunswick-Wolfenbuttel" is singularized as "princes of brunswick-wolfenbuttell". The third row in Table 5.3 is for cluster types available in the API to determine the number of some entries[9] for the Chinese Whispers algorithm. Cluster representations are created using the sense terms, and the fourth item in the table is to determine how many terms to include.

4. https://commoncrawl.org
5. https://www.sbert.net/docs/pretrained_models.html
6. We also experiment with an implementation by scikit-learn (Pedregosa et al., 2011) and report the results in the Table A.1 in Appendix A. The differences are minor, like the automatic punctuation removal (e.g. re-sign vs. re sign).
7. https://github.com/stanfordnlp/stanza
8. https://pypi.org/project/inflect
9. http://ltmaggie.informatik.uni-hamburg.de/jobimtext/documentation/sense-clustering

Similarly, the cluster labels are optionally included while creating representations, as shown in the 5th row with the number options.

While averaging, weighting is possible with weights either from the similarity between a mention and a considered term/label or from the ranking from JoBimText using 1/rank (meaning if it is ranked first, weight becomes 1, second: 0.5, third: 0.33, fourth: 0.25..), as in the sixth entry in the table. While computing similarities for the final decision, similarities are between a context and a sense, or also including the similarities between the mention and a sense. Item eight is to determine that the context contains mention or not. The last entry is for the number of predicted labels.

**Parameter Search:** We try to find the best parameters and features in the development set with a simple manual search, discussed in Table 5.4. Among the experiments, the configuration with the best F1 score, which we can reach so far, consists of the parameters and features, as shown in the last column of Table 5.3.

In Table 5.4, we show several results based on some simple different parameter configurations. As a mention shorthened method, headword reaches highest F1 scores compared to its other counterparts, while first token (first-n1) is good at coverage, the performance is not that promising. The P (precision) scores for first two (first-n2) and three (first-n3) are quite high, which could be due to coverage. Meanly, since for many samples, the system has no response and use the default prediction `person`, the number of labels is one in such cases and this would increase precision scores. Singularizing the headword before search on JoBimText shows improvements scores on fine-, and ultra-fine types, as well as on coverage. Including isas (hypernym) words while computing the sense representations increase the scores, while adding weights during averaging the sense terms/isas results in little decrease. The reason might be that an impact of a mention on individual sense term/isas is misleading, since a contextual information is important for disambiguation. Applying a lowerization to the search mention (headword) makes the coverage to the highest, however, the scores decrease, which shows that the upper case also carries some hints. Including mention similarity to the context similarity while disambiguating the sense cluster has a good impact on the final predictions. In a similar vein, if the mention is inserted in between left and right context words while encoding the context, the method performs better than just encoding left and right context words as the context representation. However, if we apply both at the same time, although there is a slight rise on the fine types, coarse and total scores decrease. Cluster types of Chinese Whisper algorithm available in JoBimText would be “200,200”, “200,50”, and “50,50”, according to the results “50,50” reaches better scores according to our set-up in terms of ultra-fine scores F1. And finally, we attempt to find the best parameters of number of isas, number of terms, and number of predictions, by tuning one of them while keeping others as default over the search space – number of isas [10, 20, 30], number of terms [10, 20, 30], number of predictions [5, 10, 15, 20, 25]. Note that due to implementation if number of predictions higher than 10, number of isas should be also the same as it. Due to the high number of parameters/features, we could not try every combination and keep some parameters static/default. That’s why further improvement over here is possible to search. According to our search space on this set-up, we reach up to 9.7 F1 score on ultra-fine with parameters headword-prep-includeisas-includemensim-5050.

| Model | Total | | | Coarse | | | Fine | | | Ultra-Fine | | | Coverage |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | |
| w/o any feature | | | | | | | | | | | | | |
| last-n1 | 18.0 | 17.8 | 17.9 | 44.7 | 49.3 | 46.9 | 12.7 | 15.4 | 13.9 | 5.5 | 7.9 | 6.5 | 77.93 |
| last-n2 | 20.3 | 16.0 | 17.9 | 41.0 | 46.9 | 43.7 | 14.5 | 12.7 | 13.5 | 5.9 | 6.1 | 6.0 | 59.06 |
| last-n3 | 21.6 | 14.5 | 17.3 | 39.5 | 45.4 | 42.2 | 13.2 | 9.9 | 11.3 | 5.4 | 4.7 | 5.0 | 50.45 |
| first-n1 | 15.5 | 15.9 | 15.7 | 42.1 | 46.2 | 44.0 | 9.8 | 13.0 | 11.2 | 4.1 | 6.4 | 5.0 | **82.13** |
| first-n2 | 20.7 | 15.0 | 17.4 | 40.7 | 45.9 | 43.1 | 13.2 | 11.0 | 12.0 | 5.2 | 5.2 | 5.2 | 58.66 |
| first-n3 | 22.0 | 14.1 | 17.2 | 39.0 | 44.9 | 41.7 | 13.1 | 9.0 | 10.7 | 5.4 | 4.6 | 5.0 | 48.05 |
| headword | 18.3 | 18.5 | **18.4** | 45.9 | 50.5 | **48.1** | 12.6 | 15.6 | **14.0** | 5.9 | 8.4 | **6.9** | 78.48 |
| with mentioned features | | | | | | | | | | | | | |
| headword-prep | 16.6 | 19.6 | 18.0 | 45.6 | 49.4 | 47.4 | 12.5 | 17.2 | 14.5 | 6.2 | 10.3 | 7.7 | 88.24 |
| headword-prep-inclisas | 17.1 | 19.9 | 18.4 | 46.5 | 48.8 | 47.6 | 13.9 | 18.5 | 15.9 | 6.7 | 10.9 | 8.3 | 88.24 |
| headword-prep-inclisas-wcosine | 16.6 | 19.7 | 18.0 | 45.6 | 49.1 | 47.3 | 13.7 | 18.2 | 15.6 | 6.4 | 10.4 | 7.9 | 88.24 |
| headword-prep-inclisas-wrank | 16.7 | 19.7 | 18.1 | 46.2 | 48.6 | 47.4 | 13.5 | 18.5 | 15.6 | 6.3 | 10.5 | 7.9 | 88.24 |
| headword-prep-inclisas-low | 12.8 | 19.8 | 15.6 | 46.2 | 50.3 | 48.2 | 11.1 | 17.2 | 13.5 | 5.7 | 10.0 | 7.2 | **95.90** |
| headword-prep-inclisas-inclmensim | 19.0 | 21.9 | **20.4** | 50.2 | 54.5 | **52.3** | 14.5 | 20.4 | 16.9 | 7.8 | 12.1 | **9.5** | 88.24 |
| headword-prep-inclisas-inclmen | 17.9 | 19.9 | 18.8 | 46.6 | 48.4 | 47.5 | 14.4 | 18.0 | 16.0 | 7.5 | 11.3 | 9.0 | 88.24 |
| headword-prep-inclisas-inclmensim -inclmen | 19.1 | 21.6 | 20.3 | 50.0 | 53.9 | 51.8 | 14.8 | 20.0 | **17.0** | 7.9 | 11.9 | **9.5** | 88.24 |
| with mentioned features (on headword-prep-inclisas-inclmensim-) | | | | | | | | | | | | | |
| -with-20050 | 19.8 | 19.0 | 19.4 | 43.5 | 43.3 | 43.4 | 14.9 | 18.7 | 16.6 | 8.6 | 11.0 | 9.6 | 88.09 |
| -with-5050 | 19.7 | 18.8 | 19.2 | 43.1 | 42.3 | 42.7 | 15.0 | 19.2 | 16.9 | 8.6 | 11.0 | **9.7** | 88.14 |
| -#isas 20-#terms 10-#preds 10 | 19.1 | 21.3 | 20.1 | 49.9 | 53.6 | 51.7 | 14.0 | 19.4 | 16.2 | 7.8 | 11.7 | 9.4 | 88.24 |
| -#isas 30-#terms 10-#preds 10 | 19.1 | 21.3 | 20.1 | 49.9 | 53.6 | 51.7 | 14.0 | 19.4 | 16.2 | 7.8 | 11.7 | 9.4 | 88.24 |
| -#isas 10-#terms 20-#preds 10 | 18.3 | 21.8 | 19.9 | 48.4 | 55.3 | 51.6 | 13.8 | 19.8 | 16.3 | 7.1 | 11.8 | 8.9 | 88.24 |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| -#isas 10-#terms 30-#preds 10 | 18.4 | 21.9 | 20.0 | 48.6 | 55.6 | 51.8 | 14.2 | 19.7 | 16.5 | 7.2 | 11.9 | 9.0 | 88.24 |
| -#isas 10-#terms 10-#preds 5 | 22.6 | 17.2 | 19.6 | 55.4 | 47.3 | 51.0 | 21.6 | 16.5 | 18.7 | 10.1 | 8.3 | 9.1 | 88.24 |
| -#isas 15-#terms 10-#preds 15 | 17.6 | 23.6 | 20.1 | 46.2 | 56.0 | 50.6 | 11.2 | 21.7 | 14.8 | 6.8 | 13.6 | 9.1 | 88.24 |
| -#isas 20-#terms 10-#preds 20 | 17.4 | 23.8 | 20.1 | 45.3 | 55.7 | 50.0 | 11.1 | 22.3 | 14.8 | 6.7 | 14.2 | 9.1 | 88.24 |
| -#isas 25-#terms 10-#preds 25 | 17.4 | 23.8 | 20.1 | 45.3 | 55.7 | 50.0 | 11.1 | 22.3 | 14.8 | 6.7 | 14.2 | 9.1 | 88.24 |
| Baselines (on headword-prep-inclisas-with5050) | | | | | | | | | | | | | |
| -first-cluster | 16.8 | 18.1 | 17.5 | 33.7 | 43.1 | 37.9 | 11.2 | 16.5 | 13.3 | 6.7 | 10.2 | 8.1 | 88.14 |
| -random-cluster (avg. of 5 runs) | 16.0 | 16.2 | 16.1 | 44.0 | 43.8 | 43.9 | 10.8 | 13.2 | 11.9 | 5.5 | 7.9 | 6.5 | 88.14 |

**Table 5.4:** Simple manual parameter search for unsupervised ultra-fine entity typing performance: Results are on UFET development set. "last-n*": the last *-gram of the mention, or "first-n*": the first *-gram of the mention, or "headword": headword of the mention is searched on JoBimText. "-prep": the mention is singularized before the search. "-inclisas": while computing sense representation, hypernym words are included. "-wrank": while averaging the vectors of sense terms and isas for sense cluster vector, the weights are the rank based on their appearance in the cluster, or "-wcosine": the weights are the cosine similarity between mention and each term/isas (otherwise average w/o weights). "-low": the mention is lowerized before the search. "-inclmensim": the mention similarity and context similarity to sense clusters are averaged (otherwise only context similarity). "-inclmen": mention is included to the context (otherwise left and right words only). "-20050" and "-5050": Chinese Whisper algorithm parameters, default is "-200200" (if not shown). "-#isas n-#terms m-#preds p": number of isas and number of terms that are included to compute sense representation, and number of predictions is number of isas used for the final prediction, default is "-#isas 10-#terms 10-#preds 10" (if not shown). Baselines are also with the best (so far) features: headword-prep-inclisas-inclmensim.

**Table 5.5: Unsupervised ultra-fine entity typing performance on UFET test set**: without pronouns: the results are for the mentions that are not pronouns (1210 samples, in the test set), 5 preds.: the results contain the first five predictions from Choi et al. (2018) and our first five predictions, Ours-PRP: pronoun mentions are searched with PRP tag.

| | Total | | | Coarse | | | Fine | | | Ultra-Fine | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Model | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 |
| first cluster | 17.4 | 18.1 | 17.7 | 34.1 | 42.9 | 38.0 | 13.0 | 18.0 | 15.1 | 6.5 | 9.6 | 7.7 |
| random cluster (avg. of 5 runs) | 16.8 | 16.2 | 16.5 | 43.4 | 43.3 | 43.4 | 12.5 | 14.5 | 13.4 | 5.5 | 7.5 | 6.3 |
| B Li et al. (2022) | 53.3 | **56.4** | **50.6** | - | - | - | - | - | - | - | - | - |
| Dai et al. (2021) | **53.6** | 45.3 | 49.1 | - | - | - | - | - | - | - | - | - |
| Choi et al. (2018) | 47.1 | 24.2 | 32.0 | 60.3 | 63.4 | 61.8 | 41.2 | 38.7 | 39.9 | 42.2 | 9.4 | 15.4 |
| Ours | 20.1 | 19.3 | 19.7 | 42.3 | 41.9 | 42.1 | 16.1 | 20.0 | 17.8 | 8.9 | 11.0 | 9.8 |
| Ours-PRP | 25.6 | 22.0 | 23.7 | 58.7 | 53.3 | 55.9 | 23.2 | 20.0 | 21.5 | 9.5 | 12.0 | 10.6 |
| Choi et al. (2018) + Ours | 23.2 | 33.0 | 27.3 | 49.0 | 74.4 | 59.0 | 24.4 | 46.2 | 31.9 | 13.9 | 17.7 | 15.5 |
| Choi et al. (2018) + Ours (5 preds.) | 27.3 | 30.3 | 28.7 | 51.9 | 72.5 | 60.5 | 30.6 | 43.8 | 36.0 | 16.7 | 14.4 | 15.5 |
| Choi et al. (2018) + Ours-PRP | 25.2 | 33.4 | 28.7 | 54.4 | 74.5 | 62.9 | 29.9 | 46.2 | 36.3 | 14.5 | 18.4 | 16.2 |
| Choi et al. (2018) + Ours-PRP (5 preds.) | 29.4 | 30.4 | 29.9 | 56.7 | 72.8 | 63.8 | 34.3 | 44.0 | 38.5 | 17.4 | 14.7 | 15.9 |
| **without pronouns** | | | | | | | | | | | | |
| Choi et al. (2018) | 46.7 | 19.6 | 27.7 | 50.3 | 50.8 | 50.5 | 44.1 | 36.0 | 39.6 | **50.2** | 7.8 | 13.4 |
| Ours | 18.8 | 25.4 | 21.6 | 46.0 | 47.4 | 46.7 | 23.1 | 34.7 | 27.8 | 12.2 | 17.7 | 14.5 |
| Choi et al. (2018) + Ours | 21.1 | 33.4 | 25.9 | 43.0 | 68.6 | 52.8 | 25.2 | 48.7 | 33.2 | 13.8 | **21.1** | **16.7** |
| Choi et al. (2018) + Ours (5 preds.) | 26.4 | 29.6 | 27.9 | 46.4 | 65.6 | 54.3 | 31.8 | 44.9 | 37.2 | 17.7 | 16.8 | **17.3** |

We have two baselines for a comparison, which are computed over headword-prep-5050 either selecting always the first cluster or random cluster and for random cluster, we run five times and report average scores. None of the baseline is better than the method, however the first-cluster baseline performs pretty good, which suggest that the first sense is prevalent in the dataset.

### 5.4.4 Evaluation

In Table 5.5, we report P (precision), R (recall), and F1 by following recent works (Dai et al., 2021; B Li et al., 2022; Onoe et al., 2021). The scores are computed with the evaluation script provided by Choi et al. (2018)[10]. We report the results of our method on the test set by Choi et al. (2018) using the features/parameters as explained in the previous section.

We also report the results of the first/random cluster baseline (with the same possible parameters applied without the ones to choose the best cluster). The improvement over first and random cluster baselines suggests that our method is able to disambiguate the induced sense at some level. Additionally, the first cluster baseline scores are pretty good, so we can say that the first sense among the induced senses is prominent in the dataset. We perform an alternative experiment by searching pronoun mentions with PRP tag rather than NN/MWE and we can see some improvement there. Note that the coverage is changed to 80.73% in this experiment.

Most of the recent works are supervised (e.g., B Li et al. (2022) and Dai et al. (2021), in Table 5.5), and thus we cannot directly compare them with our results. For this reason, we combine our predictions with the predictions from the Choi et al. (2018) model and check if additional predictions from our approach improve the scores. We directly concatenate the predictions, and then we keep unique labels (technically we

10. https://github.com/uwnlp/open_type/blob/master/scorer.py

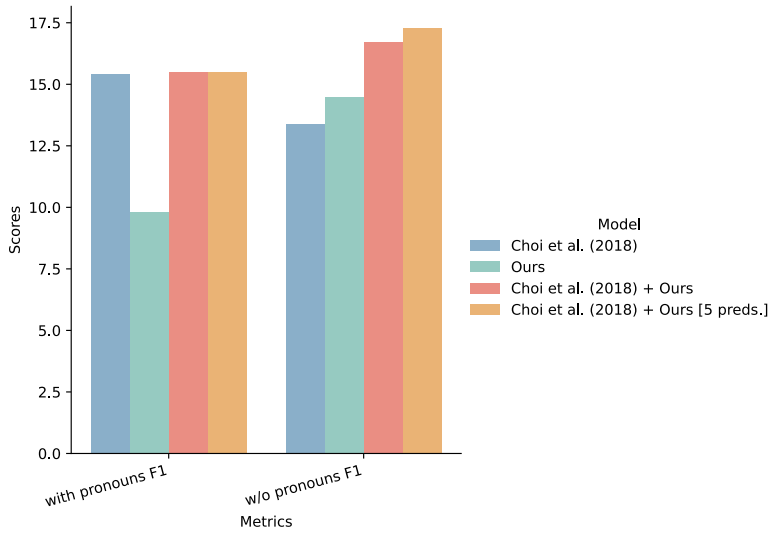


**Figure 5.3: Results on ultra-fine granularity** are shown.

make the concatenation set). They release their best model and the prediction file from this model[11], and for this experiment, we use only this prediction file.

Our solution cannot produce good hypernyms for pronouns. Therefore, we also compare the predictions for explicit mentions only, excluding pronouns. We consider pronouns: "i, me, myself, we, us, ourselves, he, him, himself, she, her, herself, it, itself, they, them, themselves, you, yourself" (and upper case of them), with references[12]. Additionally, we collect our first five predictions, for each mention, and the first five predictions from the model by Choi et al. (2018).

Based on the combination results shown in Table 5.5, the ultra-fine F1 scores are improved when the predictions of explicit mentions are combined in both cases all predictions and five predictions, which suggests our labels are complementary to the predictions by Choi et al. (2018), in this set-up, as also shown in Figure 5.3. Overall, we experiment with all granularities but only ultra-fine worked well for the tested dataset, which is a potential limitation of the approach.

We also take the first 1, 3, 5, 7 prediction(s) as the final prediction(s) and see the decrease of precision and the increase of recall, when the number of predictions are increased from 1 to 7, as shown in Figure 5.4.

## 5.5 Error Analysis and Limitations

### 5.5.1 Error Analysis

We conduct an error analysis on 100 random samples in the test set shown in Table 5.6, from our predictions explained in Section 5.4.4. Based on this analysis, we classify errors into five categories:

1. Context-dependent or pronoun mentions, where the induced senses of JoBimText are not that useful for pronoun mentions, and mentions, where the labels are

11. http://nlp.cs.washington.edu/entity_type/model/best_model.tar.gz
12. https://github.com/HKUST-KnowComp/MLMET/blob/main/prep.py#L9, https://en.wikipedia.org/wiki/English_pronouns#Full_list

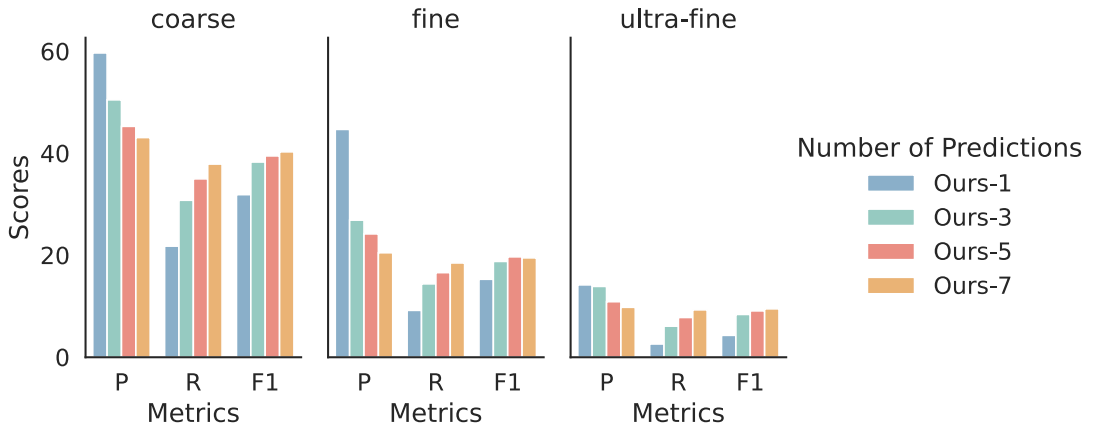


**Figure 5.4: Our results with different number of predictions on test set** are shown. Ours-{1,3,5,7}: the results contain the first {1,3,5,7} prediction(s).

**Table 5.6: A sample per category**: Mentions are marked as red, the predictions, and gold labels are examplified for each category. 1: Context-dependent or pronoun mentions, 2: JoBimText does not contain the referred sense information, 3: The labels are not the matching perfectly, 4: Some preprocessing issues, 5: The JoBimText labels are not that relevant, 6: Wrong sense selection.

| Cat. | Context | Search Mention | Predictions | True Labels |
|---|---|---|---|---|
| 1 | "They need to allow the international humanitarian organizations full and unobstructed access because they are obstructing access right now , " -Sollom- said . | Sollom | no information (assigned person) | person |
| 2 | Following a full-scale tour in support of its previous album , -Binaural- -LRB- 2000 -RRB- , Pearl Jam took a year-long break . | Binaural | company, format, feature, technology, brand, variety, mode, track, stuff, film | object, album |
| 3 | " People are getting deported for -even minor offenses- like not having an ID or a driver 's license , " said Cesar Espinosa of America for All , a group that helps immigrants in Houston . | offense | offense, **crime**, activity, charge, incident, matter, case, law, act, **violation** | **violation**, difficulty, wrongdoing, error, consequence, problem, trouble, event, **crime** |
| 4 | The devastating 21 September earthquake of 1999 left Taiwan 's landscape covered in scars , but researchers discovered that -the places least damaged by the quake- were areas of natural forest . | damaged | no information (assigned person) | city, area, location, town, region, space |
| 5 | Jack Byrne , chairman of Fireman 's Fund , said this disaster will test the catastrophe reinsurance market , causing -these rates- to soar . | rate | disease, illness, condition, information, cancer, side_effect, event, number, effect, rate | share, value, capital, price, stock |
| 6 | It was formerly located at Six Flags -New Orleans before it was relocated to Six Flags Fiesta Texas- and rethemed to Goliath . | Orleans | venue, stadium, attraction, **place** | town, city, placement, space, state, location, **place**, park |

expected to be generated context-dependently, e.g., for a given name as shown in Table 5.6.

2. JoBimText does not contain the referred sense information, although it can provide many different and fine-grained induced senses. For the sample mention "Binaural", most senses are related to Binaural beats or headsets. Note that there is one sense (sense 14), which contains person names from the group Pearl Jam as terms, however there is no IS-As, and thus we do not take into account.

3. The labels are not matching perfectly.

4. Some pre-processing issues, which are discussed in detail, in limitations paragraph below. For the sample data in Table 5.6, if "place" is searched, more relevant information can be collected.

5. The labels of JoBimText are not that relevant, though the correct sense (or one of the correct senses) is selected.

6. Wrong sense selection, for example, for "Orleans", induced sense 0 is the right sense in JoBimText, however, the method selects another cluster. Note that some samples included in one group of a category can also be included in another category, e.g., some pronouns (category-1) do not have the referred sense information from JoBimText (category-2). Note also that some labels can be mixed for the sense.

### 5.5.2 Limitations

One of the goals of ultra-fine entity typing is to generate context-dependent labels for a mention, e.g., the label for "Leonardo DiCaprio" could be `passenger` depending on its context (Dai et al., 2021). JoBimText may not produce context-dependent labels, as discussed. In UFET, mention types can be nominals, named entities, and pronouns, and for pronouns, JoBimText is unable to produce good labels and clusters. There are some mistakes or limitations specifically due to the pre-processing steps. Since head word extraction returns only one token, named entities cannot be taken properly, e.g., for the mention "Los Angeles", the head word is "Los". Sometimes, the head word loses the main information, e.g. for the mention "Perhaps the biggest of those factors", the head word is "biggest", although "factors" might be a better token for this mention. There are also some limitations due to the singularization step. Sometimes it can singularize the name entities, for example for the mention "the Cleveland Browns" with "Browns" head word, after singularization the word becomes "Brown". If the plural word is not in the last token, the singularization might fail for compounds. As explained earlier, we double-check whether the token is plural using features of stanza, however, sometimes this check causes a mistake. For instance, "works" is labeled as a verb by stanza, and so it is not singularized. The induced senses might be too fine-grained for some terms, e.g., "plan" has 14 senses (with cluster 200,200).

In some cases, the labels might be mixed and produce noise information. Therefore, the predictions are far from being usable directly in real-world and the misuse of the predictions might result in wrong information, as also discussed in Ding et al. (2022).

## 5.6 Conclusion

In this study, we generate ultra-fine entity type labels using the JoBimText framework in an unsupervised way. We observe a slight improvement when we combine our predictions with the predictions from Choi et al. (2018) for the mentions that are not pronouns, and this suggests that the labels produced through JoBimText contain helpful information. The improvement is due to the drop of the precision in favor of recall. That means, JoBimText has good lexical coverage with numerous labels, but they are also noisy.

## 5.7 Future Work

There are several promising further directions, such as, we consider an unsupervised solution in this work, yet the produced labels can be used as a weak label of some supervised models as in some previous models. Our produced unsupervised labels might help when supervised labels are not sufficient. We try to find good features and parameters based on a manual search, however, further improvement over the search space by better tuning parameters seems possible.

In this chapter, to address the data scarcity issue in ultra-fine entity typing, we generate type labels automatically, depending on graph-based information through JoBimText. We are proceeding in the next chapter with a study on entity disambiguation by leveraging the graph information, again. In this time, graph information comes from knowledge graph rather than JoBimText.

# 6

# Supervised Entity Disambiguation with Graph Embeddings

**Methods have mostly focused on unstructured data to learn entity representations, however, there is structured information in the KG itself that should be useful for the disambiguation, as discussed in Chapter 3. We use graph embeddings for integrating structured information from the KG with unstructured information from text-based representations. Our experiments confirm that graph embeddings are helpful for simple feedforward model and recent neural model of entity disambiguation. The content of this chapter' version was published as (Sevgili et al., 2019), edited to fit in the thesis, e.g. some contents are moved to some other chapter, or excluded, added new ones, corrected language issues, etc.**

## Contents

## 6.1 Introduction

The inherent and omnipresent ambiguity of language at the lexical level results in ambiguity of words, named entities, and other lexical units. Word sense disambiguation (WSD) (Navigli, 2009) deals with individual ambiguous words such as nouns, verbs, and adjectives. As discussed in the Chapter 3, the task of entity linking (EL) is devoted to the disambiguation of mentions of entities such as persons, locations, and organizations. Briefly, EL aims to resolve such ambiguity by creating an automatic reference between an ambiguous entity mention/span in a context and an entity in a knowledge graph, with two subtasks mention detection (or named entity recognition) to find entity references in the text and entity disambiguation to assign the entities (we refer the reader to Chapter 3 for more information). This work focuses on the entity disambiguation task.

The goal of an entity disambiguation task is resolving the ambiguity of entity mentions, such as *Mars, Galaxy, and Bounty are all delicious.* It is hard for an algorithm to identify whether the entity is an astronomical structure[1] or a brand of milk chocolate[2].

In Chapter 3, the general architecture of typical neural EL methods is provided, in which one of the key components is an entity representation. There are three common ways for encoding entities, i.e. using unstructured texts and algorithms like word2vec, using relations between entities in KGs, or training full-fledged neural encoder through entity-specific information (see Section 3.3.1.3). In this study, we aim to create entity embeddings through relational information based on structured data (i.e. links) using graph embeddings, integrate them into the ED models, and compare their impact with respect to text based entity embeddings.

Graph embeddings aim at representing nodes in a graph, or subgraph structure, by finding a mapping between a graph structure and the points in a low-dimensional vector space (Hamilton et al., 2017) (see Section 2.3.3 for further information).

We believe that including graph structure features of the knowledge base via graph embeddings can have a potential to make a positive impact on entity disambiguation. For this, we implement two experiments, in this chapter. In our first experiment, we have a simple neural network with the inputs of context vector, entity mention/span vector, explanation vector of a candidate entity, and graph-based vector of candidate entity. Graph-based entity representations are computed by graph embeddings, which are created using the knowledge graph, DBpedia (Lehmann et al., 2015) containing links between entities. We perform ablation tests on the types of inputs, which allow us to judge the impact on the single inputs as well as their interplay, especially on text-based entity representations from explanation vs. graph-based ones. In the second experiment, we utilize a neural entity linking/disambiguation model by Kolitsas et al. (2018), in which entity representations (Ganea and Hofmann, 2017) depend on entity description pages and the text surrounding entity mentions. We replace their entity representations with our graph embeddings, as well as combine them together. Both experiments confirm that structured information in the form of graph embeddings are an efficient and effective way of helping disambiguation.

1. http://dbpedia.org/resource/Galaxy
2. http://dbpedia.org/resource/Galaxy_(chocolate)

## 6.2 Related Work

**Entity Linking/Disambiguation** The works in (neural) entity linking are broadly analyzed in the Chapter 3. This study is more related to entity encoder methods, which are summarized in Section 3.3.1.3, and entity encoder source information per model can be found in Table 3.2. There are many approaches that depends on the relational information in knowledge graphs, which are also relevant here. The main contribution in this chapter is a simple technique to integrate the structured information through graph embeddings into neural entity disambiguation model and compare the graph embeddings with the unstructured text based entity representations.

Our experiments are conducted with two models, a simple neural network and the model by Kolitsas et al. (2018), which is an end-to-end system addressing both mention detection and entity disambiguation, jointly. They also allow to run disambiguation alone, which we utilize in our experiments. Their architecture consists of many components, e.g. entity embeddings, mention embeddings, context-aware word embeddings based on character and word embeddings, etc. Their entity embeddings, proposed by Ganea and Hofmann (2017), are computed through co-occurrence counts on Wikipedia pages and fixed-size text around entity mentions in annotated text, via Wikipedia hyperlinks.

**Graph Embeddings** There are various methods to compute graph embeddings, and we refer the reader to Section 2.3.3, for more information. In the scope of this study, to keep it simple, and efficient, we leverage DeepWalk (Perozzi et al., 2014) that uses random walks to learn latent representations and provides a representation of each node on the basis of the graph structure.

## 6.3 Learning Graph-based Entity Vectors

We construct a graph, where the nodes are entities and edges are page links between entities through the information in DBpedia. Yet, there are different number of entities in different files, in DBpedia[3]. Therefore, we leverage the intersection entities of several files, i.e., long abstract[4], labels[5], and page links[6].

Then, a vector representation per entity is computed through DeepWalk on the edges of this graph. For this, we use all default hyper-parameters of DeepWalk, e.g. *number-walks* is ten, *walk-length* is 40, and *window-size* is five. To exemplify the result, the most similar three entities of disambiguated versions of `Michael_Jordan`, in the trained model with 400-dimensional vectors are shown in Table 6.1. The first entity, `Michael_Jordan`, is a well-known basketball player[7], and his all most similar entities are all basketball players of similar age. The second entity, `Michael_I._Jordan` is a scientist[8], and again the most similar entities are either scientists in the same field or the topics of his study field. The last entity, `Michael_Jordan_(footballer)`, is a football

3. https://downloads.dbpedia.org/wiki-archive/downloads-2016-10.html
4. full abstracts of Wikipedia articles, which are usually the first section
5. titles of Wikipedia articles
6. internal links between Wikipedia articles
7. http://dbpedia.org/resource/Michael_Jordan
8. http://dbpedia.org/resource/Michael_I._Jordan

| Entity | Most similar 3 entities |
|---|---|
| Michael_Jordan | Charles_Barkley, Scottie_Pippen, Larry_Bird |
| Michael_I._Jordan | David_Blei, Machine_learning , Supervised_learning |
| Michael_Jordan_(footballer) | Dagenham_&_Redbridge_F.C., Stevenage_F.C., Yeovil_Town_F.C. |

**Table 6.1: Graph entity embeddings:** Top three most similar entities for the name "Michael Jordan" based on our 400-dimensional DeepWalk embeddings. The first entity refers to a basketball player, the second one is for a scientist, and the last entity is a football player as name refers.

player[9] and his most similar entities are football clubs. This suggests that our graph entity embeddings can differentiate different entities with the same name.

## 6.4 Experiment 1: Entity Disambiguation with Text and Graph Embeddings

In our first experiment, we build a simple neural entity disambiguation model based on a feedforward network and test the utility of the graph embeddings as compared to text-based embeddings.

### 6.4.1 Description of the Neural Entity Disambiguation Model

The inputs of an entity disambiguation task are a context and a possibly ambiguous entity span, and the output is a knowledge base entry. For example, *Desire contains a duet with Harris in the song Joey* and *Desire* given as an input and the output is *Bob Dylan's album* entity[10].

Our model in this experiment is a feedforward neural network. Its input is a concatenation of document vectors of a context, a span, and an explanation of the candidate entity, i.e. long abstract, and graph embedding of a candidate entity as in Figure 6.1, and output is a prediction value denoting whether the candidate entity is correct in this context. For learning representations, we employ doc2vec (Le and Mikolov, 2014) for text (see Section 2.3.1 for more information about doc2vec) and DeepWalk (Perozzi et al., 2014) for graphs. We will describe the input components in more detail in the following.

**Creating Negative Samples:** It is not computationally efficient to use all entities in our graph as a candidate for every context-span as negative examples for training because

9. http://dbpedia.org/resource/Michael_Jordan_(footballer)
10. http://dbpedia.org/page/Desire_(Bob_Dylan_album)

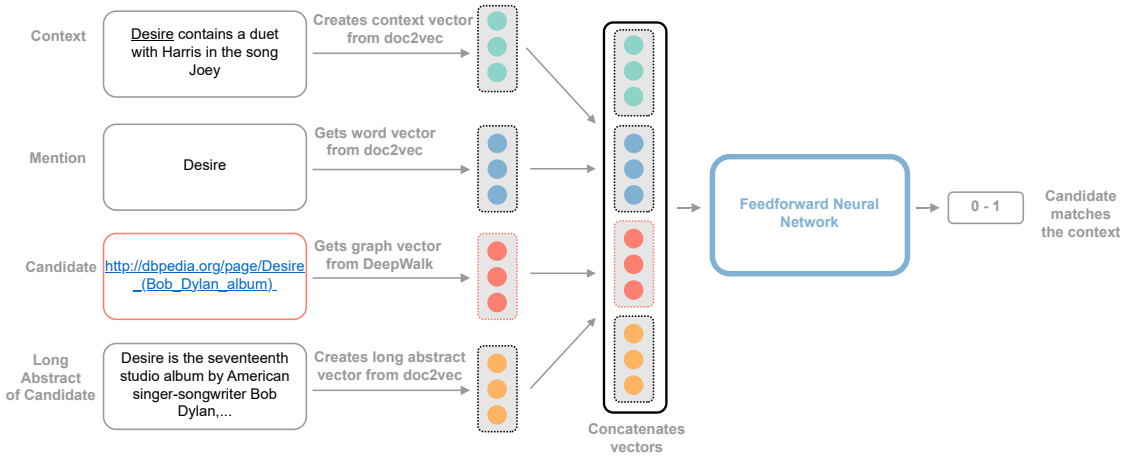


**Figure 6.1: Architecture of our feedforward neural entity disambiguation model**: using graph embeddings from internal links of Wikipedia articles as an additional input representation of entities.

of the high number of entities. Thus, we need to filter some possible entities for each context-span in order to generate negative samples. We use spans to find out possible entities. If any lemma in the span is contained in an entity's name, the entity is added to the candidates for this mention. For example, if the span is *undergraduates*, the entity Undergraduate_degree is added to the candidates.

For training, we generate negative samples by filtering this candidate list and limited the number of candidates per positive sample. Note for some mentions, no negative samples are generated, in this cases, completely random samples are used. We employ two techniques to filter the candidate list. First, we shuffle the candidate list and randomly select $n$ candidates. The other is to select the closest candidates by the following score formula: $score = \frac{\#\ of\ intersection \times page\ rank}{length}$, where $\#\ of\ intersection$ means the number of the common words between candidate entity and span/entity mention along with the gold entity, *page rank* is the page rank value (Page et al., 1999) on the entire graph for the candidate entity to include a popularity prior, and length is the number of tokens in the entity's name/title, e.g. the length of the entity Undergraduate_degree is 2. Before taking candidates with highest $n$ scores, we have pruned the most similar candidates to the correct entity on the basis of the cosine between their respective graph embeddings. The reason for pruning is to assure that the entities are distinctive enough from each other so that a classifier can learn the distinction.

**Word and Context Vectors:** Document embedding techniques like doc2vec (Le and Mikolov, 2014) assign a single vector to each document, which gets adjusted with respect to all words in the document and all document vectors in the dataset. Additionally, doc2vec provides the *infer_vector* method, which takes a word sequence and returns its representation. We employ this function for representing contexts (including the entity span), entity explanations (long abstracts), and multi-word spans.

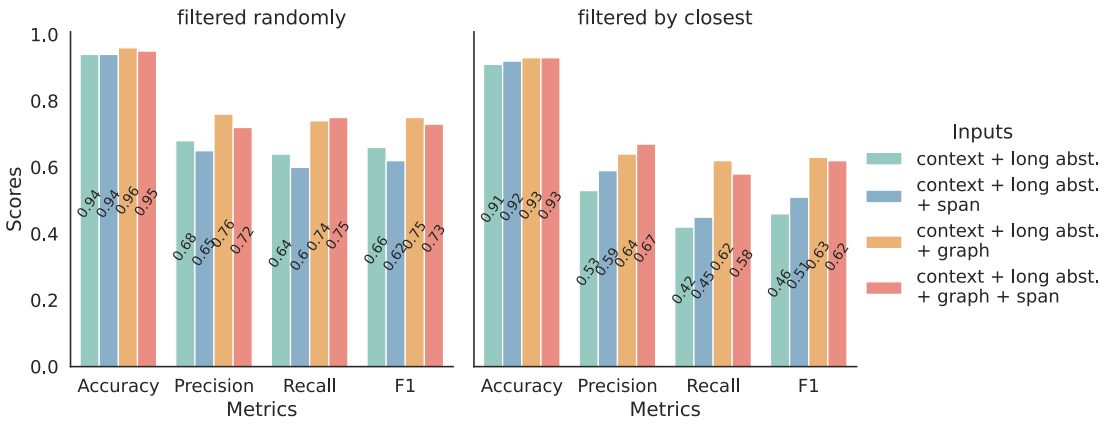


**Figure 6.2: Entity disambiguation performance**: different input configurations of our neural feedforward entity disambiguation model (cf. Figure 6.1). Reported are scores on the positive class filtered randomly and closest neighbors.

### 6.4.2 Experimental Setup

**Datasets**: An English Wikipedia 2017 dump has been used to train doc2vec, using the gensim implementation (Řehůřek and Sojka, 2010). There are about 5 million entities (nodes), and around 112 million page links (edges), in our graph.

*DBpedia Spotlight* (Mendes et al., 2011) (331 entities), *KORE50* (Hoffart et al., 2012) (144 entities), and *Reuters-128* (Röder et al., 2014) (881 entities) datasets are used to train and test our architecture (numbers are by Rosales-Méndez et al. (2018), see this work for more information about the datasets). It is worth noting that not all gold reference entities are contained in our graph, we ignore these[11]. We have used 80% of these data for training, 10% for development, and the remaining for testing.

**Implementation Details**: We fixed context, span, and long abstract embedding dimensionality to 100, the default parameter defined in the implementation of gensim (Řehůřek and Sojka, 2010). The size of the graph embeddings is 400. We optimize the graph embedding size based on the development set with the range 100-400. The overall input size is 700 when concatenating context, span, long abstract, and graph entity embeddings.

The number of negative samples per positive sample is ten. We have three hidden layers with equal sizes of 100. In the last layer, we have applied the *tanh* activation function. We have used *Adam* (Kingma and Ba, 2014) optimizer with a learning rate of 0.005 and 15000 epochs. All hyper-parameters are determined by preliminary experiments.

### 6.4.3 Evaluation

The evaluation shows the impact of graph embeddings in a rather simple learning architecture.

In this experiment, an ablation test is performed to analyze the effect of graph embeddings. We have two types of training sets, where the creation of negative samples differs (in one of them, we have filtered negative samples randomly, whereas, in the

11. After keeping unique instances, we have 990 samples with context and gold references, which are in our graph. In total, 10890 samples with 10 negative samples per positive with random filtering including positives, 10238 samples with closest filtering. The latter is less as it also contains the pruning step.

other, we filtered them by selecting the closest ones, as explained in Section 6.4.1). In Figure 6.2, the left figure shows Accuracy, Precision, Recall, and F1 values on the training set filtered randomly while the right one on the training set filtered by selecting closest neighbors. The first bar in the charts contains the result of the input, which concatenates context and long abstract embeddings (in this condition the input size becomes 200), here entity information only comes from its long abstract. The second bar presents the results of the input combination, context, word/span, and long abstract embeddings (the size of the input is 300). In the third bar, the input is the concatenation of context, long abstract, and graph embeddings (the input size is 600). Finally, the last bar indicates results for the concatenation of all types of inputs, for an input size of 700. For each configuration, we run the model five times and report the mean.

Comparing the first and third bars (or the second and last bars) in Figure 6.2, we can clearly see the results are increased when the input includes the graph embeddings for both variants of negative sampling. Comparing the third and last bars (or the first and second bars), we observe that including the span representation slightly decreases most of the results for both sampling variants. We attribute this to the presence of the context embedding, which already includes the span, thus this might increase the number of parameters of the network without substantially adding new information. Appending the graph embeddings improves the results about 0.09-0.17 in F1, 0.13-0.2 in recall, 0.07-0.12 in precision and 0.01-0.02 in accuracy scores. In general, the randomly sampled dataset is easier as it contains less related candidates.

## 6.5 Experiment 2: Integrating Graph Embeddings in the end2end ED System

### 6.5.1 Description of the Neural ED Model

For the second experiment, we have used the end2end system for EL/ED (Kolitsas et al., 2018) and expanded it with our graph embeddings. In this neural end-to-end entity disambiguation system, text-based entity embeddings are used. In the experiment described in this section, we replace or combine them (keeping the remaining architecture unchanged) with our graph embeddings build as described in Section 6.3. We replaced end2end's entity vector with our graph embeddings and the concatenation of their entity vector and our graph embeddings. We use the GERBIL (Röder et al., 2018) benchmark platform for conducting the evaluation.

### 6.5.2 Experimental Setup

**Datasets:** We train the neural end2end system in its default configuration with the combination of *MSNBC* (Cucerzan, 2007) (747 entities), *ACE2004* (Ratinov et al., 2011) (306 entities), *AQUAINT* (Ratinov et al., 2011) (727 entities), *ClueWeb*, and *Wikipedia* datasets. We test the system on the GERBIL (Röder et al., 2018) platform using *DBpedia Spotlight* (Mendes et al., 2011) (331 entities) and *Reuters-128* (Röder et al., 2014) (881 entities) datasets (numbers are again by Rosales-Méndez et al. (2018)).

| DBpedia Spotlight dataset | | | | | | |
|---|---|---|---|---|---|---|
| Model | Macro F1 | Macro Precision | Macro Recall | Micro F1 | Micro Precision | Micro Recall |
| text embeddings | 0.762 | 0.790 | 0.742 | 0.781 | 0.815 | 0.750 |
| graph embeddings | 0.796 | **0.860** | 0.758 | 0.783 | **0.847** | 0.730 |
| text and graph embeddings | **0.798** | 0.835 | **0.775** | **0.797** | 0.835 | **0.763** |
| Reuters-128 dataset | | | | | | |
| Model | Macro F1 | Macro Precision | Macro Recall | Micro F1 | Micro Precision | Micro Recall |
| text embeddings | 0.593 | 0.654 | 0.575 | 0.634 | 0.687 | 0.589 |
| graph embeddings | 0.607 | **0.694** | 0.574 | **0.660** | **0.747** | 0.592 |
| text and graph embeddings | **0.614** | 0.687 | **0.590** | 0.650 | 0.707 | **0.602** |

**Table 6.2: Entity disambiguation performance**: The end2end (Kolitsas et al., 2018) system based on the original text-based embeddings, our graph embeddings and a combination of both evaluated using the GERBIL platform on *DBpedia Spotlight* and *Reuters-128* datasets.

**Implementation Details:** We have not changed hyper-parameters for training the end2end system[12] (We used their base model + global for ED setting). We create graph embeddings with the same technique used before, however, to keep everything the same, we decided to also use 300 dimensions for the graph embeddings in this experiment to match the dimensionality of end2end's space.

We create the embeddings file with the same format they used. They give an id for each entity and call it "wiki id". First, we generate a map between this wiki id and our graph id (id of our entity). Then, we replace each entity vector corresponding to the wiki id with our graph embeddings, which refers to the entity. Sometimes there is no corresponding graph entity for the entity in the end2end system, in this case, we supply a zero vector. They have a stopping condition, which applies after six consecutive evaluations with no significant improvement in the Macro F1 score. We have changed this hyperparameter to ten, accounting for our observation that the training converges slower when operating on graph embeddings.

### 6.5.3 Evaluation

Table 6.2 reports ED performance evaluated on *DBpedia Spotlight* and *Reuters-128* datasets. There are three models, end2end trained using their text entity vectors, our graph embeddings and the combination of them. Training datasets and implementation details are the same for all models. We train the models for ten times and removed the models that did not converge (one non-converging run for each single type of embedding and two for the combination). Table 6.2 shows the mean values. The standard deviations of the models are between 0.02-0.05 in the *DBpedia Spotlight* dataset and 0.01-0.03 in the *Reuters-128* dataset over all scores. Scores are produced using the GERBIL platform; these are Micro-averaged over the set of annotations in the dataset and Macro-averaged over the average performance per document. The results are improved by including

12. https://github.com/dalab/end2end_neural_el

graph embeddings. When we compare two models, trained by graph embeddings and trained by entity vectors, the results are improved up to 0.03 in Macro F1 scores and Micro Precision, and up to 0.07 in Macro Precision. However, the improvement of the combination model is higher in Macro F1 and Recall. Micro-averaged results follow a similar trend. When we look at the scores of *Reuters-128* (Röder et al., 2014) dataset, the combination model improves Macro F1 and Recall and Micro Recall up to 0.02, 0.015, and 0.013 respectively. In the Micro-averaged evaluation, the combination model scores slightly below the model using graph embeddings alone.

To summarize the evaluation, our graph embeddings alone already lead to improvements over the original text-based embeddings, and their combination is even more beneficial. This suggests that test-based and graph-based representations in fact encode somewhat complementary information.

## 6.6 Limitations

The usual neural entity disambiguation model contains the candidate generator component, and entity ranking module to decide which candidate is best and its output is the entity, as discussed in Chapter 3.

In our feedforward neural model, we keep the candidate generation step simple and we have the negative samples per a positive sample. Further, our method is a binary classifier rather than an entity ranker. Yet, our model contains core elements to distinguish entities, so it can be used as a sub-component to build the full entity disambiguation model with all components. Furthermore, our goal here is to see the impact of the graph embeddings, rather than proposing an entity disambiguation neural architecture. We experiment with doc2vec model using infer_vector method that might produce different embeddings for the same context. Recent sentence embeddings can be utilized here for more stable embeddings.

Neural networks contain some randomness, e.g. shuffling for training, development, and test samples, and so, the scores might not be the same if reproducing the approach from scratch. Yet, we expect the closer results to what we report here.

## 6.7 Conclusion

We have shown how to integrate structured information via graph embeddings into the neural entity disambiguation task using two different experiments. In the first experiment, we use a simple neural network to gauge the impact of different text-based and graph-based embeddings. In the second experiment, we replace respectively complemented the representation of candidate entities in the recent entity disambiguation and linking model. In both setups, we demonstrate that graph embeddings lead to en par or better performance.

This confirms our research hypothesis that it is possible to use structured resources for modeling entities in entity disambiguation task and the information is complementary to a text-based representation alone.

## 6.8 Future Work

For future work, we plan to examine graph embeddings on other relationships, e.g. taxonomic or otherwise typed relations such as works-for, married-with, and so on, generalizing the notion to arbitrary structured resources. Additionally, to train the models we plan to use AIDA dataset (Hoffart et al., 2011), which is one of the common choice, as discussed in Chapter 3.

In this chapter, we present our final study in the scope of this thesis. Next chapter will cover the conclusion, general limitations, and future directions.

# 7 Conclusion

**In this section, we highlight the contributions of each study, and discuss their influences, like how the natural language processing and understanding community might leverage the outputs from each study. The general limitations are discussed. Finally, we provide future promising directions.**

## Contents

## 7.1 Conclusion

This thesis contributes natural language processing and understanding field(s) of artificial intelligence by (1) providing an abstract view of neural entity linking techniques, (2) investigating a way for an automatic entity type generation without requiring a labeled training data, (3) examining an integration way of structured information into entity disambiguation through graph embeddings. The contributions of individual chapters are discussed in Section 1.4. Their respective potential impact is discussed in the following paragraphs.

**Survey of Neural Entity Linking Models**: We present a typical general architecture of neural entity linking that is applicable most of the neural models. We discuss different components of this architecture. With the help of them, one might focus on the improvement of one of the components, consider e.g. context and/or mention encoder, or entity ranking method, through recent advancements. This has a potential to improve overall linking results. For instance, in Chapter 6, we focus on entity encoder component, and study to show an impact of an embedding constructed via structured information. Furthermore, the survey work is especially helpful for the new researchers or students, when they just start studying in this task, as they can understand e.g. what the task is aimed to, what the main components are, how the general neural entity linking model looks like, what the common modifications are, etc. One can easily go over feature choices of each models with the help of Table 3.2 and look at the performance of each model in the Tables 3.5, 3.6. One can see the results of individual models with different choices, for example, in Figure 3.8, we show the different context/mention encoder choices per model, or in Figure 3.9, model's preferences on local or global information use are displayed. Additionally, we provide the links for the available source codes of the models in Table 3.3, which might be used as a starting point to check the implementation details. All in all, these tables, figures, and information might give the idea of what works better to shape the future works.

**Unsupervised Entity Typing with JoBimText**: We generate automatically ultra-fine types for entity mentions without relying on any annotated training data, yet rather depending on a graph information constructed through terms, JoBimText. In this study, our focus is on the entity mentions in the ultra-fine entity typing task, however one can leverage the method to extract the type information for other lexical units, e.g. words. Furthermore, this work suggests that the data from another form than raw text is useful, which might help to generate new future ideas to use such kind of data in different scenarios.

**Supervised Entity Disambiguation through Graph Embeddings**: We show a straightforward way to include a structured information into neural entity disambiguation through graph embeddings. We compare such representations with the text-based entity embeddings to show the impact of graph embeddings. Hence, this study shows that a structured information can be encoded via graph embeddings techniques and can be easily used as another input component in the neural architectures. One can apply this idea to a different graph and integrate easily this embedding information to a neural model for a different task. Additionally, entity embeddings constructed from knowledge bases might be leveraged in other entity analysis tasks besides the entity disambiguation and linking.

## 7.2 Limitations

We mostly already discussed the limitations of individual works, in Sections either Limitations 5.5.2, 6.6 or in Article Collection Methodology 3.1.2, 4.1.2, or in Goal and Scope 3.1.1, 4.1.1. However, there are also several general limitations that we want to discuss. In the following items, we list these limitations.

- The entity linking survey study, presented in Chapter 3, contains papers up until and including 2021, as mentioned earlier in that chapter. It has evolved until the time of writing this thesis, and there might be some changes in the general architecture due to recent technological developments. Especially, after large language models are emerged, new models might be developed using such recent models. For instance, A Xin et al. (2024) leverage LLMs to augment data for entity linking. Therefore, it might be a possible extension of our work to conduct a research on how LLMs influence neural entity linking.

  Similarly, our summary for entity typing in Chapter 4 depends on papers collected from the search done in 2023. Therefore, new models should be presented since this time. Analysis of these models would be, again, a potential extension of our study.

- Language modeling is one of the key and old tasks in natural language processing with the goal to predict next or missing tokens (WX Zhao et al., 2024). It has evolved during time starting from statistical approaches to recent models, i.e. pre-trained language models, e.g. BERT as explained in Section 2.2.4. Very recently, such pre-trained language models are scaled and the large-sized models, called large language models, have shown many remarkable abilities (WX Zhao et al., 2024). After LLMs emerged, researchers have applied it in various scenarios, leveraged and explored it in various different ways. Many articles have been published on this, recently.

  In this thesis, we have not leveraged LLMs in our conducted experiments. The reason for it is our research studies were planned earlier than LLMs became that popular. For example, we finished most of the work in our survey study around the end of 2021. Yet, we leverage PLM based embeddings in our thesis, e.g. in unsupervised ultra-fine entity typing in Chapter 5, we utilize such embeddings to select the best sense to distinguish the meaning of the entity mention, as explained in detail in this chapter.

  We still think LLMs can be integrated to the studies discussed in this thesis. For instance, LLMs might be used to filter our noisy labels for the unsupervised ultra-fine entity typing study, in Chapter 5. Alternatively, our labels can be used along with LLMs predictions to check if it is possible to prevent both hallucinations and noisy labels for the same work. Hallucination is one of the issues that LLMs face with in especially knowledge-intensive tasks (Gao et al., 2024).

## 7.3 Future Directions

We discuss some promising directions of future work related to the tasks of entity linking/disambiguation and entity typing, in the following items.

- **Joint entity linking and entity typing**: There exist several works with the aim at providing joint model for entity related tasks as discussed in Section 1.2, e.g. Durrett and Klein (2014) provide a joint model for entity linking, typing and coreference resolution. This model is proposed in 2014. Since then, many techniques have been developed. Thus, in the future, we expect more approaches on this kind of joint models with recent methodologies, as these tasks interact with each other by helping each other. Therefore, the final model might be a powerful one showing good performance in each different tasks.

- **Retrieval Augmented Generation (RAG) in entity linking or entity typing**: As discussed earlier, LLMs have shown significant success, yet still there are several issues of them, e.g. hallucination in especially knowledge-intensive tasks (Gao et al., 2024) as mentioned earlier. Retrieval augmented generation is one of the techniques to solve this issue by integrating relevant external knowledge to LLMs. We believe that RAG can also enhance entity linking. For instance, Shlyk et al. (2024) explore RAG entity linking in biomedical domain. We expect more applications of RAG in entity linking and typing tasks.

- **Graph embeddings in entity typing**: We leverage graph embeddings technique in our entity disambiguation study, discussed in Chapter 6. In general, graph embeddings transform structural information into embedding space. There exist some works that use graph embeddings in entity typing. For example, Y Zhao et al. (2020) propose an embedding model to predict the missing entity type information for knowledge graph entity typing task, which is a sub-task of knowledge graph completion. This task is different from fine- and ultra-fine entity typing that we discussed in this thesis. Yet, we still expect some future studies that explore graph embeddings in fine- and ultra-fine entity typing tasks as well.

- **Multimodal entity linking** Throughout this thesis, we have discussed entity linking with their textual properties. However, multimodal information gains interests, recently, and multimodal entity linking models are developed to solve cross-modal ambiguity, where the contexts contain not only textual information but also image information (S Shi et al., 2024). Recently, S Shi et al. (2024) propose a generative multimodal approach using LLMs. Therefore, we think in the future there would be more techniques, in this line of the challenge.

# Appendices

# A
# Parameter Search

| Model | Total | | | Coarse | | | Fine | | | Ultra-Fine | | | Coverage |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | |
| last-n1 | 18.0 | 17.8 | 17.9 | 44.7 | 49.3 | 46.9 | 12.7 | 15.4 | 13.9 | 5.5 | 7.9 | 6.5 | 77.93 |
| last-n2 | 20.3 | 16.0 | 17.9 | 41.0 | 46.9 | 43.7 | 14.5 | 12.7 | 13.5 | 5.9 | 6.1 | 6.0 | 59.06 |
| last-n3 | 21.6 | 14.5 | 17.3 | 39.5 | 45.4 | 42.2 | 13.2 | 9.9 | 11.3 | 5.4 | 4.7 | 5.0 | 50.45 |
| first-n1 | 15.5 | 15.9 | 15.7 | 42.1 | 46.2 | 44.0 | 9.8 | 13.0 | 11.2 | 4.1 | 6.4 | 5.0 | 82.13 |
| first-n2 | 20.7 | 15.0 | 17.4 | 40.7 | 45.9 | 43.1 | 13.2 | 11.0 | 12.0 | 5.2 | 5.2 | 5.2 | 58.66 |
| first-n3 | 22.0 | 14.1 | 17.2 | 39.0 | 44.9 | 41.7 | 13.1 | 9.0 | 10.7 | 5.4 | 4.6 | 5.0 | 48.05 |
| last-n1-sklearn | 17.9 | 17.8 | 17.8 | 44.5 | 49.1 | 46.7 | 12.6 | 15.4 | 13.9 | 5.5 | 7.9 | 6.5 | 78.23 |
| last-n2-sklearn | 20.4 | 16.1 | 18.0 | 41.1 | 47.0 | 43.8 | 14.5 | 12.7 | 13.5 | 5.9 | 6.2 | 6.0 | 59.01 |
| last-n3-sklearn | 21.6 | 14.4 | 17.3 | 39.4 | 45.3 | 42.1 | 13.1 | 9.8 | 11.2 | 5.4 | 4.7 | 5.0 | 50.25 |
| first-n1-sklearn | 15.4 | 15.8 | 15.6 | 42.0 | 46.0 | 43.9 | 9.7 | 12.9 | 11.1 | 4.1 | 6.4 | 5.0 | 82.73 |
| first-n2-sklearn | 20.7 | 15.1 | 17.5 | 40.8 | 45.9 | 43.2 | 13.2 | 11.1 | 12.0 | 5.4 | 5.4 | 5.4 | 58.66 |
| first-n3-sklearn | 22.1 | 14.1 | 17.2 | 39.0 | 44.9 | 41.7 | 13.0 | 8.9 | 10.6 | 5.4 | 4.5 | 4.9 | 47.90 |

**Table A.1**: Unsupervised ultra-fine entity typing performance: Results are on UFET development set to compare sklearn vs. nltk libraries for n-gram extraction in this task. The results are collected for the n-grams from using either sklearn or nltk library. “last-n*”: the last *-gram of mention is searched on JoBimText using nltk. “first-n*”: the first *-gram of mention is searched on JoBimText using nltk. “last-n*-sklearn”: the last *-gram of mention is searched on JoBimText using sklearn. “first-n*-sklearn”: the first *-gram of mention is searched on JoBimText using sklearn.

# References


Rami Aly, Andreas Vlachos, and Ryan McDonald. 2021. Leveraging Type Descriptions for Zero-shot Named Entity Recognition and Classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers),* 1516–1528. Online: Association for Computational Linguistics. (Cited on pages 30, 33, 61).

Saba Anwar, Artem Shelmanov, Alexander Panchenko, and Chris Biemann. 2020. Generating Lexical Representations of Frames using Lexical Substitution. In *Proceedings of the Probability and Meaning Conference (PaM 2020),* 95–103. Gothenburg, Sweden: Association for Computational Linguistics. (Cited on page 73).

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015.* San-Diego, CA, USA. (Cited on pages 18, 20).

Daniel Bailey, Yuliya Lierler, and Benjamin Susman. 2015. Prepositional Phrase Attachment Problem Revisited: how Verbnet can Help. In *Proceedings of the 11th International Conference on Computational Semantics,* 12–22. London, UK: Association for Computational Linguistics. (Cited on page 3).

Krisztian Balog. 2018. Entity Linking in Entity-Oriented Search, 147–188. Cham: Springer International Publishing. (Cited on page 3).

Debayan Banerjee, Debanjan Chaudhuri, Mohnish Dubey, and Jens Lehmann. 2020. PNEL: Pointer Network Based End-To-End Entity Linking over Knowledge Graphs. In *The Semantic Web – ISWC 2020 - 19th International Semantic Web Conference, Proceedings, Part I,* 12506:21–38. Athens, Greece: Springer International Publishing. (Cited on pages 24, 40, 47, 50).

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (8): 1798–1828. (Cited on page 21).

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research* 3:1137–1155. (Cited on page 29).

G. P. Shrivatsa Bhargav, Dinesh Khandelwal, Saswati Dana, Dinesh Garg, Pavan Kapanipathi, Salim Roukos, Alexander Gray, and L. Venkata Subramaniam. 2022. Zero-shot Entity Linking with Less Data. In *Findings of the Association for Computational Linguistics: NAACL 2022,* 1681–1697. Seattle, WA, USA: Association for Computational Linguistics. (Cited on page 5).

Chris Biemann. 2006. Chinese Whispers - an Efficient Graph Clustering Algorithm and its Application to Natural Language Processing Problems. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing,* 73–80. New York City, NY, USA: Association for Computational Linguistics. (Cited on page 25).

Chris Biemann, Bonaventura Coppola, Michael R. Glass, Alfio Gliozzo, Matthew Hatem, and Martin Riedl. 2013. JoBimText Visualizer: A Graph-based Approach to Contextualizing Distributional Similarity. In *Proceedings of TextGraphs-8 Graph-based Methods for Natural Language Processing,* 6–10. Seattle, WA, USA: Association for Computational Linguistics. (Cited on page 25).

Chris Biemann and Martin Riedl. 2013. Text: now in 2D! A framework for lexical expansion with contextual similarity. *Journal of Language Modelling* 1 (1): 55–95. (Cited on pages 8, 10, 25 sq., 72, 74).

Steven Bird, Ewan Klein, and Edward Loper. 2009. Natural Language Processing with Python. O'Reilly Media Inc. (Cited on page 77).

Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Research* 32 (suppl_1): D267–D270. (Cited on page 39).

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics* 5:135–146. (Cited on page 48).

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data,* 1247–1250. SIGMOD '08. Vancouver, Canada: Association for Computing Machinery. (Cited on pages 23 sq., 34).

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in neural information processing systems,* 26:2787–2795. Stateline, NV, USA. (Cited on pages 23, 40).

Jan A. Botha, Zifei Shan, and Daniel Gillick. 2020. Entity Linking in 100 Languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 7833–7845. Online: Association for Computational Linguistics. (Cited on pages 47, 50).

Samuel Broscheit. 2019. Investigating Entity Knowledge in BERT with Simple Neural End-To-End Entity Linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL),* 677–685. Hong Kong, China: Association for Computational Linguistics. (Cited on pages 47, 49 sq.).

HongYun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Transactions on Knowledge & Data Engineering* (Los Alamitos, CA, USA) 30 (09): 1616–1637. (Cited on page 24).

Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural Collective Entity Linking. In *Proceedings of the 27th International Conference on Computational Linguistics,* 675–686. Santa Fe, NM, USA: Association for Computational Linguistics. (Cited on pages 44, 46, 50, 53).

Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge Text and Knowledge by Learning Multi-Prototype Entity Mention Embedding. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 1623–1633. Vancouver, Canada: Association for Computational Linguistics. (Cited on pages 37, 40, 45 sq., 50, 53, 56).

Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. 2013. Learning Relatedness Measures for Entity Linking. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management,* 139–148. CIKM '13. San Francisco, CA, USA: Association for Computing Machinery. (Cited on page 56).

Angel Chang, Valentin I. Spitkovsky, Christopher D. Manning, and Eneko Agirre. 2016. A comparison of Named-Entity Disambiguation and Word Sense Disambiguation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16),* 860–867. Portorož, Slovenia: European Language Resources Association (ELRA). (Cited on page 35).

Bo Chen, Xiaotao Gu, Yufeng Hu, Siliang Tang, Guoping Hu, Yueting Zhuang, and Xiang Ren. 2019. Improving Distantly-supervised Entity Typing with Compact Latent Space Clustering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* 2862–2872. Minneapolis, Minnesota: Association for Computational Linguistics. (Cited on page 66).

Danqi Chen and Christopher Manning. 2014. A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 740–750. Doha, Qatar: Association for Computational Linguistics. (Cited on page 3).

Haotian Chen, Xi Li, Andrej Zukov Gregoric, and Sahil Wadhwa. 2020. Contextualized End-to-End Neural Entity Linking. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing,* 637–642. Suzhou, China: Association for Computational Linguistics. (Cited on pages 47, 50).

Mingda Chen, Zewei Chu, Yang Chen, Karl Stratos, and Kevin Gimpel. 2019. EntEval: A Holistic Evaluation Benchmark for Entity Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),* 421–433. Hong Kong, China: Association for Computational Linguistics. (Cited on page 68).

Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020. Improving Entity Linking by Modeling Latent Entity Type Information. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 7529–7537. New York Hilton Midtown, NY, USA. (Cited on pages 5, 47, 50, 53).

Tongfei Chen, Yunmo Chen, and Benjamin Van Durme. 2020. Hierarchical Entity Typing via Multi-level Learning to Rank. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics,* 8465–8475. Online: Association for Computational Linguistics. (Cited on page 68).

Yi Chen, Jiayang Cheng, Haiyun Jiang, Lemao Liu, Haisong Zhang, Shuming Shi, and Ruifeng Xu. 2022. Learning from Sibling Mentions with Scalable Graph Inference in Fine-Grained Entity Typing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 2076–2087. Dublin, Ireland: Association for Computational Linguistics. (Cited on page 68).

Yi Chen, Haiyun Jiang, Lemao Liu, Shuming Shi, Chuang Fan, Min Yang, and Ruifeng Xu. 2021. An Empirical Study on Multiple Information Sources for Zero-Shot Fine-Grained Entity Typing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing,* 2668–2678. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. (Cited on page 67).

Xiao Cheng and Dan Roth. 2013. Relational Inference for Wikification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing,* 1787–1796. Seattle, WA, USA: Association for Computational Linguistics. (Cited on page 33).

Andrew Chisholm and Ben Hachey. 2015. Entity Disambiguation with Web Links. *Transactions of the Association for Computational Linguistics* 3:145–156. (Cited on pages 30, 52 sq.).

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-Fine Entity Typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 87–96. Melbourne, Australia: Association for Computational Linguistics. (Cited on pages 4 sq., 7, 30, 61 sqq., 68 sq., 72, 74 sqq., 81 sq., 84).

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning.* Montréal, Canada. (Cited on page 48).

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research* 12 (76): 2493–2537. (Cited on page 29).

Silviu Cucerzan. 2007. 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),* 708–716. Prague, Czech Republic: Association for Computational Linguistics. (Cited on pages 51, 93).

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised Sequence Learning. In *Advances in Neural Information Processing Systems,* vol. 28. Curran Associates, Inc. (Cited on page 20).

Hongliang Dai, Donghong Du, Xin Li, and Yangqiu Song. 2019. Improving Fine-grained Entity Typing with Entity Linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),* 6210–6215. Hong Kong, China: Association for Computational Linguistics. (Cited on page 5).

Hongliang Dai, Yangqiu Song, and Haixun Wang. 2021. Ultra-Fine Entity Typing with Weak Supervision from a Masked Language Model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers),* 1790–1799. Online: Association for Computational Linguistics. (Cited on pages 7, 64 sq., 72 sq., 75 sq., 81, 84).

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017. Question Answering on Knowledge Bases and Text using Universal Schema and Memory Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers),* 358–365. Vancouver, Canada: Association for Computational Linguistics. (Cited on page 61).

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In *9th International Conference on Learning Representations, ICLR 2021.* Virtual Event, Austria: OpenReview.net. (Cited on pages 47, 50, 53, 58).

Danilo Dessì, Francesco Osborne, Diego Reforgiato Recupero, Davide Buscaldi, and Enrico Motta. 2021. Generating knowledge graphs by employing Natural Language Processing and Machine Learning techniques within the scholarly domain. *Future Generation Computer Systems* 116:253–264. (Cited on page 34).

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 1811–1818. New Orleans, LA, USA. (Cited on page 24).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* 4171–4186. Minneapolis, MN, USA: Association for Computational Linguistics. (Cited on pages 6, 20, 41, 48).

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Xiaobin Wang, Pengjun Xie, Haitao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2022. Prompt-learning for Fine-grained Entity Typing. In *Findings of the Association for Computational Linguistics: EMNLP 2022,* 6888–6901. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. (Cited on pages 65, 67, 73, 84).

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research* 12 (61): 2121–2159. (Cited on page 16).

Greg Durrett and Dan Klein. 2014. A Joint Model for Entity Analysis: Coreference, Typing, and Linking. *Transactions of the Association for Computational Linguistics* 2:477–490. (Cited on pages 5, 101).

Jacob Eisenstein. 2018. Natural Language Processing. MIT Press. (Cited on page 6).

Cheikh Brahim El Vaigh, François Goasdoué, Guillaume Gravier, and Pascale Sébillot. 2019. Using Knowledge Base Semantics in Context-Aware Entity Linking. In *Proceedings of the ACM Symposium on Document Engineering 2019.* DocEng '19. Berlin, Germany: Association for Computing Machinery. (Cited on page 56).

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie M. Strassel. 2015. Overview of Linguistic Resources for the TAC KBP 2015 Evaluations: Methodologies and Results. In *Proceedings of the 2015 Text Analysis Conference, TAC 2015.* Gaithersburg, MD, USA: NIST. (Cited on page 51).

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named Entity Disambiguation for Noisy Text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017),* 58–68. Vancouver, Canada: Association for Computational Linguistics. (Cited on pages 46, 50, 53).

Wei Fang, Jianwen Zhang, Dilin Wang, Zheng Chen, and Ming Li. 2016. Entity Disambiguation by Knowledge and Text Jointly Embedding. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning,* 260–269. Berlin, Germany: Association for Computational Linguistics. (Cited on pages 40, 45 sq., 50, 53).

Zheng Fang, Yanan Cao, Qian Li, Dongjie Zhang, Zhenyu Zhang, and Yanbing Liu. 2019. Joint Entity Linking with Deep Reinforcement Learning. In *The World Wide Web Conference,* 438–447. WWW '19. San Francisco, CA, USA: Association for Computing Machinery. (Cited on pages 36, 44, 47, 50, 53).

Zheng Fang, Yanan Cao, Ren Li, Zhenyu Zhang, Yanbing Liu, and Shi Wang. 2020. High Quality Candidate Generation and Sequential Graph Attention Network for Entity Linking. In *Proceedings of The Web Conference 2020,* 640–650. WWW '20. Taipei, Taiwan: Association for Computing Machinery. (Cited on pages 38, 45, 47, 50, 53).

Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. 2018. Linked Data Quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web* 9 (1): 77–129. (Cited on pages 24, 32).

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as Experts: Sparse Memory Access with Entity Supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 4937–4951. Online: Association for Computational Linguistics. (Cited on page 6).

Matthew Francis-Landau, Greg Durrett, and Dan Klein. 2016. Capturing Semantic Similarity for Entity Linking with Convolutional Neural Networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 1256–1261. San Diego, CA, USA: Association for Computational Linguistics. (Cited on pages 40, 45 sq., 50, 53, 55).

Xingyu Fu, Weijia Shi, Xiaodong Yu, Zian Zhao, and Dan Roth. 2020. Design Challenges in Low-resource Cross-lingual Entity Linking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 6418–6432. Online: Association for Computational Linguistics. (Cited on pages 47, 50).

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. FACC1: Freebase annotation of ClueWeb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0). (Cited on page 51).

Octavian-Eugen Ganea, Marina Ganea, Aurelien Lucchi, Carsten Eickhoff, and Thomas Hofmann. 2016. Probabilistic Bag-Of-Hyperlinks Model for Entity Linking. In *Proceedings of the 25th International Conference on World Wide Web,* 927–938. WWW '16. Montréal, Canada: International World Wide Web Conferences Steering Committee. (Cited on pages 43 sq., 51 sqq.).

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep Joint Entity Disambiguation with Local Neural Attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing,* 2619–2629. Copenhagen, Denmark: Association for Computational Linguistics. (Cited on pages 30, 37 sqq., 43 sq., 46, 50 sqq., 55 sqq., 88 sq.).

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv: 2312.10997 [cs.CL]. (Cited on pages 100 sq.).

Felix A. Gers, Jürgen A. Schmidhuber, and Fred A. Cummins. 2000. Learning to Forget: Continual Prediction with LSTM. *Neural Comput.* 12 (10): 2451–2471. (Cited on page 17).

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2016. Context-Dependent Fine-Grained Entity Type Tagging. arXiv: 1412.1820 [cs.CL]. (Cited on pages 61 sqq., 68, 72).

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning Dense Representations for Entity Retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL),* 528–537. Hong Kong, China: Association for Computational Linguistics. (Cited on pages 38, 40, 47, 50, 53).

Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2016. Collective Entity Resolution with Multi-Focal Attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 621–631. Berlin, Germany: Association for Computational Linguistics. (Cited on pages 44, 46, 50, 53).

Yoav Goldberg. 2016. A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research* 57 (1): 345–420. (Cited on pages 6, 14 sqq., 21 sq.).

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. Deep Learning. Http://www.deeplearningbook.org. MIT Press. (Cited on pages 14 sqq., 20).

Archana Goyal, Vishal Gupta, and Manish Kumar. 2018. Recent Named Entity Recognition and Classification techniques: A systematic review. *Computer Science Review* 29:21–43. (Cited on page 34).

Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151:78–94. (Cited on pages 23 sq.).

Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* 855–864. KDD '16. San Francisco, CA, USA: Association for Computing Machinery. (Cited on page 24).

Yingjie Gu, Xiaoye Qu, Zhefeng Wang, Baoxing Huai, Nicholas Jing Yuan, and Xiaolin Gui. 2021. Read, Retrospect, Select: An MRC Framework to Short Text Entity Linking. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 12920–12928. held virtually. (Cited on pages 45, 47, 50).

Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. *Semantic Web* 9 (4): 459–479. (Cited on pages 43, 51, 53).

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity Linking via Joint Encoding of Types, Descriptions, and Context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing,* 2681–2690. Copenhagen, Denmark: Association for Computational Linguistics. (Cited on pages 5, 37 sq., 40 sq., 45 sq., 50, 53).

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. arXiv: 1709.05584 [cs.SI]. (Cited on page 88).

Xu Han, Yuqi Luo, Weize Chen, Zhiyuan Liu, Maosong Sun, Zhou Botong, Hao Fei, and Suncong Zheng. 2022. Cross-Lingual Contrastive Learning for Fine-Grained Entity Typing for Low-Resource Languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 2241–2250. Dublin, Ireland: Association for Computational Linguistics. (Cited on page 68).

Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics,* 539–545. Nantes, France. (Cited on pages 25, 64 sq.).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9 (8): 1735–1780. (Cited on pages 17, 48).

Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2012. KORE: Keyphrase Overlap Relatedness for Entity Disambiguation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management,* 545–554. CIKM '12. Maui, HI, USA: ACM. (Cited on page 92).

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing,* 782–792. Edinburgh, UK.: Association for Computational Linguistics. (Cited on pages 31, 33, 37 sq., 51 sqq., 56, 96).

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,* 541–550. Portland, OR, USA: Association for Computational Linguistics. (Cited on page 14).

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia D'amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. 2021. Knowledge Graphs. *ACM Computing Surveys* (New York, NY, USA), no. 4, (cited on page 34).

Feng Hou, Ruili Wang, Jun He, and Yi Zhou. 2020. Improving Entity Linking through Semantic Reinforced Entity Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics,* 6843–6848. Online: Association for Computational Linguistics. (Cited on pages 5, 40, 47, 50, 53).

Hongzhao Huang, Larry Heck, and Heng Ji. 2015. Leveraging Deep Neural Networks and Knowledge Graphs for Entity Disambiguation. arXiv: 1504.07678 [cs.CL]. (Cited on pages 40, 56 sq.).

James Y. Huang, Bangzheng Li, Jiashu Xu, and Muhao Chen. 2022. Unified Semantic Typing with Meaningful Label Inference. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 2642–2654. Seattle, WA, USA: Association for Computational Linguistics. (Cited on page 67).

Lifu Huang, Jonathan May, Xiaoman Pan, and Heng Ji. 2016. Building a Fine-Grained Entity Typing System Overnight for a New X (X = Language, Domain, Genre). arXiv: 1603.03112 [cs.CL]. (Cited on pages 62, 67, 73).

Lifu Huang, Jonathan May, Xiaoman Pan, Heng Ji, Xiang Ren, Jiawei Han, Lin Zhao, and James A. Hendler. 2017. Liberal Entity Extraction: Rapid Construction of Fine-Grained Entity Typing Systems. *Big Data* 5 (1): 19–31. (Cited on pages 62, 67, 73).

Abhik Jana and Pawan Goyal. 2018. Can Network Embedding of Distributional Thesaurus Be Combined with Word Vectors for Better Representation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers),* 463–473. New Orleans, LA, USA: Association for Computational Linguistics. (Cited on page 73).

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Transactions on Information Systems* 20 (4): 422–446. (Cited on page 56).

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the TAC 2010 knowledge base population track. In *Proceedings of the 2010 Text Analysis Conference, TAC 2010.* Gaithersburg, MD, USA. (Cited on page 51).

Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of TAC-KBP2015 Tri-lingual Entity Discovery and Linking. In *Proceedings of the 2015 Text Analysis Conference, TAC 2015,* 16–17. Gaithersburg, MD, USA: NIST. (Cited on pages 45, 51).

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Transactions on Neural Networks and Learning Systems* 33 (2): 494–514. (Cited on pages 24, 32, 34).

Chengyue Jiang, Yong Jiang, Weiqi Wu, Pengjun Xie, and Kewei Tu. 2022. Modeling Label Correlations for Ultra-Fine Entity Typing with Neural Pairwise Conditional Random Field. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing,* 6836–6847. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. (Cited on page 68).

Daniel Jurafsky and James H. Martin. 2024. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models. 3rd Edition. Online manuscript released August 20, 2024. (Cited on page 3).

Rijula Kar, Susmija Reddy, Sourangshu Bhattacharya, Anirban Dasgupta, and Soumen Chakrabarti. 2018. Task-Specific Representation Learning for Web-Scale Entity Disambiguation. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 5812–5819. New Orleans, LA, USA. (Cited on pages 34, 46).

Adam Kilgarriff, Pavel Rychlý, Pavel Smrž, and David Tugwell. 2004. The Sketch Engine. In *Proceedings of the 11th EURALEX International Congress,* 105–115. Lorient, France: Université de Bretagne-Sud, Faculté des lettres et des sciences humaines. (Cited on page 25).

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. arXiv: 1412.6980 [cs.LG]. (Cited on pages 16, 92).

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-End Neural Entity Linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning,* 519–529. Brussels, Belgium: Association for Computational Linguistics. (Cited on pages 38, 45 sq., 50, 52 sq., 88 sq., 93 sq.).

Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. 2008. Zero-data learning of new tasks. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2,* 646–651. AAAI'08. Chicago, IL, USA: AAAI Press. (Cited on page 14).

Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A Selective Context Model for Entity Resolution. *Transactions of the Association for Computational Linguistics* 3:503–515. (Cited on pages 29, 45, 52 sq.).

Phong Le and Ivan Titov. 2018. Improving Entity Linking by Modeling Latent Relations between Mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 1595–1604. Melbourne, Australia: Association for Computational Linguistics. (Cited on pages 44, 46, 50, 53).

———. 2019a. Boosting Entity Linking Performance by Leveraging Unlabeled Documents. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 1935–1945. Florence, Italy: Association for Computational Linguistics. (Cited on pages 14, 38, 44, 47, 50, 53).

———. 2019b. Distant Learning for Entity Linking with Automatic Noise Detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 4081–4090. Florence, Italy: Association for Computational Linguistics. (Cited on pages 14, 36, 40, 47, 50, 53).

Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning,* 32:1188–1196. Proceedings of Machine Learning Research 2. Bejing, China: PMLR. (Cited on pages 21 sq., 48, 90 sq.).

Chin Lee, Hongliang Dai, Yangqiu Song, and Xin Li. 2020. A Chinese Corpus for Fine-grained Entity Typing [in English]. In *Proceedings of the Twelfth Language Resources and Evaluation Conference,* 4451–4457. Marseille, France: European Language Resources Association. (Cited on page 68).

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. Doi:10.3233/SW-140134, *Semantic Web Journal* 6 (2): 167–195. (Cited on pages 9 sq., 24 sq., 34, 40, 88).

Bangzheng Li, Wenpeng Yin, and Muhao Chen. 2022. Ultra-fine Entity Typing with Indirect Supervision from Natural Language Inference. *Transactions of the Association for Computational Linguistics* 10:607–622. (Cited on pages 63, 65, 67, 76, 81).

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient One-Pass End-to-End Entity Linking for Questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 6433–6441. Online: Association for Computational Linguistics. (Cited on pages 47, 50).

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2022. A Survey on Deep Learning for Named Entity Recognition. *IEEE Transactions on Knowledge and Data Engineering* 34 (1): 50–70. (Cited on pages 6, 33 sq., 61).

Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design Challenges for Entity Linking. *Transactions of the Association for Computational Linguistics* 3:315–328. (Cited on pages 30 sq.).

Xiao Ling and Daniel S. Weld. 2012. Fine-Grained Entity Recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence,* 94–100. AAAI'12. Toronto, Canada: AAAI Press. (Cited on pages 40, 61, 63 sq., 68 sq., 72).

Lemao Liu, Haisong Zhang, Haiyun Jiang, Yangming Li, Enbo Zhao, Kun Xu, Linfeng Song, Suncong Zheng, Botong Zhou, Dick Zhu, Xiao Feng, Tao Chen, Tao Yang, Dong Yu, Feng Zhang, ZhanHui Kang, and Shuming Shi. 2021. TexSmart: A System for Enhanced Natural Language Understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations,* 1–10. Online: Association for Computational Linguistics. (Cited on pages 62, 67, 72 sq.).

Qing Liu, Hongyu Lin, Xinyan Xiao, Xianpei Han, Le Sun, and Hua Wu. 2021. Fine-grained Entity Typing via Label Reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing,* 4611–4622. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. (Cited on pages 64, 68).

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Ro{BERT}a: A Robustly Optimized {BERT} Pretraining Approach. arXiv: 1907.11692 [cs.CL]. (Cited on page 67).

Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. 2014. On the Computational Efficiency of Training Neural Networks. In *Advances in Neural Information Processing Systems,* 27:855–863. Montréal, Canada: Curran Associates, Inc. (Cited on page 48).

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-Shot Entity Linking by Reading Entity Descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 3449–3460. Florence, Italy: Association for Computational Linguistics. (Cited on pages 38, 41, 47, 50).

Federico López, Benjamin Heinzerling, and Michael Strube. 2019. Fine-Grained Entity Typing in Hyperbolic Space. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019),* 169–180. Florence, Italy: Association for Computational Linguistics. (Cited on page 68).

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label Embedding for Zero-shot Fine-grained Named Entity Typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers,* 171–180. Osaka, Japan: The COLING 2016 Organizing Committee. (Cited on page 66).

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. Introduction to Information Retrieval. USA: Cambridge University Press. (Cited on pages 56 sq.).

José L. Martínez-Rodríguez, A. Hogan, and I. López-Arévalo. 2020. Information extraction meets the Semantic Web: A survey. *Semantic Web* 11 (2): 255–335. (Cited on page 31).

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2019. Joint Learning of Named Entity Recognition and Entity Linking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop,* 190–196. Florence, Italy: Association for Computational Linguistics. (Cited on pages 33, 37, 47, 50).

Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia Spotlight: Shedding Light on the Web of Documents. In *Proceedings of the 7th International Conference on Semantic Systems,* 1–8. I-Semantics '11. Graz, Austria: Association for Computing Machinery. (Cited on pages 52 sq., 92 sq.).

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013.* Scottsdale, AZ, USA. (Cited on pages 21, 48).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2,* 3111–3119. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc. (Cited on pages 22, 38, 48).

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proc. Interspeech 2010,* 1045–1048. Makuhari, Chiba, Japan. (Cited on page 17).

David Milne and Ian H. Witten. 2008. Learning to Link with Wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management,* 509–518. CIKM '08. Napa Valley, CA, USA: Association for Computing Machinery. (Cited on pages 31, 51, 56 sq.).

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP,* 1003–1011. Suntec, Singapore: Association for Computational Linguistics. (Cited on page 14).

Cedric Möller, Jens Lehmann, and Ricardo Usbeck. 2022. Survey on English Entity Linking on Wikidata: Datasets and approaches, 6. (Cited on page 31).

Jose G. Moreno, Romaric Besançon, Romain Beaumont, Eva D'Hondt, Anne-Laure Ligozat, Sophie Rosset, Xavier Tannier, and Brigitte Grau. 2017. Combining Word and Entity Embeddings for Entity Linking. In *The Semantic Web,* 337–352. Cham: Springer International Publishing. (Cited on pages 36, 39, 45 sq., 50).

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity Linking meets Word Sense Disambiguation: A Unified Approach. *Transactions of the Association for Computational Linguistics* 2:231–244. (Cited on pages 33, 35, 52 sq.).

Tareq Al-Moslmi, Marc Gallofré Ocaña, Andreas L. Opdahl, and Csaba Veres. 2020. Named Entity Extraction for Knowledge Graphs: A Literature Overview. *IEEE Access* 8:32862–32881. (Cited on pages 31, 36).

David Mueller and Greg Durrett. 2018. Effective Use of Context in Noisy Entity Linking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing,* 1024–1029. Brussels, Belgium: Association for Computational Linguistics. (Cited on pages 47, 50).

Isaiah Onando Mulang', Kuldeep Singh, Chaitali Prabhu, Abhishek Nadgeri, Johannes Hoffart, and Jens Lehmann. 2020. Evaluating the Impact of Knowledge Graph Context on Entity Disambiguation Models. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management,* 2157–2160. CIKM '20. Virtual Event, Ireland: Association for Computing Machinery. (Cited on pages 41, 47, 50, 53).

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical Losses and New Resources for Fine-grained Entity Typing and Linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 97–109. Melbourne, Australia: Association for Computational Linguistics. (Cited on pages 4, 68).

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticæ Investigationes* 30 (1): 3–26. (Cited on pages 33 sq.).

Vivi Nastase, Rada Mihalcea, and Dragomir R. Radev. 2015. A survey of graphs in natural language processing. *Natural Language Engineering* 21 (5): 665–698. (Cited on page 8).

Roberto Navigli. 2009. Word Sense Disambiguation: A Survey. *ACM Computing Surveys* 41 (2). (Cited on pages 3, 35, 88).

Mojtaba Nayyeri, Sahar Vahdati, Can Aykul, and Jens Lehmann. 2021. 5* Knowledge Graph Embeddings with Projective Transformations. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 9064–9072. held virtually. (Cited on page 24).

Mojtaba Nayyeri, Sahar Vahdati, Jens Lehmann, and Hamed Shariat Yazdi. 2019. Soft Marginal TransE for Scholarly Knowledge Graph Completion. arXiv: 1904.12211 [cs.AI]. (Cited on page 23).

Rostislav Nedelchev, Debanjan Chaudhuri, Jens Lehmann, and Asja Fischer. 2020. End-to-End Entity Linking and Disambiguation leveraging Word and Knowledge Graph Embeddings. arXiv: 2002.11143 [cs.CL]. (Cited on pages 24, 40).

Denis Newman-Griffis, Albert M. Lai, and Eric Fosler-Lussier. 2018. Jointly Embedding Entities and Text with Distant Supervision. In *Proceedings of The Third Workshop on Representation Learning for NLP,* 195–206. Melbourne, Australia: Association for Computational Linguistics. (Cited on pages 37, 39, 45 sq., 50).

Thien Huu Nguyen, Nicolas Fauceglia, Mariano Rodriguez Muro, Oktie Hassanzadeh, Alfio Massimiliano Gliozzo, and Mohammad Sadoghi. 2016. Joint Learning of Local and Global Features for Entity Linking via Neural Networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers,* 2310–2320. Osaka, Japan: The COLING 2016 Organizing Committee. (Cited on pages 40, 45 sq., 50, 53).

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning,* 809–816. ICML'11. Bellevue, WA, USA: Omnipress. (Cited on pages 23 sq.).

Feng Nie, Yunbo Cao, Jinpeng Wang, Chin-Yew Lin, and Rong Pan. 2018. Mention and Entity Description Co-Attention for Entity Disambiguation. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 5908–5915. New Orleans, LA, USA. (Cited on pages 41, 46, 50, 53).

Michael A. Nielsen. 2015. Neural Networks and Deep Learning. Determination Press. (Cited on pages 14, 16).

Rasha Obeidat, Xiaoli Fern, Hamed Shahbazi, and Prasad Tadepalli. 2019. Description-Based Zero-shot Fine-Grained Entity Typing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* 807–814. Minneapolis, MN, USA: Association for Computational Linguistics. (Cited on pages 61 sqq., 66).

Italo L. Oliveira, Renato Fileto, René Speck, Luís P.F. Garcia, Diego Moussallem, and Jens Lehmann. 2021. Towards holistic Entity Linking: Survey and directions. *Information Systems* 95:101624. (Cited on pages 30 sq., 33).

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling Fine-Grained Entity Types with Box Embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers),* 2051–2064. Online: Association for Computational Linguistics. (Cited on pages 68, 81).

Yasumasa Onoe and Greg Durrett. 2019. Learning to Denoise Distantly-Labeled Data for Entity Typing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* 2407–2417. Minneapolis, MN, USA: Association for Computational Linguistics. (Cited on pages 65 sq.).

———. 2020. Fine-Grained Entity Typing for Domain Independent Entity Linking. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 8576–8583. New York Hilton Midtown, NY, USA. (Cited on pages 5, 37, 46 sq., 50, 53, 61).

Pekka Orponen. 1994. Computational Complexity of Neural Networks: A Survey. *Nordic Journal of Computing* 1 (1): 94–110. (Cited on page 48).

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66. Previous number = SIDL-WP-1999-0120. Stanford InfoLab. (Cited on pages 43, 91).

Weiran Pan, Wei Wei, and Xian-Ling Mao. 2021. Context-aware Entity Typing in Knowledge Graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2021,* 2240–2250. Punta Cana, Dominican Republic: Association for Computational Linguistics. (Cited on page 61).

Weiran Pan, Wei Wei, and Feida Zhu. 2022. Automatic Noisy Label Correction for Fine-Grained Entity Typing. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22,* 4317–4323. Vienna, Austria: International Joint Conferences on Artificial Intelligence Organization. (Cited on page 66).

Alexander Panchenko, Fide Marten, Eugen Ruppert, Stefano Faralli, Dmitry Ustalov, Simone Paolo Ponzetto, and Chris Biemann. 2017. Unsupervised, Knowledge-Free, and Interpretable Word Sense Disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations,* 91–96. Copenhagen, Denmark: Association for Computational Linguistics. (Cited on page 73).

Alexander Panchenko, Eugen Ruppert, Stefano Faralli, Simone P. Ponzetto, and Chris Biemann. 2018. Building a Web-Scale Dependency-Parsed Corpus from Common Crawl. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018),* 1816–1823. Miyazaki, Japan: European Language Resources Association (ELRA). (Cited on page 76).

Alexander Panchenko, Eugen Ruppert, Stefano Faralli, Simone Paolo Ponzetto, and Chris Biemann. 2017. Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers,* 86–98. Valencia, Spain: Association for Computational Linguistics. (Cited on page 73).

Kunyuan Pang, Haoyu Zhang, Jie Zhou, and Ting Wang. 2022. Divide and Denoise: Learning from Noisy Labels in Fine-Grained Entity Typing with Cluster-Wise Loss Correction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 1997–2006. Dublin, Ireland: Association for Computational Linguistics. (Cited on page 66).

Alberto Parravicini, Rhicheek Patra, Davide B. Bartolini, and Marco D. Santambrogio. 2019. Fast and Accurate Entity Linking via Graph Embedding. In *Proceedings of the 2nd Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA).* GRADES-NDA'19. Amsterdam, Netherlands: Association for Computing Machinery. (Cited on page 40).

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (85): 2825–2830. (Cited on page 77).

Maria Pelevina, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. Making Sense of Word Embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP,* 174–183. Berlin, Germany: Association for Computational Linguistics. (Cited on page 73).

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* 701–710. KDD '14. New York, NY, USA: Association for Computing Machinery. (Cited on pages 23, 40, 89 sq.).

Maria Pershina, Yifan He, and Ralph Grishman. 2015. Personalized Page Rank for Named Entity Disambiguation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 238–243. Denver, CO, USA: Association for Computational Linguistics. (Cited on pages 37, 43, 55).

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers),* 2227–2237. New Orleans, LA, USA: Association for Computational Linguistics. (Cited on pages 41, 48, 67).

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),* 43–54. Hong Kong, China: Association for Computational Linguistics. (Cited on pages 6, 37 sq., 45, 47, 50).

Francesco Piccinno and Paolo Ferragina. 2014. From TagME to WAT: A New Entity Annotator. In *Proceedings of the First International Workshop on Entity Recognition &; Disambiguation,* 55–62. ERD '14. Gold Coast, Australia: Association for Computing Machinery. (Cited on pages 52 sq.).

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-Yet-Effective Entity Embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020,* 803–818. Online: Association for Computational Linguistics. (Cited on pages 47, 50, 58).

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations,* 101–108. Online: Association for Computational Linguistics. (Cited on page 77).

Jing Qian, Yibin Liu, Lemao Liu, Yangming Li, Haiyun Jiang, Haisong Zhang, and Shuming Shi. 2021. Fine-grained Entity Typing without Knowledge Base. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing,* 5309–5319. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. (Cited on pages 64, 72).

Priya Radhakrishnan, Partha Talukdar, and Vasudeva Varma. 2018. ELDEN: Improved Entity Linking Using Densified Knowledge Graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers),* 1844–1853. New Orleans, LA, USA: Association for Computational Linguistics. (Cited on pages 37, 40, 45 sq., 50, 53).

Jonathan Raiman and Olivier Raiman. 2018. DeepType: Multilingual Entity Linking by Neural Type System Evolution. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 5406–5413. New Orleans, LA, USA. (Cited on pages 5, 46 sq., 49 sq., 53).

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and Global Algorithms for Disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1,* 1375–1384. HLT '11. Portland, OR, USA: Association for Computational Linguistics. (Cited on pages 29, 51 sqq., 93).

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora [in English]. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks,* 45–50. Valletta, Malta: European Language Resources Association (ELRA). (Cited on page 92).

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),* 3982–3992. Hong Kong, China: Association for Computational Linguistics. (Cited on pages 22, 74).

———. 2020. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 4512–4525. Online: Association for Computational Linguistics. (Cited on page 23).

Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016. AFET: Automatic Fine-Grained Entity Typing by Hierarchical Partial-Label Embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing,* 1369–1378. Austin, TX, USA: Association for Computational Linguistics. (Cited on pages 61, 64, 68).

Xiang Ren, Wenqi He, Meng Qu, Clare R. Voss, Heng Ji, and Jiawei Han. 2016. Label Noise Reduction in Entity Typing by Heterogeneous Partial-Label Embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* 1825–1834. KDD '16. San Francisco, CA, USA: Association for Computing Machinery. (Cited on page 66).

Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R. Voss, Heng Ji, Tarek F. Abdelzaher, and Jiawei Han. 2017. CoType: Joint Extraction of Typed Entities and Relations with Knowledge Bases. In *Proceedings of the 26th International Conference on World Wide Web,* 1015–1024. WWW '17. Perth, Australia: International World Wide Web Conferences Steering Committee. (Cited on page 64).

Yankun Ren, Jianbin Lin, and Jun Zhou. 2020. Neural Zero-Shot Fine-Grained Entity Typing. In *Companion Proceedings of the Web Conference 2020,* 846–847. WWW '20. Taipei, Taiwan: Association for Computing Machinery. (Cited on page 66).

Martin Riedl and Chris Biemann. 2013. Scaling to Large$^3$ Data: An Efficient and Effective Method to Compute Distributional Thesauri. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing,* 884–890. Seattle, WA, USA: Association for Computational Linguistics. (Cited on page 25).

Giuseppe Rizzo, Marieke van Erp, and Raphaël Troncy. 2014. Benchmarking the Extraction and Disambiguation of Named Entities on the Semantic Web. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14),* 4593–4600. Reykjavik, Iceland: European Language Resources Association (ELRA). (Cited on page 33).

Herbert Robbins and Sutton Monro. 1951. A Stochastic Approximation Method. *The Annals of Mathematical Statistics* 22 (3): 400–407. (Cited on page 16).

Michael Röder, Ricardo Usbeck, Sebastian Hellmann, Daniel Gerber, and Andreas Both. 2014. N$^3$ - A Collection of Datasets for Named Entity Recognition and Disambiguation in the NLP Interchange Format. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14),* 3529–3533. Reykjavik, Iceland: European Language Resources Association (ELRA). (Cited on pages 92 sq., 95).

Michael Röder, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. 2018. GERBIL - Benchmarking Named Entity Recognition and Linking consistently. *Semantic Web* 9 (5): 605–625. (Cited on pages 52, 93).

Henry Rosales-Méndez, Aidan Hogan, and Barbara Poblete. 2018. VoxEL: A Benchmark Dataset for Multilingual Entity Linking. In *International Semantic Web Conference (2),* 11137:170–186. Lecture Notes in Computer Science. Monterey, CA, USA: Springer. (Cited on pages 92 sq.).

Daniel Ruffinelli, Samuel Broscheit, and Rainer Gemulla. 2020. You CAN Teach an Old Dog New Tricks! On Training Knowledge Graph Embeddings. In *International Conference on Learning Representations.* Online. (Cited on page 24).

Eugen Ruppert, Manuel Kaufmann, Martin Riedl, and Chris Biemann. 2015. JoBimViz: A Web-based Visualization for Graph-based Distributional Semantic Models. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations,* 103–108. Beijing, China: Association for Computational Linguistics / The Asian Federation of Natural Language Processing. (Cited on pages 8, 25, 72, 74).

**Özge Sevgili**, Alexander Panchenko, and Chris Biemann. 2019. Improving Neural Entity Disambiguation with Graph Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop,* 315–322. Florence, Italy: Association for Computational Linguistics. (Cited on pages iv, 40, 87).

**Özge Sevgili**, Steffen Remus, Abhik Jana, Alexander Panchenko, and Chris Biemann. 2024. Unsupervised Ultra-Fine Entity Typing with Distributionally Induced Word Senses. In *Analysis of Images, Social Networks and Texts,* 126–140. Yerevan, Armenia, Cham: Springer Nature Switzerland. (Cited on pages iv, 71).

**Özge Sevgili**, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. 2022. Neural entity linking: A survey of models based on deep learning. *Semantic Web* 13 (3): 527–570. (Cited on pages iv, 6, 28 sq., 38, 41, 43, 45 sq.).

Hamed Shahbazi, Xiaoli Fern, Reza Ghaeini, Chao Ma, Rasha Mohammad Obeidat, and Prasad Tadepalli. 2018. Joint Neural Entity Disambiguation with Output Space Search. In *Proceedings of the 27th International Conference on Computational Linguistics,* 2170–2180. Santa Fe, NM, USA: Association for Computational Linguistics. (Cited on pages 37, 44, 46, 50, 53).

Hamed Shahbazi, Xiaoli Z. Fern, Reza Ghaeini, Rasha Obeidat, and Prasad Tadepalli. 2019. Entity-aware ELMo: Learning Contextual Entity Representation for Entity Disambiguation. arXiv: 1908.05762 [cs.CL]. (Cited on pages 38, 41, 47, 50, 53 sqq., 58).

Rahul Sharnagat. 2014. Named entity recognition: A literature survey. *Center For Indian Language Technology,* (cited on page 34).

Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Transactions on Knowledge & Data Engineering* 27 (02): 443–460. (Cited on pages 30 sq., 36, 51).

Senbao Shi, Zhenran Xu, Baotian Hu, and Min Zhang. 2024. Generative Multimodal Entity Linking. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024),* 7654–7665. Torino, Italia: ELRA / ICCL, May. (Cited on page 101).

Wei Shi, Siyuan Zhang, Zhiwei Zhang, Hong Cheng, and Jeffrey Xu Yu. 2020. Joint Embedding in Named Entity Linking on Sentence Level. arXiv: 2002.04936 [cs.CL]. (Cited on pages 40, 56 sq.).

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An Attentive Neural Architecture for Fine-grained Entity Type Classification. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction,* 69–74. San Diego, CA, USA: Association for Computational Linguistics. (Cited on page 68).

———. 2017. Neural Architectures for Fine-grained Entity Type Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers,* 1271–1280. Valencia, Spain: Association for Computational Linguistics. (Cited on page 68).

Darya Shlyk, Tudor Groza, Marco Mesiti, Stefano Montanelli, and Emanuele Cavalleri. 2024. REAL: A Retrieval-Augmented Entity Linking Approach for Biomedical Concept Recognition. In *Proceedings of the 23rd Workshop on Biomedical Natural Language Processing,* 380–389. Bangkok, Thailand: Association for Computational Linguistics, August. (Cited on page 101).

Avirup Sil, Gourab Kundu, Radu Florian, and Wael Hamza. 2018. Neural Cross-Lingual Entity Linking. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 5464–5472. New Orleans, LA, USA. (Cited on pages 37 sq., 46, 50, 53).

Jiří Šíma and Pekka Orponen. 2003. General-Purpose Computation with Neural Networks: A Survey of Complexity Theoretic Results. *Neural Computation* 15 (12): 2727–2778. (Cited on page 48).

Daniil Sorokin and Iryna Gurevych. 2018. Mixing Context Granularities for Improved Entity Linking on Question Answering Data across Entity Categories. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics,* 65–75. New Orleans, LA, USA: Association for Computational Linguistics. (Cited on pages 24, 40, 46, 50).

Valentin I. Spitkovsky and Angel X. Chang. 2012. A Cross-Lingual Dictionary for English Wikipedia Concepts. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12),* 3168–3175. Istanbul, Turkey: European Language Resources Association (ELRA). (Cited on pages 37 sq.).

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web,* 697–706. WWW '07. Banff, Canada: Association for Computing Machinery. (Cited on page 37).

Xuhui Sui, Ying Zhang, Kehui Song, Baohang Zhou, Guoqing Zhao, Xin Wei, and Xiaojie Yuan. 2022. Improving Zero-Shot Entity Linking Candidate Generation with Ultra-Fine Entity Type Information. In *Proceedings of the 29th International Conference on Computational Linguistics,* 2429–2437. Gyeongju, Republic of Korea: International Committee on Computational Linguistics. (Cited on pages 5, 72).

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized Language and Knowledge Embedding. In *Proceedings of the 28th International Conference on Computational Linguistics,* 3660–3670. Barcelona, Spain (Online): International Committee on Computational Linguistics. (Cited on page 6).

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling Mention, Context and Entity with Neural Networks for Entity Disambiguation. In *Proceedings of the 24th International Conference on Artificial Intelligence,* 1333–1339. IJCAI'15. Buenos Aires, Argentina: AAAI Press. (Cited on pages 40, 46, 50, 53).

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2,* 3104–3112. NIPS'14. Montréal, Canada: MIT Press. (Cited on page 18).

Hongyin Tang, Xingwu Sun, Beihong Jin, and Fuzheng Zhang. 2021. A Bidirectional Multi-paragraph Reading Model for Zero-shot Entity Linking. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 13889–13897. held virtually. (Cited on pages 47, 50).

Li Tianran, Yang Erguang, Zhang Yujie, Chen Yufeng, and Xu Jinan. 2021. Improving Entity Linking by Encoding Type Information into Entity Embeddings [in English]. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics,* 1087–1095. Huhhot, China: Chinese Information Processing Society of China. (Cited on page 5).

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of The 33rd International Conference on Machine Learning,* 48:2071–2080. Proceedings of Machine Learning Research. New York, NY, USA: PMLR. (Cited on page 24).

Chen-Tse Tsai and Dan Roth. 2016. Cross-lingual Wikification Using Multilingual Embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 589–598. San Diego, CA, USA: Association for Computational Linguistics. (Cited on pages 39, 45 sq., 50, 53).

Shyam Upadhyay, Nitish Gupta, and Dan Roth. 2018. Joint Multilingual Supervision for Cross-lingual Entity Linking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing,* 2486–2495. Brussels, Belgium: Association for Computational Linguistics. (Cited on pages 46, 50, 53, 55).

Johannes M. van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P. de Vries. 2020. REL: An Entity Linker Standing on the Shoulders of Giants. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval,* 2197–2200. New York, NY, USA: Association for Computing Machinery. (Cited on page 33).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems,* 6000–6010. NIPS'17. Long Beach, CA, USA: Curran Associates Inc. (Cited on pages 18 sq., 48).

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Communications of the ACM* (New York, NY, USA) 57 (10): 78–85. (Cited on pages 23 sq., 31, 34).

Han Wang, Jin Guang Zheng, Xiaogang Ma, Peter Fox, and Heng Ji. 2015. Language and Domain Independent Entity Linking with Quantified Collective Validation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing,* 695–704. Lisbon, Portugal: Association for Computational Linguistics. (Cited on page 45).

Jianing Wang, Wenkang Huang, Minghui Qiu, Qiuhui Shi, Hongbin Wang, Xiang Li, and Ming Gao. 2022. Knowledge Prompting in Pre-trained Language Model for Natural Language Understanding. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing,* 3164–3177. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. (Cited on page 6).

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering* 29 (12): 2724–2743. (Cited on pages 23 sq.).

Ruili Wang, Feng Hou, Steven F. Cahan, Li Chen, Xiaoyun Jia, and Wanting Ji. 2023. Fine-Grained Entity Typing With a Type Taxonomy: A Systematic Review. *IEEE Transactions on Knowledge and Data Engineering* 35 (5): 4794–4812. (Cited on pages 61, 69).

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021,* 1405–1418. Online: Association for Computational Linguistics. (Cited on page 6).

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Transactions of the Association for Computational Linguistics* 9:176–194. (Cited on page 6).

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence,* 1112–1119. Québec City, Canada. (Cited on pages 23 sq.).

Ralph Weischedel and Ada Brunstein. 2005. BBN Pronoun Coreference and Entity Type Corpus. Philadelphia: Linguistic Data Consortium. (Cited on page 69).

WordNet: An Electronic Lexical Database. 1998. Cambridge, MA, USA: MIT Press. (Cited on pages 35, 38).

Junshuang Wu, Richong Zhang, Yongyi Mao, Hongyu Guo, and Jinpeng Huai. 2019. Modeling Noisy Hierarchical Types in Fine-Grained Entity Typing: A Content-Based Weighting Approach. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19,* 5264–5270. Macao, China: International Joint Conferences on Artificial Intelligence Organization. (Cited on page 66).

Junshuang Wu, Richong Zhang, Yongyi Mao, Hongyu Guo, Masoumeh Soflaei, and Jinpeng Huai. 2020. Dynamic Graph Convolutional Networks for Entity Linking. In *Proceedings of The Web Conference 2020,* 1149–1159. WWW '20. Taipei, Taiwan: Association for Computing Machinery. (Cited on pages 40, 44 sq., 47, 50, 53).

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable Zero-shot Entity Linking with Dense Entity Retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 6397–6407. Online: Association for Computational Linguistics. (Cited on pages 38, 41, 47, 49 sqq., 53 sq., 58).

Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. 2019. Zero-Shot Learning—A Comprehensive Evaluation of the Good, the Bad and the Ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41 (9): 2251–2265. (Cited on page 14).

Amy Xin, Yunjia Qi, Zijun Yao, Fangwei Zhu, Kaisheng Zeng, Xu Bin, Lei Hou, and Juanzi Li. 2024. LLMAEL: Large Language Models are Good Context Augmenters for Entity Linking. arXiv: 2407.04020 [cs.CL]. (Cited on page 100).

Ji Xin, Hao Zhu, Xu Han, Zhiyuan Liu, and Maosong Sun. 2018. Put It Back: Entity Typing with Language Model Enhancement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing,* 993–998. Brussels, Belgium: Association for Computational Linguistics. (Cited on page 66).

Wenhan Xiong, Jiawei Wu, Deren Lei, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Imposing Label-Relational Inductive Bias for Extremely Fine-Grained Entity Typing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* 773–784. Minneapolis, MN, USA: Association for Computational Linguistics. (Cited on pages 67 sq.).

Nan Xu, Fei Wang, Bangzheng Li, Mingtao Dong, and Muhao Chen. 2022. Does Your Model Classify Entities Reasonably? Diagnosing and Mitigating Spurious Correlations in Entity Typing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing,* 8642–8658. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. (Cited on page 65).

Peng Xu and Denilson Barbosa. 2018. Neural Fine-Grained Entity Type Classification with Hierarchy-Aware Loss. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers),* 16–25. New Orleans, LA, USA: Association for Computational Linguistics. (Cited on page 68).

Mengge Xue, Weiming Cai, Jinsong Su, Linfeng Song, Yubin Ge, Yubao Liu, and Bin Wang. 2019. Neural Collective Entity Linking Based on Recurrent Random Walk Network Learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19,* 5327–5333. Macao, China: International Joint Conferences on Artificial Intelligence Organization. (Cited on pages 43, 47, 50, 53).

Vikas Yadav and Steven Bethard. 2018. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. In *Proceedings of the 27th International Conference on Computational Linguistics,* 2145–2158. Santa Fe, NM, USA: Association for Computational Linguistics. (Cited on page 34).

Yadollah Yaghoobzadeh, Heike Adel, and Hinrich Schütze. 2017. Noise Mitigation for Neural Entity Typing and Relation Extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers,* 1183–1194. Valencia, Spain: Association for Computational Linguistics. (Cited on page 61).

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations,* 23–30. Online: Association for Computational Linguistics. (Cited on pages 39 sq.).

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 6442–6454. Online: Association for Computational Linguistics. (Cited on page 6).

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning,* 250–259. Berlin, Germany: Association for Computational Linguistics. (Cited on pages 37, 40, 43, 45 sq., 50, 53, 56).

———. 2017. Learning Distributed Representations of Texts and Entities from Knowledge Base. *Transactions of the Association for Computational Linguistics* 5:397–411. (Cited on page 39).

Ikuya Yamada, Koki Washio, Hiroyuki Shindo, and Yuji Matsumoto. 2021. Global Entity Disambiguation with Pretrained Contextualized Embeddings of Words and Entities. arXiv: 1909.00426 [cs.CL]. (Cited on pages 38, 41, 44, 47, 50, 53 sqq., 58).

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. arXiv: 1412.6575 [cs.CL]. (Cited on page 24).

Xiyuan Yang, Xiaotao Gu, Sheng Lin, Siliang Tang, Yueting Zhuang, Fei Wu, Zhigang Chen, Guoping Hu, and Xiang Ren. 2019. Learning Dynamic Context Augmentation for Global Entity Linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),* 271–281. Hong Kong, China: Association for Computational Linguistics. (Cited on pages 43 sq., 47, 50, 53).

Zonghai Yao, Liangliang Cao, and Huapu Pan. 2020. Zero-shot Entity Linking with Efficient Long Range Sequence Modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020,* 2517–2522. Online: Association for Computational Linguistics. (Cited on pages 47, 50).

Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, and Gerhard Weikum. 2012. HYENA: Hierarchical Type Classification for Entity Names. In *Proceedings of COLING 2012: Posters,* 1361–1370. Mumbai, India: The COLING 2012 Organizing Committee. (Cited on page 69).

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2018. Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. *IEEE Computational Intelligence Magazine* 13 (3): 55–75. (Cited on page 29).

Siyu Yuan, Deqing Yang, Jiaqing Liang, Zhixu Li, Jinxi Liu, Jingyue Huang, and Yanghua Xiao. 2022. Generative Entity Typing with Curriculum Learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing,* 3061–3073. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. (Cited on pages 62, 67).

Zheng Yuan and Doug Downey. 2018. OTyper: A Neural Architecture for Open Named Entity Typing. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence.* AAAI'18/IAAI'18/EAAI'18. New Orleans, LA, USA: AAAI Press. (Cited on page 66).

Wayne W. Zachary. 1977. An Information Flow Model for Conflict and Fission in Small Groups. *Journal of Anthropological Research* 33 (4): 452–473. (Cited on page 23).

Haoyu Zhang, Dingkun Long, Guangwei Xu, Muhua Zhu, Pengjun Xie, Fei Huang, and Ji Wang. 2020. Learning with Noise: Improving Distantly-Supervised Fine-grained Entity Typing via Automatic Relabeling. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20,* 3808–3815. Main track. Yokohama, Japan: International Joint Conferences on Artificial Intelligence Organization. (Cited on page 66).

Sheng Zhang, Kevin Duh, and Benjamin Van Durme. 2018. Fine-grained Entity Typing through Increased Discourse Context and Adaptive Classification Thresholds. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics,* 173–179. New Orleans, LA, USA: Association for Computational Linguistics. (Cited on page 68).

Shujian Zhang, Chengyue Gong, and Eunsol Choi. 2021. Learning with Different Amounts of Annotation: From Zero to Many Labels. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing,* 7620–7632. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics. (Cited on page 65).

Tao Zhang, Congying Xia, Chun-Ta Lu, and Philip Yu. 2020. MZET: Memory Augmented Zero-Shot Fine-grained Named Entity Typing. In *Proceedings of the 28th International Conference on Computational Linguistics,* 77–87. Barcelona, Spain (Online): International Committee on Computational Linguistics. (Cited on page 66).

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 1441–1451. Florence, Italy: Association for Computational Linguistics. (Cited on pages 6, 24).

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2024. A Survey of Large Language Models. arXiv: 2303.18223 [cs.CL]. (Cited on pages 6, 17, 100).

Yu Zhao, Anxiang Zhang, Ruobing Xie, Kang Liu, and Xiaojie Wang. 2020. Connecting Embeddings for Knowledge Graph Entity Typing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics,* 6419–6428. Online: Association for Computational Linguistics. (Cited on pages 61, 101).

Ben Zhou, Daniel Khashabi, Chen-Tse Tsai, and Dan Roth. 2018. Zero-Shot Open Entity Typing as Type-Compatible Grounding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing,* 2065–2076. Brussels, Belgium: Association for Computational Linguistics. (Cited on pages 67, 73).

Shuyan Zhou, Shruti Rijhwani, and Graham Neubig. 2019. Towards Zero-resource Cross-lingual Entity Linking. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019),* 243–252. Hong Kong, China: Association for Computational Linguistics. (Cited on pages 47, 50, 53).

Xinyu Zuo, Haijin Liang, Ning Jing, Shuang Zeng, Zhou Fang, and Yu Luo. 2022. Type-enriched Hierarchical Contrastive Strategy for Fine-Grained Entity Typing. In *Proceedings of the 29th International Conference on Computational Linguistics,* 2405–2417. Gyeongju, Republic of Korea: International Committee on Computational Linguistics. (Cited on pages 64, 68).

Stefan Zwicklbauer, Christin Seifert, and Michael Granitzer. 2016a. DoSeR - A Knowledge-Base-Agnostic Framework for Entity Disambiguation Using Semantic Embeddings. In *The Semantic Web. Latest Advances and New Domains,* 182–198. Cham: Springer International Publishing. (Cited on pages 39, 43).

———. 2016b. Robust and Collective Entity Disambiguation through Semantic Embeddings. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval,* 425–434. SIGIR '16. Pisa, Italy: Association for Computing Machinery. (Cited on pages 36, 39, 43, 45 sq., 49 sq., 53).