Gesellschaft für Informatik (Hrsg.): SKILL 2019,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2019 1

Open Information Extraction gestützte Pipeline für einen deutschsprachigen Wissensgraphen

Marco Lehner[1] Anna Sauer[2] Christopher Schmidt[3] Lukas Schwarz[4]

Abstract: Eine zentrale Herausforderung bei der Erstellung von Wissensgraphen aus natürlichsprachigen Texten besteht darin, geeignete Werkzeuge für unterschiedliche Sprachen zu entwickeln. Besonders abseits des Englischen sind einsatzfähige Architekturen Mangelware. In diesem Paper stellen wir eine mögliche Pipeline vor, die auf Basis von Open Information Extraction (OIE) einen RDF/OWL-Wissensgraphen aus deutschen Texten extrahiert. Dabei verbinden wir verschiedene bestehende Werkzeuge zur Natürlichen Sprachverarbeitung miteinander, die eigens für die deutsche Sprache konstruiert wurden. Während die Relation Extraction zum Großteil auf Dependency Parsing basiert, konzentrieren wir uns bei der Entity Extraction mithilfe von Named Entity Recognition auf Eigennamen, vor allem von Personen.

Keywords: Wissensgraph; Open Information Extraction; Natürliche Sprachverarbeitung; Semantic Web

1 Motivation

Wissengraphen wie Googles Knowledge Graph, YAGO, Wikidata oder DBpedia stellen Versuche dar, für das Semantic Web und seine Anwendungen eine große Wissensbasis über Entitäten (Personen, Orte, Organisationen) und ihren Beziehungen aufzubauen [EW16]. Eine der Methoden zur Erstellung von Wissengraphen ist die Open Information Extraction (OIE): Dabei werden Informationen mithilfe Natürlicher Sprachverarbeitung automatisiert aus Texten extrahiert. Die OIE benötigt dabei im Gegensatz zur klassischen Information Extraction keine manuell annotierten Trainingsdaten [Ba07].

Allerdings muss die Vorgehensweise bei der OIE an die Sprache der gewählten Textquellen angepasst werden [Et11]. Während für das Englische [Ga17] und weitere Sprachen wie Chinesisch [Wa15] oder Arabisch [KL17] bereits Ansätze existieren, um auf OIE basierend Wissensgraphen zu generieren, bestehen für das Deutsche noch keine uns bekannten Versuche in dieser Richtung. In diesem Paper stellen wir daher die Pipeline zum Aufbau eines Wissensgraphen aus deutschsprachigen Texten vor, die wir im Rahmen eines einsemestrigen Projekts an der Universität Bamberg entwickelt haben.

[1] Universität Bamberg, marco.lehner@posteo.net
[2] Universität Bamberg, auersanna@googlemail.com
[3] Universität Bamberg, c.schmidt914@gmail.com
[4] Universität Bamberg, lukas.schwarz@posteo.de

2 Verwandte Arbeiten

	Unsere Pipeline	**FRED** [Ga17]	**KParser** [Sh15]	**AWAKE** [Bo14]
Extraktionsart	Erstellung von OIE-Tupeln mithilfe von PropsDE [Fa16]	Ermitteln von Diskursrepräsentationsstrukturen mit Boxer [Bo08]	Ableitung semantischer Rollen aus Dependency Trees	Extraktion von wenigen Entitätstypen, Relationen und Quantitäten mittels BBN SERIF [Ra11]
Pipelineaufbau	• Part of Speech Tagging • Coreference Resolution • Named Entity Recognition • Entity Linking • Relation Extraction	• Named Entity Recognition • Coreference Resolution • Einführen einer Taxonomie • Relation Extraction • Semantic Role Labeling • Frame/Situation Extraction • Entity Linking • Validierung	• Dependency Parsing • Mapping auf Relationen der Knowledge Machine Ontologie [CPW04] • Entity Linking • Semantic Role Labeling	• Dependency Parsing • Mention Detection • Coreference Resolution • Relation Extraction
Art des Graphen	Entitätszentrisch	Semantikorientiert	Semantikorientiert	Entitätszentrisch
Sprachunabhängigkeit	Nur Deutsch	Nur Englisch, anwendbar auf andere Sprachen dank BING-Übersetzung	Nur Englisch	Nur Englisch

Tab. 1: Vergleich bestehender Pipelines zur Wissensextraktion mit unserer Eigenentwicklung.

In Tabelle 1 werden vier Systeme zur Informationsextraktion nebeneinandergestellt. Die Art des Graphen unterteilen wir dabei in die Kategorien entitätszentriert und semantikorientiert. Entitätszentrierte Graphen fokussieren sich auf in den Texten enthaltene Entitäten und verbinden diese mit einer vergleichsweise geringen Anzahl an Prädikaten. Semantikorientierte Graphen versuchen die semantischen Rollen in einem Satz möglichst granular darzustellen und so den einzelnen Satz zu repräsentieren. Hier werden wenige Entitäten über vielen Kanten verbunden.

3 Lösungsansatz

Unsere Pipeline besteht aus folgenden Verarbeitungsschritten:

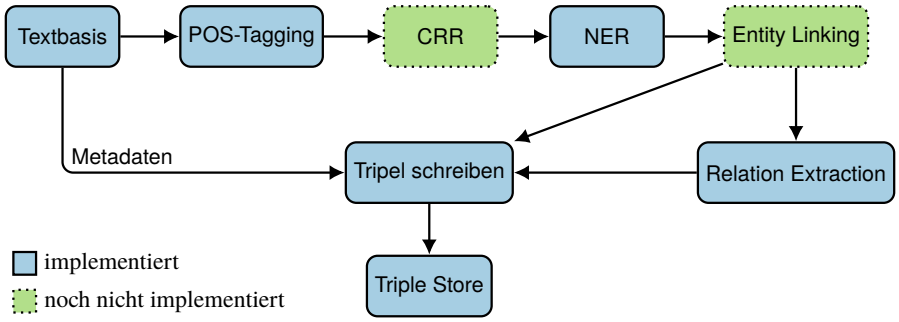


Abb. 1: Gesamtstruktur der Pipeline

Zunächst muss eine **Textdatenbasis** geschaffen werden. Je nach späterem Anwendungszweck des Wissensgraphen kann es dabei nützlich sein, **Metadaten zu den einzelnen Texten zu extrahieren**, um jedes verarbeitete Dokument im Wissensgraphen zu repräsentieren und mit den jeweils extrahierten Tripeln zu verknüpfen. Dies gewährleistet zudem die spätere Verifizierbarkeit der Tripel über einen Rückgriff auf die Textquelle.

Im nächsten Schritt wird durch **Part of Speech (POS) Tagging** jedem Wort der Texte seine jeweilige Wortart zugewiesen. Auf diesem sehr grundlegenden Verarbeitungsschritt bauen viele der weiteren Pipeline-Etappen auf.

Anschließend werden für die **Entity Extraction** mithilfe von **Named Entity Recognition (NER)** Eigennamen in den Texten identifiziert, die auf Personen, Organisationen, Orte etc. referieren und sich daher für die Aufnahme als Entitäten in den Graphen eignen. Wenn man zuvor noch durch **Coreference Resolution (CRR)** Verweise auf Named Entities z. B. von Pronomina ausgehend auflöst (z. B. „*Andreas Starke* ist Bürgermeister von Bamberg. *Er* gehört der SPD an.“), erschließt dies eine große Anzahl an zusätzlichen Informationen.

Als nächstes wird beim **Entity Linking** jeder Entität ein eindeutiger Identifikator zugeordnet und geprüft, ob sich der gleiche Name auf verschiedene Entitäten bezieht. Um den

Wissensgraphen im Sinne des Linked Open Data Prinzips mit anderen Ressourcen des Semantic Web zu verknüpfen, ergibt es Sinn, sich bei der Wahl der Identifikatoren auf bereits bestehende, umfangreiche Ontologien wie Babelnet oder Wikidata zu stützen.

Bei der anschließenden **Relation Extraction** werden durch Dependency Parsing Relationen zwischen den Entitäten in einem Satz identifiziert. Diese Relationen liegen dann in Tupelform mit Subjekt, Prädikat und Objekt(en) sowie eventuellen Modifikatoren vor. Ein ausführliches Beispiel dazu folgt im Abschnitt zur Implementierung.

Zum Abschluss müssen aus jedem Tupel noch ein oder mehrere **RDF-Tripel** gebildet werden. Jedes dieser Tripel besteht aus einem Prädikat, einem Subjekt und einem Objekt (z. B. in informeller Notation `pred:besuchen, subj:Angela Merkel, obj:Emmanuel Macron`). Zudem werden die Tupel der dem Graphen zugrundeliegenden Ontologie angepasst, die unter anderem festlegt, welche Typen von Relationen konkret im Graphen modelliert werden. Das jeweilige Tupelprädikat wird dabei in eine der in der Ontologie vorhandenen generischen Relationen umgewandelt. Beispielsweise wird „wohnen/leben in“ zu `dbo:residence` [Ab]. Die fertigen RDF-Tripel werden im Terse RDF Triple Language (Turtle) Format in einem Triple Store gespeichert. Der resultierende Wissensgraph kann dann mit SPARQL-Queries abgefragt werden.

4 Implementierung

Nach dieser allgemeinen Beschreibung der Grundarchitektur folgt nun eine eingehende Erläuterung der Implementierung in Python und der dabei eingesetzten Werkzeuge.

Unsere **Textdatenbasis** bilden lokaljournalistische Texte, die uns die Mediengruppe Oberfranken in Form eines XML-Dumps von 160.000 Artikeln aus dem Zeitraum ab 2005 zur Verfügung stellte. Zum Auslesen und Bereinigen des Dumps verwenden wir einen DOM-Parser. Wir extrahieren **Metadaten** wie Titel, Autor*in und Erscheinungsdatum zu den einzelnen Artikeln und speichern diese, wie alle anderen Zwischenergebnisse der Pipeline, zunächst in einer SQL-Datenbank.

Für das **POS-Tagging** kommt *spaCy* zum Einsatz, eine Python-basierte Open-Source-Bibliothek für Natürliche Sprachverarbeitung [HM].

Die **Named Entity Recognition** übernimmt ebenfalls *spaCy*. Für jede erkannte Entität wird ein RDF-Tripel geschrieben, das dieser einen Typ (Person, Organisation, Ort) zuweist. Für unseren ersten Testlauf entschieden wir uns, zunächst einen Fokus auf Personen und ihre Beziehungen zu legen. Deshalb werden die Sätze in diesem Schritt danach gefiltert, ob sie eine Named Entity der Kategorie Person enthalten.

Da das **Entity Linking** während der Erstellung der Tripel zu sehr hohen Laufzeiten führen würde, findet das Linking vorläufig erst bei Abruf der jeweiligen Entität in der später noch beschriebenen Weboberfläche statt. Dabei werden gleichlautende Einträge in

Wikidata gesucht und gegebenenfalls einige Eckdaten zur Person, wie Geburtsdatum und Parteizugehörigkeit, über den SPARQL-Endpoint abgefragt.

Bei der **Relation Extraction** bildet der von uns in Python implementierte *Restrictive Apposition Handler (RAH)* zunächst Tripel aus engen Appositionen vor Personennamen. „die Aschaffenburgerin Ruth Weiss“ ergibt hierbei etwa `pred:from_rah, subj:Ruth Weiss, obj:Aschaffenburgerin`. Enge Appositionen, d. h. substantivische Beiwörter vor anderen Substantiven, eignen sich besonders für die Wissensextraktion, da sie vor allem in der journalistischen Sprache häufig auftreten und in den meisten Fällen vor Personennamen einen eng definierten semantischen Gehalt in Bezug auf die Person besitzen (Berufsfeld, geographische Herkunft). Um den RAH zu schreiben, setzten wir uns mit der dependenzgrammatischen Struktur von Sätzen mit engen Appositionen auseinander und leiteten daraus eine Anzahl an Regeln ab, um entsprechende Tripel zu extrahieren. Wir berücksichtigten auch Fälle, in denen eine Apposition sich auf mehrere Personen bezieht („die Informatikerinnen Constanze Kurz und Rena Tangens“) oder mehrere Appositionen für eine Person vorhanden sind („die Informatikerin und Datenschutzaktivistin Constanze Kurz“).

Das OIE-Tool *PropsDE* [Fa16] analysiert daraufhin die syntaktische Struktur jedes Satzes und gibt eine Reihe von Tupeln zurück. In den meisten Fällen bilden diese wie bereits beschrieben die dependenzgrammatische Baumstruktur des Satzes ab:

„2019 konstruierte Hedwig Zuckerl für die deutsche Sprache einen Wissensgraphen.“

```
konstruieren:(subj:Hedwig Zuckerl , dobj:einen Wissensgraphen ,
          prep_für:die deutsche Sprache , mod:2019 )
```

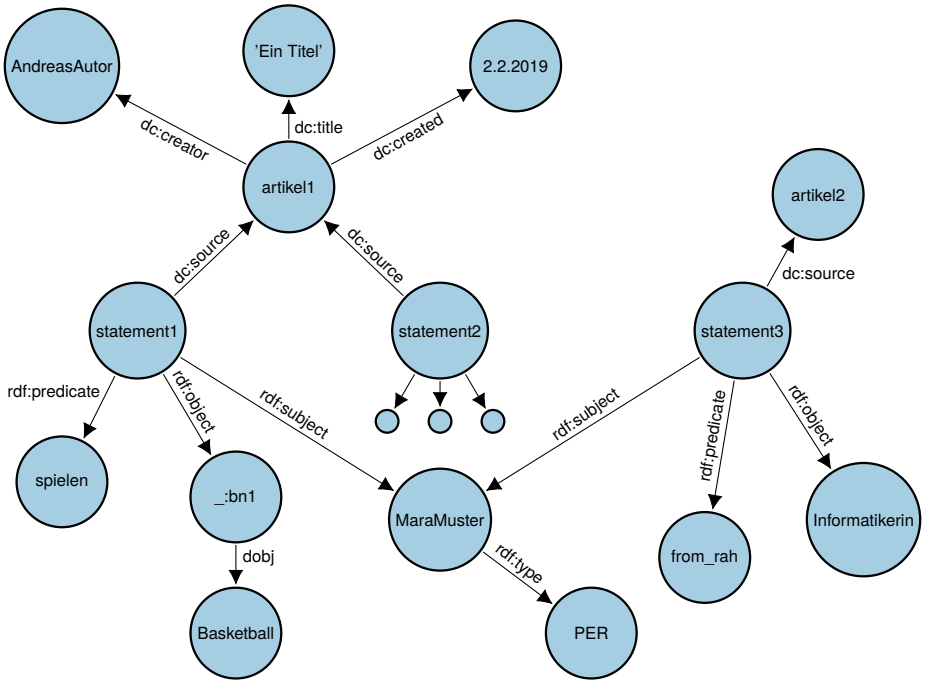
Andere syntaktische Muster werden dagegen als logische Aussagen modelliert. Weite Appositionen wie „Andreas Starke, Bürgermeister von Bamberg“ werden etwa in „SameAs“-Aussagen umgewandelt: `SameAs(subj:Andreas Starke , obj:Bürgermeister von Bamberg )`.

Bei der Umwandlung der Tupel in RDF-Tripel trafen wir eine Reihe von Modellierungsentscheidungen zur Gestaltung unseres Wissensgraphen (vgl. Abb. **??**[5]): Viele der Prädikate im Wissensgraphen entsprechen natürlichsprachigen Verben und können deshalb potentiell mehr als ein Objekt besitzen. Dies lässt sich in der klassischen RDF-Tripelform jedoch schwer abbilden. Deshalb wird als Objekt bei solchen Tripeln ein Blank Node eingefügt. An diesen Knoten können dann alle zum Prädikat gehörenden Objekte, aber auch Modifikatoren und adverbiale Bestimmungen gehängt werden.

Auch die zuvor erwähnten Metadaten zur Textquelle ordnen wir den Tripeln jeweils mithilfe von Reifikation zu, indem wir an jedes Tripel eine Statement-ID vergeben und diese dann als Subjekt eines weiteren Tripels mit der ID des Herkunftsartikels verknüpfen.

[5] dc: Dublin Core Prädikate für bibliographische Angaben [Du12]

Derzeit wird das Mapping von Prädikaten aus den Texten auf Relationen unserer Ontologie über einfache Zuordungslisten vorgenommen, die beinhalten, welche synonymen Prädikate zu einer Relation passen. Beispielsweise werden über eine solche Liste Verben des Sprechens wie „sagen“ oder „erzählen“ der Relation `sagen` zugeordnet.



Die fertigen RDF-Tripel werden in einen Apache Fuseki Triple Store überführt, so dass der Zugriff über SPARQL möglich wird. Zusätzlich gibt es ein einfaches Query-Frontend als Django-App, mi dem Endandwender*innen auch ohne die komplexe SPARQL-Syntax Anfragen stellen können und das neben einer Freitextsuche einige Filterfunktionen bietet.

5 Ergebnisdiskussion

Der von uns erzeugte Graph hat eine andere Struktur als die der meisten anderen Implementierungen, da wir nicht versuchen, die semantische Struktur eines oder mehrerer Sätze bis ins Detail abzubilden. Im hier beschriebenen Graphen stehen die Entitäten im Mittelpunkt und der Graph wächst dadurch, dass bei der Verarbeitung der Textbasis ein immer dichteres Netz aus Kanten zwischen diesen Entitäten entsteht. Wir halten diesen Ansatz für zielführender, da dadurch eine konsistente Struktur gewährleistet wird. So vereinfachen wir die Abfrage für die Anwender*innen.

Gleichzeitig müssen wir feststellen, dass OIE-Tupel ihrer Art nach nicht optimal beschaffen sind, um konsistente Einträge im Graphen zu erreichen. Häufig werden ganze Nominalphrasen extrahiert (z. B. `schwimmen(subj: der Lokalpolitiker Tizian Rieth , der für den Bamberger Stadtrat kandidiert , prep_in:Fluss , mod:täglich )`). So stehen Knoten nicht mehr wie gewünscht für eine Entität, sondern bilden eine semantische Einheit ab, die nicht sinnvoll abgefragt werden kann. Um dieses Problem zu lösen ist es nötig, über Alternativen zu PropsDE nachzudenken und möglicherweise sogar ein eigenes OIE-Werkzeug zu implementieren, um einen geeigneten reduzierten Output zu erzielen.

Was das Laufzeitverhalten unserer Pipeline betrifft, können wir aufgrund der Aneinanderreihung von teils bereits existierenden Tools keine detaillierte Gesamteinschätzung etwa in Form einer O-Notation geben. Im Einzelnen fiel bei spaCy ein verhältnismäßig geringer Zeitaufwand an, sobald das zugrundeliegende Sprachmodell für das Deutsche nur einmal geladen wurde. Der selbstgeschriebene RAH schlug ebenfalls nur mit geringem Aufwand zu Buche, da sich die Extraktion auf einige wenige zentrale Regeln konzentriert. PropsDE dagegen verursacht höhrere Laufzeitkosten durch eine größere Anzahl an Regeln, allerdings lässt die Verarbeitung, etwa durch satzweises Einlesen, gut parallelisieren. Beim Entity Linking fällt durch den Rückgriff auf die Web-API von Wikidata ebenfalls ein relativ hoher Aufwand bei jeder Anfrage an.

6 Ausblick

Um in Zukunft die Zahl der extrahierten Tripel zu erhöhen und so einen dichteren Graphen zu generieren, ist eine Implementierung der oben beschriebenen Coreference Resolution unerlässlich, etwa durch Einbindung des Tools CorZu [Tu16]. Auch eine elaboriertere Disambiguierung der Entitäten stellt eine weitere Herausforderung für die Zukunft dar.

Im Bereich des Predicate Mapping erfordert unsere derzeitige Vorgehensweise eine händische Identifikation von Zuordnungen zwischen Prädikaten und Relationen der Ontologie bereits zum Zeitpunkt, zu dem die Ontologie festgelegt wird. In Zukunft wäre ein flexiblerer Ansatz wünschenswert, das auch zum anfänglichen Modellierungszeitpunkt noch nicht mitbedachte Prädikate einer Relation in der Ontologie zuordnen kann. Dafür könnte ein lexikalische Datenbank für das Deutsche wie Open German WordNet [Op19] eingesetzt werden, die Aussagen zum Grad semantischer Nähe zwischen zwei Begriffen erlaubt. Mit einer solchen Datenbasis könnte man bei der Eingabe eines Prädikats herausfinden, ob dieses einer bestehenden Ontologierelation ähnlich genug ist, um auf sie gemappt zu werden.

Um einen dichteren und aussagekräftigeren Wissensgraphen zu erzielen, ist es außerdem notwendig, dass wir unsere Ontologie auch auf Entitäten wie Organisationen und Orte ausdehnen, nachdem wir zunächst auf die Repräsentation von Informationen zu Personen konzentrierten. Nicht zuletzt wäre es sinnvoll, Inferenzmechanismen zu entwickeln, um aus den bereits gefundenen Tripeln durch logisches Schließen automatisiert neue Tripel zu generieren.

Literaturverzeichnis

[Ab] About: residence. http://dbpedia.org/ontology/residence. Letzter Zugriff: 01.05.2019.

[Ba07] Banko, Michele; Cafarella, Michael J; Soderland, Stephen; Broadhead, Matthew; Etzioni, Oren: Open Information Extraction from the Web. In (Veloso, Manuela M., Hrsg.): Proceedings of the 20th International Joint Conference on Artificial Intelligence. S. 2670–2676, 2007.

[Bo08] Bos, Johan: Wide-coverage Semantic Analysis with Boxer. In: Proceedings of the 2008 Conference on Semantics in Text Processing. STEP '08, Association for Computational Linguistics, Stroudsburg, PA, USA, S. 277–286, 2008.

[Bo14] Boschee, Elizabeth; Freedman, Marjorie; Khanwalkar, Saurabh; Kumar, Anoop; Srivastava, Amit; Weischedel, Ralph: Researching persons & organizations: AWAKE: From Text to an Entity-centric Knowledge Base. In: 2014 IEEE International Conference on Big Data (Big Data). IEEE, S. 1030–1039, 2014.

[CPW04] Clark, Peter; Porter, Bruce; Works, Boeing Phantom: Km – The Knowledge Machine 2.0: Users Manual. Department of Computer Science, University of Texas at Austin, 2(5), 2004.

[Du12] Dublin Core Metadata Initiative (DCMI) Metadata Terms. http://dublincore.org/specifications/dublin-core/dcmi-terms/, 2012. Letzter Zugriff: 01.05.2019.

[Et11] Etzioni, Oren; Fader, Anthony; Christensen, Janara; Soderland, Stephen et al.: Open Information Extraction: The Second Generation. In: Twenty-Second International Joint Conference on Artificial Intelligence. 2011.

[EW16] Ehrlinger, Lisa; Wöß, Wolfram: Towards a Definition of Knowledge Graphs. SEMANTiCS (Posters, Demos, SuCCESS), 48, 2016.

[Fa16] Falke, Tobias; Stanovsky, Gabriel; Gurevych, Iryna; Dagan, Ido: Porting an Open Information Extraction System from English to German. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. The Association for Computational Linguistics, S. 892–898, 2016.

[Ga17] Gangemi, Aldo; Presutti, Valentina; Recupero, Diego Reforgiato; Nuzzolese, Andrea Giovanni; Draicchio, Francesco; Mongiovì, Misael: Semantic Web Machine Reading with FRED. Semantic Web, 8:873–893, 2017.

[HM] Honnibal, Matthew; Montani, Ines: , spaCy. https://spacy.io/. Letzter Zugriff: 31.01.2019.

[KL17] Ktob, Ahmed; Li, Zhoujun: The Arabic Knowledge Graph: Opportunities and Challenges. In: 2017 IEEE 11th International Conference on Semantic Computing (ICSC). S. 48–52, 2017.

[Op19] Open German WordNet. https://github.com/hdaSprachtechnologie/odenet, 2019. Letzter Zugriff: 23.06.2019.

[Ra11] Ramshaw, Lance; Boschee, Elizabeth; Friedman, Marjorie; MacBride, Jessica; Weischedel, Ralph; Zamanian, Alex: SERIF Language Processing — Effective Trainable Language Understanding. In (Olive, Joseph; Christianson, Caitlin; McCary, John, Hrsg.): Handbook of Natural Language Processing and Machine Translation, S. 626–631. Springer, 2011.

[Sh15] Sharma, Arpit; Vo, Nguyen Ha; Aditya, Somak; Baral, Chitta: Towards Addressing the Winograd Schema Challenge - Building and Using a Semantic Parser and a Knowledge Hunting Module. In (Yang, Qiang; Wooldridge, Michael, Hrsg.): Proceedings of the 24th International Joint Conference of Artificial Intelligence. S. 1319–1325, 2015.

[Tu16] Tuggener, Don: Incremental Coreference Resolution for German. Dissertation, 2016.

[Wa15] Wang, C.; Gao, M.; He, X.; Zhang, R.: Challenges in Chinese Knowledge Graph Construction. In: 2015 31st IEEE International Conference on Data Engineering Workshops (ICDEW). S. 59–61, 2015.