

Theorie und Anwendung von Named Entity Recognition in den Digital Humanities mit Fokus auf historische Texte des 17. Jahrhunderts

Masterarbeit

zur Erlangung des akademischen Grades

Master of Arts (MA)

an der Karl-Franzens-Universität Graz

vorgelegt von

Jacqueline More

am Institut Zentrum für Informationsmodellierung – Austrian
Centre for Digital Humanities

Begutachter: Roman Bleier, Mag. MA PhD

2021

Danksagung

Besonders danken möchte ich zuallererst meinem Betreuer Roman Bleier für seine tatkräftige Unterstützung während des Verfassens dieser Arbeit. Er stand mir bei all meinen Fragen immer hilfreich zur Seite und förderte mein Interesse für das Themengebiet der NER immer wieder aufs Neue.

Außerdem möchte ich meinen Studienkolleginnen und Studienkollegen am Zentrum für Informationsmodellierung danken, die mich mit aufmunterten Worten immer wieder motivierten und mir fachliche aber vor allem seelische Unterstützung gaben.

Ein ganz besonderer Dank geht von ganzem Herzen an meine Mutter, die immer für mich da war und an mich geglaubt hat, selbst wenn ich es nicht mehr getan habe. Ich wünsche, sie könnte diese Arbeit noch lesen.

Abschließend möchte ich meinem Freund Fabian danken, der mit viel Geduld, selbst in den schwersten Zeiten, für mich da war und mich jeden Tag anspornte. Er gab mir alle Zeit der Welt und sagte mir immer wieder, dass ich alles schaffe.

Inhaltsverzeichnis

1. Einleitung	1
1.1. Natural Language Processing in den Digital Humanities	1
1.2. Zielsetzung und Fragestellung	6
2. Forschungsfeld Named Entity Recognition	8
2.1. Geschichte und Entwicklungen	8
2.2. Begriffserklärung: Named Entity	13
2.3. Techniken und Verfahren	19
2.3.1. Named Entity Recognition und Classification	20
2.3.2. Disambiguation und Linking	23
2.4. Anwendungsmöglichkeiten	34
3. Maschinelles Lernen	38
3.1. Überwachtes Lernen (supervised Learning)	41
3.1.1. Klassifikation	42
3.1.2. Regression	44
3.1.3. Wichtige Algorithmen	45
3.2. Unüberwachtes Lernen (unsupervised Learning)	51
3.2.1. Clustering	52
3.3. Teilüberwachtes Lernen (semi-supervised Learning)	53
3.4. Bestärkendes Lernen (Reinforcement Learning)	54
3.5. Deep Learning	55
4. NER und die Anwendung auf historische Texte	57
4.1. Annotation der Trainingsdaten	57
4.2. Trainings-, Validierungs- und Testdaten	61
4.3. Evaluierungsmethoden	62

Inhaltsverzeichnis

4.4. NER Systeme und Tools für die DH	64
4.5. Probleme der NER bei historischen Texten	66
4.5.1. Digitalisierung von historischen Texten	66
4.5.2. Nested Entities	68
4.5.3. ältere Sprachstufen / Nicht-standardisierte Sprachen . .	69
5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.	71
5.1. Daten	71
5.1.1. Testdaten	71
5.1.2. Trainingsdaten	72
5.2. NER Workflow für Standardmodelle	74
5.2.1. Auswahl der Tools und Methoden	74
5.2.2. Anwendung auf Texte des 17. Jahrhunderts	76
5.2.3. Evaluierung	76
5.3. NER Workflow für das Training eines eigenen Modells	80
5.3.1. Auswahl des Tools	80
5.3.2. Vorverarbeitung	81
5.3.3. Training	83
5.3.4. Evaluierung	83
6. Conclusio	85
A. NLP-Tools	90
Literatur	92

Abbildungsverzeichnis

1.1. Terminologie: NLU, NLP, ASR	2
2.1. NE Typologie im Kontext der MUC	10
2.2. automatische Annotation mit SpaCy	14
2.3. NE Hierarchie	18
2.4. NER-Zugänge	21
2.5. Entity Linking	26
2.6. EL Task	26
2.7. von NL zum KG	32
2.8. NLP Tasks	33
3.1. Vergleich data-driven vs. symbolic	39
3.2. KI und maschinelles Lernen	40
3.3. überwachtes Lernen	41
3.4. binäre Klassifikation	43
3.5. lineare Regression	45
3.6. POS Tagging mit Spacy	46
3.7. SVM	49
3.8. Clustering	52
3.9. Skizzen verschiedener Pflanzen	53
3.10. bestärkendes Lernen	55
4.1. Aufteilung eines Datensatzes	61
4.2. Hierarchie linguistischer Tools	65
4.3. Synchrone und diachrone Schreibvariation	70
5.1. Workflow spaCy Standardmodell	77
5.2. Vergleich spaCy Standardmodell und annotierte Testdaten	78

Abbildungsverzeichnis

5.3. NLTK Ergebnisse	79
5.4. Training eines eigenen NER Modells	81
5.5. Anpassung der Tag Map	82
5.6. Teilung der Trainingsdaten in Sätze	82
5.7. Säuberung der Trainingsdaten	83
5.8. Training des NER Modells	83

Abkürzungsverzeichnis

DH	Digital Humanities
DL	Deep Learning
GIS	Geographic Information System
HTR	Handwritten Text Recognition
IE	Information Extraction
IR	Information Retrieval
KB	Knowledge Base
KG	Knowledge Graph
NE	Named Entity
NED	Named Entity Disambiguation
NEL	Named Entity Linking
NER	Named Entity Recognition
NERC	Named Entity Recognition and Classification
NLP	Natural Language Processing
NN	Neural Networks
SGML	Standard Generalized Markup Language
TEI	Text Encoding Initiative

OCR Optical Character Recognition

XML eXtensible Markup Language

1. Einleitung

1.1. Natural Language Processing in den Digital Humanities

Sprache ist ein komplexes Phänomen und eines der wichtigsten Kommunikationsmittel des Menschen. Täglich kommen wir in Kontakt mit gesprochener und geschriebener Sprache: beim Reden, Lesen, Chatten, Telefonieren oder Musik hören. Mit den heutigen Technologien gibt es kaum noch Beschränkungen wie wir kommunizieren. Wir leben in einer digitalisierten und vernetzten Welt, die sich in den letzten Jahren stark verändert hat und neue Methoden und Entwicklungen von Natural Language Processing (NLP) ermöglichen es uns, natürliche Sprache mit bestimmten Techniken und Methoden automatisiert zu verarbeiten.

Das Ziel von NLP ist gesprochene und geschriebene Sprache zu erkennen und zu analysieren. Mittlerweile wurden zahlreiche Tools zur Textanalyse entwickelt, wie beispielsweise Lemmatisierer, Part-of-Speech Tagger oder Parser. Aber auch Anwendungen, die über die linguistische Annotation hinausgehen, wie die Sentiment-Analyse, Information Retrieval (IR) oder Übersetzungen, gehören zur natürlichen Sprachverarbeitung (Kuhn, 2019, S. 566). Mattingly (2020) nennt in Zusammenhang mit NLP noch die Termini Automatic Speech Recognition (ASR) und Natural Language Understanding (NLU), vgl. Abbildung 1.1. So zählt er Methoden, wie beispielsweise die Sentiment-Analyse, Relation Extraction oder semantisches Parsing zu NLU. Da in dieser Arbeit nur mit geschriebenen Texten gearbeitet wird, ist die ASR hier nicht von Relevanz.

Der Großteil von NLP-Anwendungen wurde für aktuell gesprochene Sprachen entwickelt, d.h. anders als beispielsweise bei Sentiment-Analysen

1. Einleitung

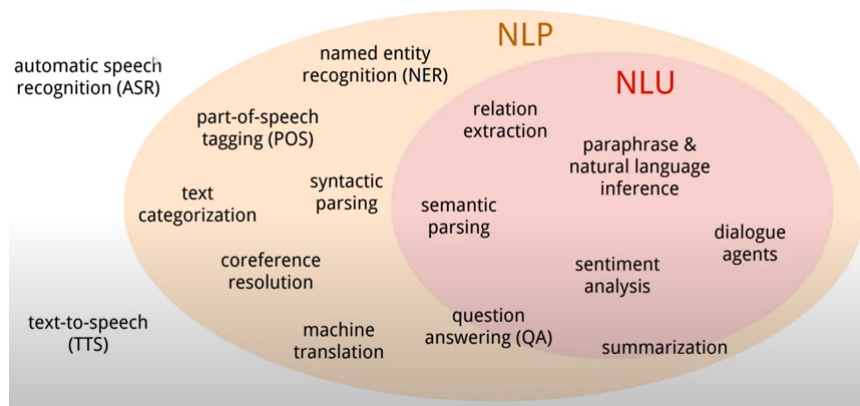


Abbildung 1.1.: Terminologie: NLU, NLP, ASR (Mattingly, 2020)

oder biomedizinischem Textmining, wird NLP für historische Texte meistens nicht durch kommerzielle Anwendungen unterstützt. Obwohl diese Tatsache für die digitale Geisteswissenschaft eine Herausforderung darstellt, werden die neuen Möglichkeiten, die sich durch digitale Daten eröffnen, für die geisteswissenschaftliche Forschung genutzt. Die Digital Humanities (DH) verbinden traditionelle qualitative Methoden mit quantitativen, computerbasierten Methoden und Tools, dazu zählen IR, Textanalysen, Data Mining, Visualisierungen und Geographic Information System (GIS) (vgl. Piotrowski, 2012, S. 6). Der Großteil der Testkorpora in den DH weicht von aktuellen Standards (frühe Sprachstufen, spezielle Textgenres, unerforschte Sprachen etc.) ab. Deshalb wurden Methoden entwickelt, die Workflows für Trainingsdaten, Anpassungen, Evaluierungen und neue Analyse-Komponenten für die DH definieren sollen (vgl. Kuhn, 2019, S.566).

Ein Großteil der geisteswissenschaftlichen Disziplinen beruhen auf Texten. Immer mehr Texte werden in digitaler Form zugänglich gemacht. Bibliotheken, Archive und Museen besitzen eine große Anzahl an historischen Dokumenten (Bücher, Aufzeichnungen, Journale, Briefe etc.). Es ist notwendig Dokumente des kulturellen Gedächtnisses zu schützen und zu erhalten, des-

1. Einleitung

halb wurden in den letzten Jahren große Anstrengungen in der Digitalisierung der Bestände unternommen. Um historische Texte maschinenlesbar und mit automatisierten Methoden analysierbar zu machen, müssen diese digitalisiert werden. Dadurch soll die Sichtbarkeit, der Zugang und die Nutzungsmöglichkeiten für Wissenschaft und Forschung sowie für die Öffentlichkeit verbessert werden.

Der Digitalisierungsvorgang ist der erste Schritt zur automatisierten Verarbeitung von natürlicher Sprache in historischen Dokumenten. Je besser und genauer die Daten digitalisiert werden, desto schneller und einfacher lassen sie sich verarbeiten. Der Digitalisierungsvorgang besteht aus mehreren Schritten. Um ein digitales Bild oder Faksimile zu erstellen, wird das Dokument zuerst eingescannt oder fotografiert. Die Bilder werden danach mit Metadaten (Autor, Datum, Dokumenttyp etc.) angereichert. Der dritte Schritt für reine Textdokumente besteht darin, den Text auf dem Papier in digitalen Text zu konvertieren. Texte können manuell oder automatisiert digitalisiert werden. Die manuelle Texterfassung wird auch *keying* genannt (Jannidis et al., 2017, S. 193), dabei wird die Vorlage per Hand abgetippt. Zur automatischen Texterfassung werden Anwendungen wie die Optical Character Recognition (OCR) oder Handwritten Text Recognition (HTR) eingesetzt. Ein OCR-Algorithmus verarbeitet ein hochauflösendes Bild einer bestimmten Quelle (Buch, Zeitungsartikel, etc.) und wandelt es in Text um. Viele DH-Projekte, die digitalisierte historische Dokumente verwenden, haben den Anspruch, Texte zu durchsuchen und automatisiert zu analysieren. Leider ist OCR sehr fehleranfällig, vor allem bei historischen Dokumenten. Diese Fehler beeinträchtigen die Forschung und die Erhaltung dieser Texte. Um diese Fehler zu korrigieren, werden Algorithmen zur Nachbearbeitung wie Neuronale Netzwerke (neural networks) eingesetzt, die auf Analysen natürlicher Sprache und maschinellen Lerntechniken basieren (vgl. Suissa et al., 2020). Leider ist die Verwendung dieser Neuronalen Netzwerke zur Korrektur von OCR-Fehlern in historischen Dokumenten noch immer wenig erforscht. Suissa et al. (2020) entwickeln beispielsweise eine Methode

1. Einleitung

zur Korrektur von OCR-Fehlern in hebräischen historischen Texten basierend auf einem optimierten neuronalen Netzwerkmodell. Eine andere Technik zur automatisierten Digitalisierung von Texten ist die HTR (vgl. Sánchez et al., 2014). In handschriftlichen Texten kann die traditionelle OCR nicht angewendet werden, weil es nicht möglich ist, die einzelnen Schriftzeichen automatisiert voneinander zu trennen. HTR ist ein Teilbereich der Mustererkennung (Pattern Recognition). Hierbei ist es nicht notwendig, dass die Zeichen oder die Wörter eines handschriftlichen Textes präzise segmentiert oder isoliert werden müssen. HTR-Techniken (vgl. Sánchez et al., 2014) verwenden Methoden und Techniken der automatischen Spracherkennung (Automatic Speech Recognition). HTR wird im Projekt tranScriptorium¹ (Sánchez et al., 2014) oder bei Tools wie Transkribus² eingesetzt. Leider ist auch HTR nicht fehlerfrei und benötigt wie die OCR weitere Verbesserungen.

Sind die gewünschten Texte digitalisiert, ist es möglich NLP-Tools und -Techniken anzuwenden, denn wie Michael Piotrowski schon geschrieben hat:

In principle, everything that can be done with modern texts can also be done with historical texts: full-text search, concordancing, lemmatization, morphological and syntactic analysis, text mining, machine translation, and so on. (Piotrowski, 2012, S. 5)

Durch NLP-Methoden können große Mengen an Texten verarbeitet werden. Für die DH bedeutet das, dass sie NLP als Basis für alle komplexeren Textanalysen brauchen (vgl. Piotrowski, 2012, S. 5). Aktuell gibt es mehrere internationale Initiativen mit dem Ziel, eine gemeinsame Infrastruktur in den DH zu schaffen und dabei auch Zugang zu Tools und Korpora zur Verfügung stellen. Wichtig zu erwähnen wären hier CLARIN³ (Common Language Resources and Technology Infrastructure) und DARIAH⁴ (Digital Research Infrastructure for the Arts and

¹<http://transcriptorium.eu/>

²<https://transkribus.eu/Transkribus/>

³<https://www.clarin.eu/>

⁴<https://www.dariah.eu/>

1. Einleitung

Humanities). Die CLARIN ERIC Infrastruktur bietet beispielsweise Zugang zu 74 historischen Korpora⁵ und 24 Tools für NER⁶.

Die Anwendung von NER Standardmodellen auf schlecht erforschte Sprachen, frühe Sprachstufen, mangelhaften oder unstrukturierten Daten und benutzerspezifische Klassifikationsaufgaben kann den Bedürfnissen von interdisziplinären Feldern wie den DH meist nicht gerecht werden. Manuelle Notation von großen und komplexen Korpora von Grund auf erfordert extrem viel Zeit und Aufwand. Daher sind NLP-Methoden zur Lösung solcher Probleme in den DH ein essentieller Bestandteil der Forschung geworden. Leider kann kein automatisiertes Sprachtool hundertprozentig fehlerfreie Ergebnisse liefern, nicht mal wenn es auf Texte angewendet wird, bei denen die Merkmale exakt mit dem Tool übereinstimmen. Es ist daher nicht überraschend, dass die meisten schnellen Versuche, ein existierendes Tool auf einen geisteswissenschaftlichen Textkorpus anzuwenden, zu Enttäuschungen führen (vgl. Kuhn, 2019, S. 573). Kuhn (vgl. 2019, S. 571) ist der Meinung, dass sich die computergestützten Methodik schrittweise für hermeneutisch orientierte Forschungsfragen öffnen sollte, d.h. dass Ressourcen und Tools für die relevanten Analysen geschaffen werden müssen.

Im Anhang habe ich eine Liste mit aktuellen NER-Tools zusammengestellt, die auch für geisteswissenschaftliche Kontexte geeignet sind. Das ist insofern auch für diese Arbeit relevant, da versucht wird, Standardmodelle von zwei Tools auf historische Texte anzuwenden. Als Fallbeispiele werden Reiseberichte aus dem 17. Jahrhundert verwendet. Dabei handelt es sich um Texte der sprachgeschichtlichen Periode des Frühneuhochdeutschen (1350-1650).

⁵<https://www.clarin.eu/resource-families/historical-corpora>

⁶<https://www.clarin.eu/resource-families/tools-named-entity-recognition>

1.2. Zielsetzung und Fragestellung

Ein Ziel der Arbeit ist es, einen Einblick in den Bereich der Named Entity Recognition (NER) innerhalb der DH zu geben. In Kapitel 1 soll daher ein geschichtlicher Überblick über die Entwicklungen im Bereich der NER beschrieben und deren aktuelle Anwendungsgebiete in den DH erläutert werden.

Ein weiteres Ziel der Arbeit ist es, Möglichkeiten, Nutzen und Grenzen bezogen auf die Anwendung von NER auf deutsche Texte des 17. Jahrhunderts zu erforschen. Dabei werden anhand einer Fallstudie Standardmodelle der NER Toolkits spaCy und NLTK (Natural Language Toolkit) getestet und ein eigenes NER Modell auf Basis eines Reiseberichts aus dem 17. Jahrhundert trainiert. Die verschiedenen spaCy Modelle werden evaluiert und miteinander verglichen. Die Methode soll dabei nicht nur angewendet, sondern auch kritisch hinterfragt werden. Folgende Fragen werden dabei aufgeworfen: Welche Toolkits können für Projekte in den DH verwendet werden? Welche Annotationsmöglichkeiten für Textkorpora gibt es? Welche Probleme und Schwierigkeiten entstehen bei der Anwendung von NER auf historische Texte? Wie wird das Training eines eigenen Modells in spaCy umgesetzt? Wie kann die Qualität der Ergebnisse der NER beeinflusst bzw. verbessert werden?

Um diese Fragen beantworten zu können, sind vor allem theoretische Grundlagen nötig. In Kapitel 2 wird geklärt, was der Begriff Named Entity bedeutet. Dabei werden die in der Literatur diskutierten Kategorien angeführt und unterschiedliche bereichsspezifische Definitionen des Begriffs genannt. Außerdem werden die wichtigsten Techniken, die bei Anwendungen, die sich mit NEs beschäftigen, genauer untersucht. Zum einen wird dabei näher auf die Named Entity Recognition and Classification (NERC) eingegangen, um einen Einblick in die verschiedenen NER-Verfahren zu bekommen. Zum anderen soll auch die Disambiguierung und das Linking von Entitäten erklärt werden, da auch diese beiden Techniken eng mit der NER verwoben sind.

In Kapitel 3 werden verschiedene Methoden des maschinellen Lernens

1. Einleitung

(ML) thematisiert. Speziell wird auf Methoden eingegangen, die in Bezug auf die NER eine wichtige Rolle spielen und mit Beispielen aus dem Bereich der DH untermauert. Das Hauptaugenmerk liegt dabei auf dem überwachten Lernen, da diese Methode im praktischen Teil der Arbeit eingesetzt wird. Um ein besseres Verständnis der Funktionsweise des überwachten maschinellen Lernens zu bekommen, werden ein paar der wichtigsten Algorithmen genauer behandelt.

Kapitel 4 beschäftigt sich mit der Anwendung von NER auf historische Texte. Hierbei werden NER Systeme und Tools aus der Perspektive der DH betrachtet. Es werden verschiedenen Annotationsmöglichkeiten für Textkorpora genannt und erklärt, was Trainings-, Validierungs- und Testdaten sind. Zusätzlich wird erläutert, welche Evaluierungsmethoden auf NER-Systeme angewendet werden können und welche Probleme sich bei der NER auf historische Texte ergeben.

Kapitel 5 der Arbeit befasst sich mit den Ergebnissen des praktischen Teiles dieser Masterarbeit⁷, in dem die Anwendung deutscher NER Standardmodelle von spaCy und NLTK auf einen Reisebericht des 17. Jahrhunderts getestet wird. Zusätzlich wird ein eigenes kleines NER Modell mithilfe von spaCy trainiert. Der Workflow und die eingesetzten Tools werden dabei evaluiert und in einzelnen Schritten erläutert. Dieser Teil fokussiert sich auf wichtige Aspekte, wie die ausgewählten Trainings- und Testdaten, die entstandenen Probleme und Herausforderungen vor und während der Analyse sowie die Präsentation der Ergebnisse.

Abschließend wird zusammengefasst, wie erfolgreich die Analyse war, welche Ziele erfüllt werden konnten, welche Probleme sich ergaben und ob bzw. wie man diese lösen könnte, wo es noch Verbesserungsmöglichkeiten gibt und wo gegenwärtig noch Grenzen liegen.

⁷Daten und Jupyter Notebooks zur Dokumentation des praktischen Teiles befinden sich auf Github unter: https://github.com/jackymore/NER_historical_texts

2. Forschungsfeld Named Entity Recognition

2.1. Geschichte und Entwicklungen

Untersuchungen zur Geschichte und Entwicklung im NER-Bereich wurden bereits von mehreren Forschern durchgeführt. Die erste zusammenfassende Erhebung wurde von Nadeau und Sekine (2007) gemacht. In ihrem Artikel beschreiben sie eine Vielfalt von überwachten, unüberwachten und teilüberwachten NER-Systemen und sie fokussierten sich auf gemeinsame Merkmale in den damals aktuellen NER-Systemen. Sharnagat (2014) liefert eine aktuellere Untersuchung, die sich ebenfalls auf überwachte, unüberwachte und teilüberwachte NER-Systeme bezieht. Nouvel (2016) beschäftigt sich generell mit NEs und NER in allen Bereichen. Die neueste Erhebung wurde von Yadav und Bethard (2018) durchgeführt, welche sich auf tiefe neuronale Netzwerkarchitekturen fokussiert und diese in Kontrast mit den früheren NER-Zugängen, die auf Feature Engineering und anderen überwachten oder semi-überwachten Lernalgorithmen basieren, stellt. Da sich der wesentliche Teil dieser Arbeit mit NER beschäftigt, wird im Anschluss ein Überblick über die Geschichte und die Entwicklungen im Bereich NER mit Bezug auf die oben genannten Untersuchungen gegeben.

Der Ursprung von NER liegt in den 1980er Jahren, damals wurde NER von Computerlinguisten als Teilaufgabe der Information Extraction (IE) entwickelt. 1987 wurde die erste *Message Understanding Conference* (MUC) veranstaltet. Diese wurden vom *Naval Research And Development* (NRAD), eine Division des *Naval Ocean Systems Center* (NOSC), mit Unterstützung von DARPA, der *Defense Advanced Research Projects Agency*, veranstaltet

2. Forschungsfeld Named Entity Recognition

(Grishman & Sundheim, 1996). Innerhalb dieses Programms fanden sieben Konferenzen zwischen 1987 und 1997 statt. Der Sinn dieser Veranstaltungen lag in der Unterstützung der Forschung an automatischen Analysen von militärischen Textnachrichten (vgl. Grishman & Sundheim, 1996, S. 466). Diese Aufgabe stellte sich schnell als sehr komplex heraus und man kam auf die Idee einen Arbeitsschritt zu definieren, der NEs entdeckt und kategorisiert. Nouvel (2016) merkt an, dass die ersten Arbeiten Teil von US-Forschungsprogrammen waren, und daher grundsätzlich das Englische betreffen.

Eine der ersten Forschungsarbeiten im Bereich NER wurde von Lisa F. Rau (1991) auf der *Seventh IEEE Conference on Artificial Intelligence Applications* präsentiert. Ihre Arbeit (vgl. Jacobs & Rau, 1990) beschreibt ein System für „Conceptual Information Summarization, Organization, and Retrieval (SCISOR)“. Es verarbeitete sechs Artikel in der Minute und analysierte Namen, Datumsformate, Nummern und Kontraktionen. Außerdem konnten damit die Struktur der Texte und die Topiks der einzelnen Artikel erkannt werden. Die erste MUC, die sich eingehender mit NER beschäftigte, fand 1996 statt (Grishman & Sundheim, 1996). Hier wurde das Konzept einer „Named Entity“ erstmals eingeführt:

[...] the named entity task, which basically involves identifying the names of all the people, organizations, and geographic locations in a text. (Grishman und Sundheim, 1996, S. 467)

Dort fokussierte man sich auf Algorithmen des maschinellen Lernens und Modelle für multilinguale NER. Damals wurde festgestellt, wie wichtig es ist, Informationseinheiten automatisiert zu erkennen. Auf der MUC-7 (Chinchor & Robinson, 1998) wurden die Entitäten in folgende Kategorien eingeteilt:

- ENAMEX (Entity NAME EXpression): Person, Ort, Organisation
- TIMEX (TIME EXpression): Datum, Zeit
- NUMEX (NUMeric EXpression): Geld, Prozent, Quantität

2. Forschungsfeld Named Entity Recognition

Die Kodierung von Texten sollte folgendermaßen aussehen (Beispiele aus Chinchor und Robinson, 1998):

```
<ENAMEX TYPE="LOCATION">North and South America</ENAMEX>
<TIMEX TYPE="TIME">8:24 a.m. Chicago time</TIMEX>
the <ENAMEX TYPE="PERSON">Clinton</ENAMEX> government
```

Diese Aufgabe wurde „NERC“ genannt und führte zur ersten NE Typologie. Abbildung 2.1 zeigt die NE Typologie im Kontext der oben genannten Kennzeichnung von NEs. Sie zeigt jeweils ein Beispiel und ein Gegenbeispiel für die verschiedenen Kategorien. Die *Conference on Natural Language Learning*

Types	Example	Counter-example
ORG	<i>DARPA</i>	our university
PERS	<i>Harry Schearer</i>	St. Michael
LOC	<i>U.S.</i>	53140 Gatchell Road
MONEY	<i>19 dollars</i>	in dollars ? 19
TIME	<i>8 o'clock</i>	last night (*)
DATE	<i>the 23 July</i>	last July (*)

Abbildung 2.1.: NE Typologie im Kontext der MUC, (*) hinzugefügt auf der MUC-7 (Nouvel, 2016)

(CoNLL) basierte auf Typologien von NEs und lieferte ein annotiertes Korpus in Englisch und Deutsch und arbeitete an Zugängen, die auf überwachten und semi-überwachten Lernen basierten (vgl. Nouvel, 2016, S. 4). Ende der 1990er Jahre entstand das Programm *Automatic Content Extraction* (ACE), das bis 2008 durchgeführt wurde. In dieser Zeit wurde die NER weiter entwickelt und neu definiert. Darunter fallen die Erkennung von Entitäten, die Erkennung von Beziehungen zwischen den Entitäten und die Erkennung von Events (vgl. Nouvel, 2016, S. 5). Die NER basiert also nicht mehr nur auf Eigennamen, sondern auch auf anderen Referenztypen, wie Wortgruppen oder Wörter, die sich auf Entitäten beziehen. Damit wurde der Grundstein für viele NLP-Prozesse gelegt. Relativ früh folgten in der zweiten Hälfte der 2000er Jahre mehrere

2. Forschungsfeld Named Entity Recognition

NE-Kampagnen, die sich nicht mehr nur auf das Englische bezogen, sondern auch auf das Französische (ESTER und ESTER-2), Portugiesische (HAREM), Italienische (EVALITA) oder Deutsche (GermEval). Auch im geisteswissenschaftlichen Kontext gibt es Veranstaltungen, die sich mit computergestützter Textanalyse beschäftigen, so fokussierte man sich beispielsweise auf der *Conference on Computational Linguistics* (COLING) 2016⁸ mit Sprachtechnologie in den DH. In den letzten 20 Jahren wurde die NER rasant weiter entwickelt, sodass sie heute in den unterschiedlichsten Bereichen (Marshall, 2020) eingesetzt wird, dazu zählen beispielsweise folgende Einsatzgebiete:

- Suchmaschinen (z.B. Google): Verbesserung der Schnelligkeit und der relevanten Sucheinträge
- Kunden-Support: Verbesserung von Anfragezeiten bei Kategorisierungen von Fragen, Beschwerden durch die Filterung von Schlüsselwörtern
- Textklassifikation: Einblick in Trends von News, Blogbeiträgen oder Zeitungsartikeln
- Gesundheitswesen: Filterung von wichtigen Informationen in Pathologie- und Radiologieberichten

Laut Nouvel (2016) zeigt sich ein Trend in Richtung *Slot Filling*. *Slot Filling* ist seit 2009 Teil der Evaluationskampagne der *Text Analysis Conferences* (TACs)⁹. Das Ziel von TACs ist die Forschung in der NLP-Community zu unterstützen und eine Infrastruktur für umfassende Evaluationen von NLP Methoden zu bieten. Slot Filling wird definiert als „extracting fillers for a set of predefined relations (‘slots’) from a large corpus of text data“ (Adel et al., 2016). Es wird also versucht, in einem Textkorpus ein Set von Attributen zu finden, das zu einer bestimmten Entität gehört. Beispielsweise müssen für die Entität „Person“ folgende Informationen gefunden werden (Nouvel, 2016):

⁸<https://coling2016.anlp.jp/>

⁹<https://tac.nist.gov/>

2. Forschungsfeld Named Entity Recognition

- Namen: andere Namen der Person (Alias, falsche Namen, Bühnennamen etc.)
- Funktionen und Aktivitäten: ihre Arbeit, Tätigkeiten etc.
- Daten (Alter): Geburtsdatum, Todestag, unterschiedliche Ereignisse in ihrem Leben
- Orte: Orte, die zur ihren Lebensereignissen gehören (Geburts- oder Sterbeort)
- andere Personen, die in Beziehung zu ihr stehen: Kinder, Familienmitglieder
- andere Informationen: Schulen, Universitäten, etc.

Diese Aufgabe ist vor allem dann wichtig, wenn es in Richtung Named Entity Disambiguation (NED) und Named Entity Linking (NEL) geht, denn nur so kann man verschiedene Beziehungen und Referenzen entdecken, die die Entitäten miteinander verbinden bzw. unterscheiden. In Verbindung dazu beschäftigte sich auch das Programm der TAC 2019¹⁰ mit *Entity Discovery and Linking* (EDL). Weitere Programmpunkte waren *Streaming Multimedia Knowledge Base Population* (SM-KBP) und *Drug-Drug Interaction Extraction from Drug Labels* (DDI).

¹⁰<https://tac.nist.gov/2019/index.html>

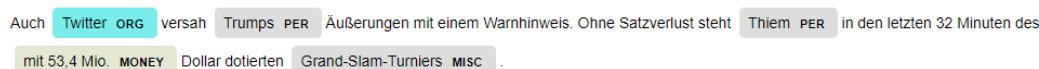
2.2. Begriffserklärung: Named Entity

Im Begriff „Named Entity“, dient das Wort „Named“ als Beschränkung und zielt nur auf die Entitäten ab, für die es ein oder mehrere starre Designatoren („rigid designators“) (vgl. Kripke, 1981) gibt. Laut Kripke verwenden wir Eigennamen wie starre Designatoren, d.h. sie referenzieren auf denselben Gegenstand in allen möglichen Welten. Starre Designatoren inkludieren Eigennamen sowie „Namen für natürliche Arten“ („natural kind terms“), wie biologische Spezies und Substanzen. Es können auch zeitliche Ausdrücke und einige numerische Bezeichnungen, wie Mengenangaben für Geld oder anderen Einheiten, miteinbezogen werden können. Ein gutes Beispiel für einen starren Designator wäre z.B.: *Wasser*, weil es sich immer auf dieselbe Substanz bezieht. Es gibt aber auch viele Entitäten, die nicht so klar erkennbar sind, wie z.B.: *im Juni*, bezieht sich auf den Monat eines undefinierten Jahres - *vergangener Juni*, *in diesem Juni*, *Juni 2020*, etc. (vgl. Nadeau & Sekine, 2007, S. 3). NE-Definitionen werden in Bezug auf solche Fälle etwas gelockert, d.h. dass auch unklare Datums- oder Zeitangaben als „Datum“ oder „Zeit“ markiert werden können.

Die Kategorien, die am stärksten untersucht wurden, sind drei spezielle Typen von Eigennamen: Namen von „Personen“, „Orten“ und „Organisationen“. Die drei Typen sind, wie oben schon erwähnt, bekannt als „ENAMEX“ (Nadeau & Sekine, 2007, S. 3). Es ist auch möglich, diese Kategorien in weitere Unterkategorien einzuteilen, so kann man „Orte“ beispielsweise in Stadt, Staat, Land, etc. aufteilen oder „Personen“ in Politiker, Geschäftsmann oder Entertainer (vgl. Fleischman & Hovy, 2002). Auf der CONLL-2002 und 2003 (E. Tjong Kim Sang 2002, E. Tjong Kim Sang & De Meulder 2003) wurde dann der Begriff einer Named Entity (NE) durch die Kategorie „Miscellaneous“ erweitert. Unter Miscellaneous (Sonstiges) fällt alles, das nicht in die Kategorien Personennamen, Orte und Organisationen passt, wie beispielsweise Events, Kunstwerke und Nationalitäten. Später wurde der Begriff noch weiter gefasst, indem auch Produkte oder Ereignisse zu sogenannten NEs zählen können (vgl.

2. Forschungsfeld Named Entity Recognition

van Hooland et al., 2015, S. 264).



Auch **Twitter** **ORG** versah **Trumps** **PER** Äußerungen mit einem Warnhinweis. Ohne Satzverlust steht **Thiem** **PER** in den letzten 32 Minuten des mit 53,4 Mio. **MONEY** Dollar dotierten **Grand-Slam-Turniers** **MISC** .

Abbildung 2.2.: Beispiel für eine automatisierte Annotation von NEs mit SpaCy

Manchmal werden die Kategorien auch einfach an bestimmte Bedürfnisse angepasst, so werden in der Biomedizin beispielsweise Zellen, Gene, Krankheiten, Medikamente etc. gefiltert (vgl. Song et al., 2015, S. 13). Auch in der Wirtschaft setzt man NER gezielt ein. Zum Beispiel kommt NER im Kundensupport zum Einsatz, um beispielsweise Feedback oder Beschwerden von Kunden, den richtigen Bereichen automatisiert zuzordnen und damit eine schneller Abwicklung der Beschwerden zu gewährleisten. Dabei handelt es sich dann um Entitäten, wie Produkte oder Firmennamen. Um die Vielfalt der Anwendungsfälle zu veranschaulichen werden im Folgenden vier Definitionen von NEs aus unterschiedlichen Fachbereichen gegeben:

A named entity is a term for which one or many strings, such as words or phrases, stands (fairly) consistently for some referent. (Stepanyan, 2020)

Étant donné un modèle applicatif et un corpus, on appelle entité nommée toute expression linguistique qui réfère à une entité unique du modèle de manière autonome dans le corpus. (Ehrmann, 2017)

WIKIPEDIA:

In information extraction, a named entity is a real-world object, such as persons, locations, organizations, products, etc., that can be denoted with a proper name. It can be abstract or have a physical existence. („Named entity - Wikipedia“, 20.08.2020)

BUSINESS ANALYTICS:

2. Forschungsfeld Named Entity Recognition

In data mining, a named entity is a phrase that clearly identifies one item from a set of other items that have similar attributes. Examples of named entities are first and last names, geographic locations, ages, addresses, phone numbers, companies and addresses. Named entities are often mined for marketing initiatives. (Rouse, n. d.)

BIOMEDIZIN:

[...]chunks of text that refer to specific entities of interest, such as gene, protein, drug and disease names. (Campos et al., 2012)

Der Begriff einer NE ist also ein weites Feld und es gibt keine exakte Definition, denn diese hängt von dem jeweiligen Anwendungsbereich und Zweck ab. Großen Einfluss auf die Definitionen einer NE hat auch das Beantworten der Fragen: *Was, Wo, Wann, Wer* und *Warum*. So rechtfertigt die NE-Hierarchie¹¹ von Sekine (siehe Abbildung 2.3) diese Erweiterung, da 7 oder 8 Klassen nicht ausreichen, um die Bedürfnisse von Anwendungen, die sich mit IE oder QA beschäftigen, zu befriedigen (vgl. Marrero et al., 2013, S. 484).

Einen möglichen Zugang um NEs allgemein zu definieren haben Marrero et al. (2013, S. 484) gefunden, dafür haben sie NEs aus verschiedenen Quellen analysiert, um herauszufinden, welche Faktoren bestimmen, ob etwas eine NE ist oder nicht. Sie legten folgende vier Kategorien fest: „grammatical category“, „rigid designator“, „unique identification“ und „domain of application“. Tabelle

Examples of NE from Sekine's hierarchy (SH) and MUC, CoNLL03, ACE and GENIA (GEN) annotated corpora or guidelines. Question marks indicate context-dependency.

Named Entity	Proper noun	Rigid desig.	Unique identifier	Example of domain/purpose	Source and NE type
Water	No	Yes	? (Sparkling or still)	Chemistry	SH: natur. obj-mineral
\$10 million	No	?	? (American dollars or Mexican pesos)	Finances	MUC-7: money
Whale	No	?	? (Orca or minke)	Biology	SH: sea animal
Bedouins	No	?	? (Specific people)	Army/ News	CoNLL03: miscellanea
Airbus A310	Yes	?	? (Specific airplane)	Army/Tech. Watch	CoNLL03: miscellanea
Batman	Yes	Yes	? (People disguised)	Movies/Costumes	ACE: person
Cytokine	Yes	?	? (Specific proteins)	Biomedicine	GEN: protein family/group
Twelve o'clock noon	No	?	? (Specific day)	News	MUC-7: time

Tabelle 2.1.: Vergleich der einzelnen NE-Kategorien von Marrero et al. (2013, S. 484)

2.1 zeigt einen Ausschnitt an NEs, die Marrero et al. (2013) für ihre Analyse

¹¹Version 7.1.0: https://nlp.cs.nyu.edu/ene/version7_1_0Beng.html

benutzt haben. Sie versuchten die einzelnen NEs jeder Kategorie zuzuordnen. Sie kamen zum Ergebnis, dass die grammatische Kategorie „Proper Noun“ kein ausreichendes Kriterium darstellt, weil die gemeinsamen Eigenschaften von Eigennamen, wie die fehlenden Inflexionen, Determinationen, lexikalische Bedeutung und die Verwendung von Großbuchstaben nicht ausreichen, um NEs zu beschreiben. Auch die Definition der „starrten Designatoren“ von Kripke ist für Marrero et al. (2013, S. 484) nicht passend, da die Identifizierung von starren Designatoren nicht eindeutig ist. Kripke nennt das Beispiel von *Richard Nixon* als *Präsident der Vereinigten Staaten von Amerika*, obwohl das im Grunde kein starrer Bezeichnungsausdruck ist, da sich das referenzierende Element alle paar Jahre ändert, und *Richard Nixon* sich auch auf viele andere Menschen mit dem gleichen Namen beziehen könnte, aber trotzdem wird *Richard Nixon* als starrer Designator akzeptiert. Marrero et al. (2013) konnten also auch hier nicht alle NEs der Kategorie „Rigid Designator“ (siehe Tabelle 2.1) zuordnen. Die dritte Kategorie ist das Konzept eines eindeutigen Identifikationsmerkmals („unique identification“). Ein Unique Identifier ist „[...]actually the referent of that to which we refer“ (Marrero et al., 2013, S. 484), d.h. dass die gleiche Entität mit der gleichen Bezeichnung verschiedene Referenten bezeichnet, so dass der eindeutige Bezeichner eigentlich der Referent dessen ist, auf den wir uns beziehen. Das setzt die vorherige Kenntnis des Referenten voraus, um festzustellen, dass er eindeutig ist. Diesem Argument folgend könnte praktisch alles eindeutig bezeichnet werden, abhängig vom Kontext oder dem Vorwissen des Empfängers, obwohl ein eindeutiger Bezeichner für einen Empfänger nicht unbedingt auch für einen anderen gilt, entweder aufgrund von fehlendem gemeinsamen Wissen oder der Mehrdeutigkeit des Kontextes. In Abbildung 2.1 sieht man, dass auch in diesem Fall nicht alle NEs der Kategorie „Unique Identifier“ zugeordnet werden konnten, da die Existenz eines einzigartigen Identifikationsmerkmals nicht in jedem Fall gesichert ist, weil es von der jeweiligen Aufgabe abhängt. Als Beispiel nennen Marrero et al. (2013) die NE *Airbus A310*. Hier scheint die Klassifikation als einzigartiges Identifikationsmerkmals eines Flugzeugtyps klar

2. Forschungsfeld Named Entity Recognition

zu sein, wohingegen die Klassifikation von Wasser (*water*) nicht wirklich ein einzigartiges Merkmal hat, außer man unterscheidet zwischen verschiedenen Typen von Wasser, wie kohlensäurehaltigem Wasser und stillem Wasser. Die letzte Kategorie „domain of application“ bezieht sich auf den Zweck und den Anwendungsbereich der NEs. Anfangs bezogen sich die Klassifikationen auf Informationen, die einem militärischen Zweck dienten (MUC), wie Zeit, Grund und Orte eines Events, oder Kategorien wie Fahrzeuge oder Waffen im Air Combat Evolution (ACE) Programm (Marrero et al., 2013, S. 484). Später wurde dann durch das Question Answering (QA) eine Erweiterung der Klassifikationen für NEs geschaffen. Marrero et al. (2013) kamen schlussendlich zum Ergebnis, dass die Definition einer NE in Bezug auf Anwendungsbereich und Zweck die einzig konstante in der Literatur, den Evaluierungsforen und Tools zu sein scheint.

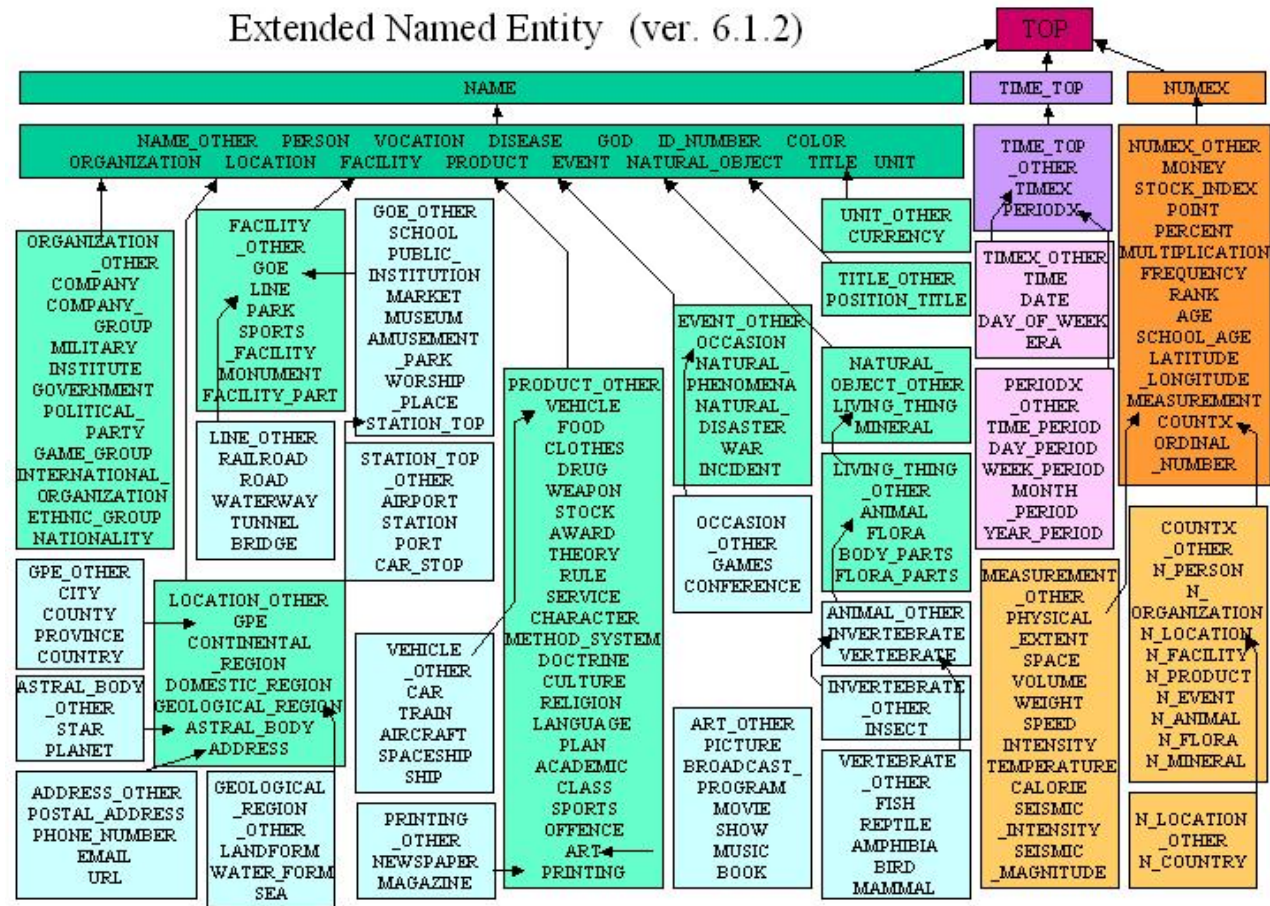


Abbildung 2.3.: NE Hierarchie von Sekine

2.3. Techniken und Verfahren

Die Erkennung und Abgrenzung von Entitäten ist für die meisten NLP-Anwendungen essentiell. Grundsätzlich gibt es zwei Schritte, die bei der NER wichtig sind (Stepanyan, 2020):

- **Recognition:** Erkennung von möglichen Strings, die eine NE bilden
- **Klassifikation:** Kategorisierung der erkannten Strings durch den Entitätstyp, auf den sie referenzieren

Ehrmann et al. (2016) nennen zusätzlich noch zwei Schritte, die bei der NER wichtig sind:

- **Disambiguation/Linking:** Verlinkung von erwähnten NEs mit einer eindeutigen Referenz
- **Beziehungsextraktion:** Entdeckung von Beziehungen zwischen den NEs

Es gibt zwei Hauptanwendungsbereiche von NEs, je nachdem ob sich die Anwendung auf referentielle Entitäten fokussiert (Indexierung und Wissensintegration) oder auf Erwähnungen (Anonymisierung und IE im Allgemeinen)(vgl. Ehrmann et al., 2016). Die einzelnen NE-Tasks können kombiniert werden, beispielsweise sind die Erkennung und Klassifizierung Teil von NER, wohingegen Erkennung und Disambiguierung zu Entity Linking gehören. Für die meisten NE-Systeme und Algorithmen sind bestimmte Ressourcen ein wichtiges Mittel, um diese Aufgaben erfolgreich bewältigen zu können. Ehrmann et al. (2016) nennen drei Haupttypen von Quellen, die es zu unterscheiden gilt:

- **Typologien:** werden dazu verwendet, einen semantischen Rahmen für die Entitäten zu definieren
- **Lexika und Wissensbasis:** liefern Informationen über NEs und werden zu Erkennung, Klassifizierung und Disambiguierung verwendet.

- **Annotierte Korpora:** werden dazu verwendet, um ein Ziel zu veranschaulichen und können als Lernbasis oder als Referenzpunkt für Evaluierungszwecke verwendet werden.

2.3.1. Named Entity Recognition und Classification

Bei „Recognition“ handelt es sich, wie in Kapitel 2 schon beschrieben wurde, um das automatisierte Erkennen von Elementen oder Entitäten in einem Text, wie beispielsweise Personen, Orte oder Organisationen. Das Problem der Erkennung von Entitäten, kann als Segmentierungsaufgabe (vgl. Stepanyan, 2020, S. 2) beschrieben werden, hierbei muss das Modell versuchen die möglichen Entitäten aus einer Spanne von Tokens zu lokalisieren. Danach muss eine Ontologie gewählt werden, um die segmentierten Strings zu klassifizieren.

NER-Verfahren

Im Laufe der Zeit haben sich verschiedene NER-Verfahren etabliert. So klassifizieren Li et al. (2018) drei traditionelle NER Zugänge: regelbasierte Verfahren, unüberwachtes Lernen und merkmalsbasiertes überwachtes Lernen. Wohingegen Al-Moslmi et al. (2020) die letzten beiden Verfahren in die gemeinsame Kategorie „Learning-based approaches“ zusammenfassen und als dritte Kategorie „Feature-inferring Neural Network“ nennen (vgl. Abbildung 2.4). Nachfolgend werde ich die drei NER-Verfahren auf Basis von Al-Moslmi et al. (2020) kurz vorstellen.

Regelbasierte Verfahren (rule-based approaches) Frühere NERC-Systeme basierten fast ausschließlich auf wörterbuch- und regelbasierten Systemen (Nadeau und Sekine, 2007; Sharnagat, 2014). Hierbei werden Eigenschaften im Zuge des Segmentierungs- und Klassifikationsprozesses per Hand erstellt. Der Precision-Wert dieser Modelle ist generell höher im Vergleich zu anderen Zugängen, da sie auf Lexika und bereichsspezifischem Wissen basieren. Der Nachteil

2. Forschungsfeld Named Entity Recognition

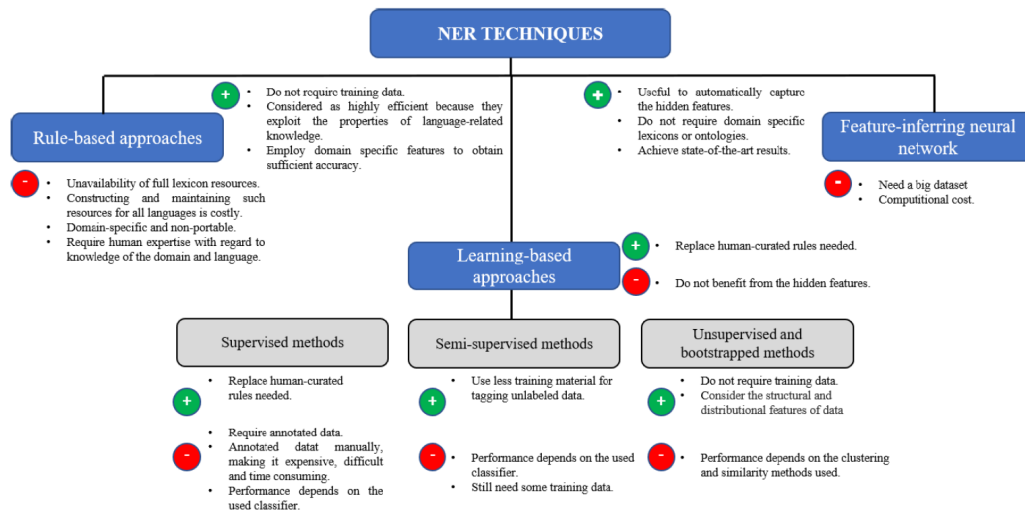


Abbildung 2.4.: NER-Zugänge (Al-Moslmi et al., 2020, S. 3)

liegt darin, dass diese Modelle einen schlechten Recall-Wert auswerfen und sie benötigen Monate an Arbeit von erfahrenen Computerlinguisten, da sie vom jeweiligen Anwendungsbereich abhängen (vgl. Stepanyan, 2020, S. 2). Oft sind auch die Quellen der Lexika unzugänglich und die Wartung solcher Ressourcen ist für die meisten Sprachen sehr teuer (vgl. Al-Moslmi et al., 2020, S. 3).

Lernbasierte Verfahren (learning-based approaches) Normalerweise werden statistische Zugänge und Methoden des maschinellen Lernens (ML) für die NER verwendet. Unter dem Begriff „maschinelles Lernen“ versteht man, dass „Computer anhand gegebener Beispiele selbstständig Zusammenhänge und Muster erkennen und insofern ‚lernen‘ können“ (Schöch, 2017, S. 289). Es kann zwischen überwachtem, unüberwachtem und teilüberwachtem Lernen unterschieden werden. Beim überwachten Lernen wird das System so trainiert, dass es NEs anhand spezifischer Eigenschaften erkennt. Teilüberwachte Lernalgorithmen werden aus einer Kombination von annotierten und nicht annotierten Daten trainiert. Support Vector Machines (SVM), Hidden Mar-

kov Models (HMM), Conditional Random Fields (CRF) und Decision Trees sind die üblichen Methoden dieser Kategorie (Sharnagat, 2014; Etzioni et al., 2005; S. Zhang und Elhadad, 2013). Die NER-Genauigkeit wird manchmal durch den verwendeten Klassifikator eingeschränkt (Al-Moslmi et al., 2020, S. 3). Wenn beispielsweise HMM und SVM verwendet werden, werden die Abhängigkeiten zwischen den Wörtern nicht berücksichtigt. Unüberwachte und bootstrapped Methoden sind automatisierter, benötigen aber einen sehr kleinen Trainingsdatensatz (vgl. Al-Moslmi et al., 2020, S. 3).

Feature-inferring Neural-Network Verfahren Neuronale Netzwerke (NN) basieren, wie lernbasierte Verfahren, auf maschinellem Lernen. NN-Verfahren unterscheiden sich von den vorhergehenden Zugängen durch automatische erschließende Eigenschaften anhand von Deep Learning (DL) (vgl. Al-Moslmi et al., 2020, S. 4). DL kann folgendermaßen definiert werden:

Deep Learning ist eine spezielle Methode der Informationsverarbeitung und ein Teilbereich des Machine Learnings. Deep Learning nutzt neuronale Netze, um große Datensätze zu analysieren. (News Center Microsoft Deutschland, 2020)

Laut aktuellen Forschungsberichten übertreffen DL-Verfahren frühere Methode, da sie keine Ontologien oder bereichsspezifische Lexika benötigen und daher auch nicht von einem bestimmten Anwendungsbereich abhängig sind (Yadav und Bethard, 2018; Goyal et al., 2018; Li et al., 2018). Der Nachteil dieses Verfahrens liegt darin, dass große Datenmengen benötigt werden, um starke und solide Modelle zu bauen (vgl. Al-Moslmi et al., 2020, S. 4).

Die meisten modernen NER-Systeme verwenden eine Variation der folgenden Algorithmen (vgl. Stepanyan, 2020, S. 2):

1. Hidden Markov Model (HMM)
2. Conditional Random Fields (CRF)

3. Neural Networks (inkl. Convolution und Recurrence)

Jede dieser Methoden zeigt vielversprechende und gute Ergebnisse. Systeme wie SpaCy oder Stanford CoreNLP verwenden eine Mischung aus HMM, CRF und Neuronalen Netzwerken. Da ich in dieser Arbeit mit SpaCy arbeite, möchte ich im Kapitel 3.1.3 näher auf diese drei genannten Methoden eingehen.

2.3.2. Disambiguation und Linking

Named Entity Disambiguierung (NED) und Named Entity Linking (NEL) sind stark miteinander verwoben, weil eine mehrdeutige Entität zuerst disambiguiert werden muss, bevor sie verlinkt werden kann. Außerdem ist eine einzigartige Kennung, also ein Internationalized Resource Identifier (IRI), ein guter Weg, um das Ergebnis der Disambiguierung zu präsentieren. Manchmal werden auch beide Begriffe in der NLP Community synonym gebraucht (vgl. W. Shen et al., 2015, S. 2). Daher werde ich NED und NEL hier gemeinsam diskutieren.

Named Entity Linking oder Entity Linking (EL) ist eine Teilaufgabe der Named Entity Recognition und Disambiguation (NERD). Unter NEL versteht man die Aufgabe, mehrdeutige Erwähnungen in einem Text den entsprechenden Entitäten einer Knowledge Base (KB) zuzuordnen (Inan & Dikenelli, 2018). Segmente in einem Text, die auf eine NE referieren, werden oft als NE-Erwähnungen (entity mentions) (Frontini et al., 2015) bezeichnet. Eine mögliche Definition für NEL nennen Dredze et al. (2010):

[...]matching a textual entity mention, possibly identified by a named entity recognizer, to a KB entry, such as a Wikipedia page that is a canonical entry for that entity.

Eine KB ist das erste Bauteil, das man für jedes EL-System benötigt. Es gibt verschiedene KBs für verschiedene Arten von Informationen über Entitäten. C. M. Phan (2019) unterscheidet hier zwischen allgemeine und kontextabhängige KBs. Allgemeine KBs decken die meisten bekannten Entitäten ab, die

beispielsweise in Zeitungsartikeln oder Social Media Texten verwendet werden. Wikipedia ist die bekannteste KB für EL, wenn es darum geht moderne Texte zu verarbeiten. Weitere bekannte KBs sind DBpedia¹² und YAGO¹³. Kontextabhängige KBs beinhalten Entitäten eines bestimmten Typs, wie Krankheiten, Gene, historische Persönlichkeiten oder Autoren. Eine bekannte KB in den DH im deutschsprachigen Raum ist beispielsweise die DNB (Deutsche Nationalbibliothek)¹⁴.

Ein EL-System benötigt also eine KB, die ein Set an Entitäten E enthält und eine Textsammlung, in der ein Set an NE-Erwähnungen M im Voraus identifiziert wurde. Das Ziel von EL ist, jede textuelle NE-Erwähnung $m \in M$ ihrer entsprechenden Entität $e \in E$ in der KB zuzuordnen (W. Shen et al., 2015). Eine NE-Erwähnung m ist eine Tokensequenz in einem Text, die potentiell auf eine NE referiert und schon im Voraus identifiziert wurde. W. Shen et al. (2015, S. 2) sprechen auch von Erwähnungen in Texten, die keine entsprechende Entität in der vorhandenen KB haben. Diese NEs werden als „unlinkable mentions“ definiert und mit NIL (Not-in-list) gekennzeichnet. Für die semantische Annotation von Texten, ist es wichtig, die NE-Erwähnungen mit einem einzigartigen Link zu deren Referenten anzureichern. Durch diesen Link wird auf eine externe Quelle gezeigt. Dafür verwendet man beispielsweise einen URI. Das Ergebnis von EL ist ein wichtiger Schritt in der Beziehungsextraktion (relation extraction), Link-Prediction und Knowledge-Graph Vervollständigung. Die wesentliche Herausforderung ist aber die Disambiguierung der Entitäten. In der Computerlinguistik versteht man unter Named Entity Disambiguation (NED)

[...] the problem of mapping ambiguous entity mentions (or just mentions for short) occurring in natural-language texts onto a set of known target entities [...] (Dat Ba Nguyen et al., 2014, S. 1)

¹²<https://wiki.dbpedia.org/>

¹³<https://yago-knowledge.org/>

¹⁴<https://portal.dnb.de/>

NERD-Algorithmen erkennen Entitäten in Texten automatisch und teilen diese einer Klasse zu. Dann wird der Entität durch das Linking eine einzigartige Kennzeichnung verliehen. Die Verlinkung ist deshalb so wichtig, weil dieselbe Erwähnung unterschiedliche Entitäten in unterschiedlichen Kontexten repräsentieren kann und zur selben Zeit kann eine Entität auch in verschiedenen Formen im Text vorkommen (vgl. Frontini et al., 2015, S. 77). Es gibt drei große Hauptprobleme, die beim EL auftreten:

Namensvariationen. Es kommt häufig vor, dass eine Entität mehrere Formen von Erwähnungen aufweist, wie Abkürzungen (Digital Humanities vs. DH), verkürzte Formen (Pfalzgraf Ludwig vs. Pfalzgraf Philipp Ludwig), alternative Schreibweisen (Pfalzgraf vs. pfalzgraff) und Aliases (Joanne K. Rowling vs. Robert Galbraith). EL sollte eine Entität auch dann finden, wenn die Erwähnung sich ändert.

Ambiguität. Eine einzige Erwähnung, wie beispielsweise *Paris*, kann mit verschiedenen Einträge in einer KB übereinstimmen, da viele NEs wie Städte, Personen oder Organisationen mehrdeutig sind. Oft haben sich auch Ortsbezeichnungen geändert. Durch Pest, Türkenzüge, Kriege oder aus wirtschaftlichen Gründen wurden oft Orte ausgelöscht oder an anderer Stelle neu aufgebaut, auch manchmal unter gleichem Namen. Zusätzlich kommt das Problem einer unterschiedlichen Schreibweise der Orte hinzu. Aber auch Orte in Homers Erzählungen, wie der Eingang zu Hades, Circes Insel von Aea oder die Höhle von Polyphemus haben einen wichtigen geschichtlichen und literarischen Einfluss. Geografische Informationssysteme (GIS) werden immer öfter in der Geschichtswissenschaft verwendet und trotzdem gab es wenig Diskussionen über die Konstruktion und Entwicklung von digitalen Ortsverzeichnissen und deren wichtigen Rolle im Linked Data Ökosystem, das sich rund um die ortsbezogenen Geisteswissenschaften entwickelt (Horne, 2020, S. 37).

Das Fehlen eines Eintrags in einer KB. Bei der Verarbeitung von großen Textmengen kommt es häufig vor, dass viele Entitäten nicht in der KB auftau-

2. Forschungsfeld Named Entity Recognition

chen (NIL).

Zum Thema NED bringen W. Shen et al. (2015) folgendes Beispiel: In modernen Texten muss beispielsweise zwischen dem Footballspieler *Michael W. Jordan* und dem Wissenschaftler *Michael I. Jordon* unterschieden werden (vgl. Abbildung 2.5). Beide Begriffe bezeichnen eine Person mit dem selben Namen, aber repräsentieren vollkommen unterschiedliche Entitäten. In historischen

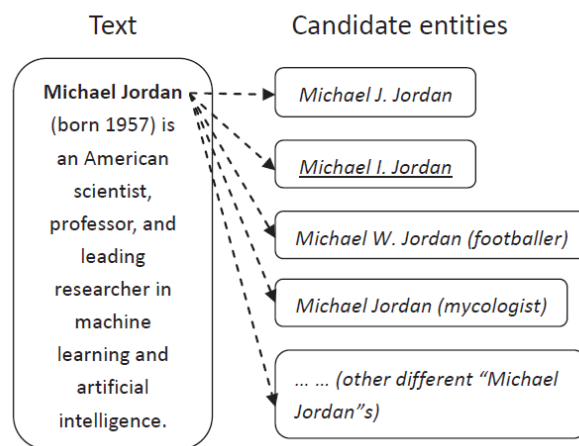


Abbildung 2.5.: Die EL Task. Die Erwähnung ist fett dargestellt; die korrekte dazugehörige Entität ist unterstrichen. (W. Shen et al., 2015, S. 2)

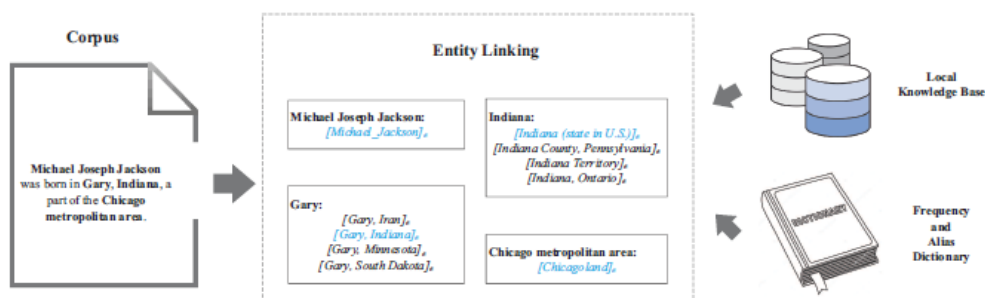


Abbildung 2.6.: Verlinkung eines Korpus mit einem lokalen KG (Hu et al., 2019, S. 2)

2. Forschungsfeld Named Entity Recognition

Texten ist es meist aufgrund der nicht-standardisierten Schreibweise viel schwieriger Entitäten zu unterscheiden oder Entitäten, die auf das gleiche referieren, zu erkennen. So könnte in einem historischen Text des 17. Jhds., die Erwähnung *pfalzgraff* sich auf *pfalzgraff Wolfgang* oder auf *pfalzgraff Philip Ludwig* beziehen. Es könnte aber auch mit *Pfgf. Philipp Ludwig von Pfalz-Neuburg* auf *Pfgf. Philipp*, *Pfgf.*, oder *Philipp* referiert werden. Um also automatisch auf alle Abschnitte im Text, wo *pfalzgraff* vorkommt, referieren zu können, ist es notwendig, nicht nur alle diese Erwähnungen als eine NE auszuzeichnen, sondern auch jede mit einem einzigartigen Schlüssel zu versehen, damit man eine Person, in diesem Fall einen bestimmten Pfalzgrafen, von anderen Pfalzgrafen im Text unterscheiden kann. Solch eine einzigartige Kennung könnte für *Pfalzgraf Philipp Ludwig von Pfalz-Neuburg* der folgende Link darstellen:

<http://d-nb.info/gnd/118593951>

Je nachdem für welche Annotationsrichtlinien man sich entscheidet, wird diese Kennung zur Annotation hinzugefügt. Sehr oft werden die Konventionen der *Text Encoding Initiative* (TEI)¹⁵ in den DH verwendet. Damit würde die Annotation so aussehen:

```
<persName ref="http://d-nb.info/gnd/118593951">  
pfalzgraff Philip Ludwig</persName>
```

Da die Annotation der Daten ein wichtiger Bestandteil der NER ist, werden verschiedenen Annotationsmöglichkeiten in Kapitel 4.1 genauer behandelt.

Eines der Hauptprobleme in den DH ist, dass die erwähnten Personen meist zu Individuen führen, die nicht in den allgemeinen Ontologien wie Yago¹⁶ oder DBpedia zu finden sind. Meist findet man sie, wie auch im Fall von Pfalzgraf Philipp Ludwig, in anderen KBs, wie bibliographisch verlinkte Datenrepositorien (Deutsche Nationalbibliothek (DNB)). Aber auch das Linking benötigt Zugang

¹⁵<http://www.tei-c.org/index.xml>

¹⁶<https://yago-knowledge.org/>

zu ontologischem Wissen, da die Wahl zwischen zwei Entitäten mit dem selben Namen auch Hintergrundwissen benötigt. Daher sind auch KBs wie DBpedia eine wichtige Quelle für das allgemeine Weltwissen (vgl. Frontini et al., 2015, S. 78).

NED Verfahren

Al-Moslmi et al. (2020) nennen drei Kategorien für NED Verfahren: Traditionelle NED Verfahren, Neuronale Netzwerke und Joint NER und NED. Die Einteilungen basieren darauf, wie diese Verfahren Entitäten ranken.

Traditionelle NED Verfahren Traditionelle NED Verfahren verwenden Eigenschaften, die von Hand geschaffen wurden, um die Ähnlichkeit zwischen den Erwähnungen und den dazugehörigen Entitäten zu berechnen. Al-Moslmi et al. (2020) teilen diese Verfahren weiter auf in unabhängige und kollektive Verfahren. Unabhängige Verfahren (Fang et al., 2016; Daiber et al., 2013) verwenden semantische Ähnlichkeitstechniken, um die Entitäten ausschließlich anhand ihrer lexikalischen Ähnlichkeit und/oder ihrer empirischen Kookkurrenz mit den Erwähnungen zu bewerten. Bei dieser Methode wird jede Erwähnung einer Entität unabhängig voneinander disambiguiert. Die Merkmale werden per Hand erstellt. Je komplexer die Texte werden, desto geringer ist die Genauigkeit dieser Methoden, da per Hand geschaffene Eigenschaften dazu tendieren, wenig textuelle Information zu beinhalten. Auch scheitern sie daran, Interaktionen zwischen NE-Erwähnungen im gleichen Dokument abzufangen.

Kollektive Verfahren basieren auch auf semantischer Ähnlichkeit. Sie berechnen aber, dass Erwähnungen im gleichen Teil eines Textes in den meisten Fällen auch zum gleichen Topik gehören. Daher sind kookkurrierende (zwei lexikalische Einheiten treten gemeinsam in einer übergeordneten Einheit auf) Entitäten oft semantisch verwandt (Zhu & Iglesias, 2018). Kollektive Verfahren sind robuster als die unabhängigen, jedoch steigen die Kosten für die Rechen-

leistung stark an, je mehr der NE-Erwähnungen in den Texten vorkommen und je länger die Dokumente werden (vgl. Al-Moslmi et al., 2020, S. 5).

Neuronale Netzwerke Neuronale Netzwerk-Methoden (NN) wurden in den letzten Jahren immer häufiger verwendet und erreichten ähnlich gute Resultate wie traditionelle Verfahren (Hu et al., 2019; M. C. Phan et al., 2017; Eshel et al., 2017; Sherzod Hakimov et al., 2016). Es gibt NN-Verfahren, die Wörter in Vektorräumen oder ähnlichen Modellen abbilden. Andere verwenden Deep Neural Network Methoden, um die semantischen Eigenschaften automatisch zu lernen. Der Großteil der bisherigen NN-Methoden geben allen NE-Erwähnungen im gleichen Kontext eine gleiche Gewichtung, was für die meisten Anwendungsfälle ausreichend ist (Wang et al., 2019; M. C. Phan et al., 2017; Nie et al., 2018). Die Genauigkeit von aktuellen NED-Systemen ist noch lange nicht perfekt, vor allem bei kurzen Texten, die wenig Kontext haben, wie Tweets, Suchanfragen, usw. (vgl. Zhu & Iglesias, 2018, S. 9).

Für die DH sind NN oder Deep Learning Methoden nicht immer geeignet, da für NN sehr viele Daten benötigt werden, damit die Maschine selbstständig Muster erlernen kann. Hier scheitern manche historische Disziplinen schon oft daran, dass nicht genügend digitales Datenmaterial vorhanden ist. Es gibt zwar Unmengen an edierten Texten in Druck, jedoch sind diese meist nicht in digitaler Form verfügbar oder müssen erst aufbereitet werden, d.h. Fehler, die beim Scannen oder OCR entstehen, müssen entsprechend korrigiert werden. Wenn jedoch genügend Datenmaterial zur Verfügung steht bieten NN neue Möglichkeiten, wie das beispielsweise im Cultural Heritage Bereich der Fall ist. So kann die Produktion von Kunstwerken automatisiert werden (Deltorn, 2017). NN besitzen den Vorteil, dass sie die Fähigkeit besitzen, aus einer Trainingsphase ein generatives Modell abzuleiten, aus dem neue Artefakte erzeugt werden können. Neue Gemälde oder Musik im Stil berühmter Künstler können dabei erzeugt werden oder zu neuen Kunstwerken kombiniert werden.

Joint NER und NED Bisherige Studien haben NER und NED als zwei separate Schritte behandelt. Im NER-Schritt wurden alle NE-Erwähnungen im Text entdeckt und markiert. Unabhängig davon wurden im NED-Schritt die Erwähnungen zu Entitäten in der KB anhand textueller Ähnlichkeit und semantischer Kookurrenz der ausgewählten Entitäten zugeordnet. Die Konsequenz daraus ist, dass beim NED nicht alle Informationen, die von der NER zur Verfügung gestellt werden, benutzt werden können (Luo et al., 2015). Der Nachteil liegt darin, dass Informationen, wie Entitätstypen, die für beide Aufgaben wichtig wären, nicht geteilt werden oder schwache NER Precision, die NED Genauigkeit verschlechtert (Nguyen et al., 2016). Daher sollte NERD mit NER und NED gemeinsam agieren. Bei Systemen, wie dem JERL (Joint Entity Recognition and Linking) Modell (Luo et al., 2015) sind NER und NED voneinander abhängig. Laut Luo et al. (2015) war JERL das erste Modell, dass NER und Linking Aufgaben gemeinsam optimiert hatte. TwitterNEED (Habib & van Keulen, 2016) unterstützt NERD von kurzen informativen Tweets und liefert bessere Resultate als DBpedia Spotlight, Stanford NER und AIDA. Ein weiteres Joint NERD System ist J-NERD (Nguyen et al., 2016). J-NERD benutzt ein probabilistisches Graphen-Modell und erfasst Typen von NE-Erwähnungen, Spannen und die Verlinkung der Erwähnungen zu deren Entitäten in einer KB.

NEL Verfahren

NEL fügt jeder NE-Erwähnung in einem Text eine IRI ihrer zugehörigen Entität in einer KB hinzu. Es gibt verschiedene Ansätze, wie NEL-Methoden eingeteilt werden können. Frühe NEL Zugänge können in drei Haupttypen unterteilt werden: Entity-by-Entity Methoden, Maschinelles Lernen und Methoden des Collective-Linking. Durch den aktuellen Aufschwung von Neuralen Netzwerken ist es laut Al-Moslmi et al. (2020) besser die NEL Verfahren neu zu unterteilen, und zwar in unabhängige oder feature-engineering Methoden, Kollektive Methoden und NN-basierte Methoden. Frontini et al. (2015) wiederum un-

terteilen NEL in zwei Gruppen: Eine Gruppe benutzt Textähnlichkeit (text similarity), die andere graphbasierte Methoden. Bei beiden Gruppen handelt es sich um unüberwachte Methoden, d.h. sie benötigen für das Training keine vorannotierte Korpora (vgl. Frontini et al., 2015, S. 78). Im folgenden wird näher auf den Ansatz von Frontini et al. (2015) eingegangen, da sie einen NEL-Algorithmus (Brando et al., 2016) für den geisteswissenschaftlichen Kontext entwickelt haben.

Unabhängig vom Typ, besteht ein typisches EL-System aus folgenden drei Modulen (W. Shen et al., 2015):

1. Candidate Entity Generation

In diesem Modul zielt das EL-System darauf ab, für jede NE-Erwähnung $m \in M$ die irrelevanten Entitäten in der KB herauszufiltern und ein Set an passenden Entitäten abzurufen E_m .

2. Candidate Entity Ranking

In den meisten Fällen ist die Größe der Sets an passenden Entitäten E_m größer als eins. Hier gibt es verschiedene Möglichkeiten um die Entitäten in einem passenden Set E_m zu ranken und die Entität $e \in E_m$ herauszufiltern, die am wahrscheinlichsten zu der NE-Erwähnung m passt.

3. Unlinkable Mention Prediction

Dieses Modul versucht mit dem Problem von unlinkbaren NE-Erwähnungen umzugehen. Wenn eine NE-Erwähnung m nicht zuweisbar ist, soll NIL ausgegeben werden.

Schnelle NED Methoden (z.B.: TagMe¹⁷, DBpedia Spotlight¹⁸) verwenden sehr einfache kontextuelle Eigenschaften, wie beispielsweise nur charakteristische Wörter. Diese zählen zur ersten Gruppe. Sie vermeiden es mit Datenbanken zu interagieren und verwenden stattdessen eine individuell angepasste speicherinterne Datenstruktur. Diese Methoden sind sehr effizient bei

¹⁷<https://tagme.d4science.org/tagme/>

¹⁸<https://www.dbpedia-spotlight.org/>

2. Forschungsfeld Named Entity Recognition

unkomplizierten Texten mit bekannten Entitäten, das Ergebnis wird jedoch bei sehr komplexen Einträgen mit mehrdeutigen Namen und Erwähnungen von Entitäten ungenauer (vgl. Dat Ba Nguyen et al., 2014). Leider können sie nur zwischen Quellen wie DBpedia, deren Beschreibungen in Form von unstrukturierten Texten bestehen, verlinken. Andere KBs bieten keine textuelle Beschreibung für ihre Einträge an. Das ist der Fall bei bibliographischen Datenbanken, die das perfekte Linking für die Erwähnung von Autoren bilden (vgl. Frontini et al., 2015, S. 79).

Graphbasierte Methoden basieren auf formalisiertem Wissen, das in Form von Graphen beschrieben wird und anhand einer KB aufgebaut ist. Ein Knowledge Graph (KG) repräsentiert semantische Daten als Triple, die aus Subjekt, Prädikat und Objekt zusammengesetzt werden. Das können entweder IRIs, leere Knoten oder Literale sein. Die IRIs können aus Vokabluarien oder Ontologien in der Linked Open Data (LOD) Cloud entnommen werden. Die literalen Werte, die als Objekte verwendet werden, können durch gut definierte Datentypen, wie beispielsweise Datentypen, die durch eine XML Schemadefinition definiert wurden (XSD), repräsentiert werden (vgl. Al-Moslmi et al., 2020, S. 2). Der KG versucht die Semantik von Entitäten und ihren Beziehungen präzise zu beschreiben und die Beschreibungen für weitere Informationen auf semantische LOD Repositorien zu verlinken. Abbildung 2.7 zeigt den Weg von NL-Dokumenten durch NLP zu KGs. Anhand von Graphen kann man



Abbildung 2.7.: Von natürlicher Sprache (NL) zum Knowledge Graphen (KG) (Al-Moslmi et al., 2020, S. 2)

zumindest zum Teil den Entscheidungsprozess von Menschen, die Entitäten in

2. Forschungsfeld Named Entity Recognition

einem Text unterscheiden, nachbilden (vgl. Frontini et al., 2015, S. 79). Wenn man einen Text liest und auf den Namen „Paul“ stößt, könnte man entscheiden, dass „Paul“ auf den Schriftsteller „Jean Paul“ referiert und nicht auf den Schauspieler „Paul Walker“, da der Name im selben Kontext erscheint wie „Hölderlin“ und „Schiller“. Auf diese Art bauen solche Algorithmen einen Graphen aus Kandidaten, die für jeden Referenten in einem bestimmten Kontext möglich sind. Sie verwenden verwandte Positionen jedes Referenten innerhalb des Graphen, um den korrekten Referenten für jede Erwähnung auszuwählen. Der Graph wird für einen ausgewählten Kontext gebaut, wie beispielsweise für einen Absatz, der möglicherweise mehr als eine Erwähnung beinhaltet (vgl. Frontini et al., 2015, S. 79).

Um NEs aus Texten zu extrahieren und diese als Knoten in KGs zu repräsentieren, werden drei Hauptschritte benötigt: NER, NED und NEL. Abbildung 2.8 zeigt die Abfolge der Tasks von NL zu KGs. Außerdem sieht man die aktuellen End-to-End Zugänge (zero-shot), die alle drei Tasks zusammenfassen, indem sie Deep Neural Networks verwenden (vgl. Al-Moslmi et al., 2020, S. 2). Komplexere Methoden, die auf reichen kontextuellen Eigenschaften, wie

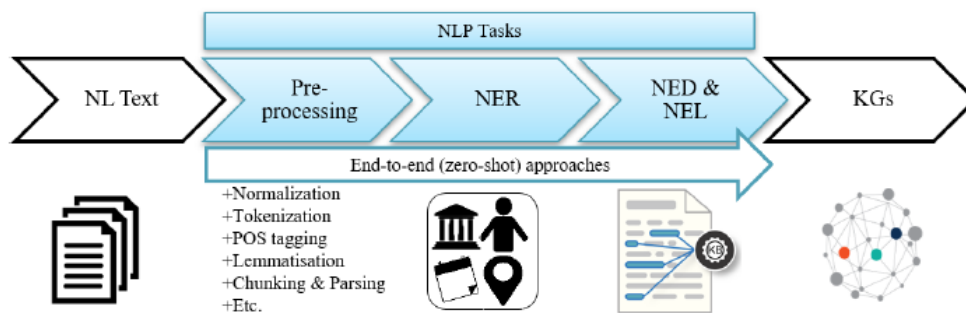


Abbildung 2.8.: NLP Tasks

Schlüsselphrasen oder auf Joint-Inference Algorithmen zurückgreifen, sind aber meist langsamer und stoßen auf Einschränkungen der Skalierbarkeit (vgl. Dat Ba Nguyen et al., 2014). Die genauesten Resultate für komplexe Texte liefern

meist Zugänge, die auf gemeinsamen Lernmodellen basieren (Dat Ba Nguyen et al., 2014). Diese Modelle führen in einem Zug eine gemeinsame Ausarbeitung aller Erwähnungen auf ihren übereinstimmenden Zielentitäten aus, was jedoch sehr viel Rechenleistung benötigt.

2.4. Anwendungsmöglichkeiten

Auch in den DH erfreut sich die Anwendung von NER-Methoden immer größerer Beliebtheit. Im Folgenden werden ein paar Projekte im DH-Bereich vorgestellt, die NER-Methoden und Tools verwenden.

Das Projekt „NERDPool - Data Pool for Named Entity Recognition“¹⁹ ist ein laufendes Projekt, das sich mit NER für deutschsprachige historische Korpora beschäftigt, die am ZIM (Zentrum für Informationsmodellierung, ACDH (Austrian Centre For Digital Humanities and Cultural Heritage) und UIBK (Universität Innsbruck) in den letzten Jahren erstellt wurden. Dabei werden NEs in historischen Texten semi-automatisch identifiziert und annotiert, um einen Gold-Standard für das Training von NER-Modellen zu erstellen. Durch die Bereitstellung von Gold-Standards, NER-Modellen, Dokumentations- und Trainingsmaterial für das Training von historischen deutschsprachigen Texten, wird NERDPool einen wichtigen Beitrag für die DH-Community liefern.

Ruokolainen und Kettunen (2020) evaluierten in ihrem Projekt Daten aus einer digitalisierten Sammlung von historischen finnischen Zeitungen aus den Jahren 1771 bis 1929. Die Sammlung stammt aus der digitalisierten Finnischen Nationalbibliothek (Digi) und enthält 7,61 Millionen Seiten auf Schwedisch und Finnisch. Das Ziel des Projekts war die Verbesserung der Suchfunktion der Zeitungsartikel und Journale in Digi durch die Verwendung von NER. Für ihr Projekt verwendeten sie den Stanford Named Entity Recognizer und das LSTM-CRF NER Modell. Damit sollen neue Möglichkeiten zur Strukturierung,

¹⁹<https://digital-humanities.at/en/dha/s-project/nerdpool-data-pool-named-entity-recognition>

des Zugangs und die Anreicherung von Informationen entstehen. Verschiedene Namenstypen, vor allem Personen- und Ortsnamen werden sehr häufig als Suchbegriffe in den verschiedenen Zeitungssammlungen verwendet. Der Stanford NER lieferte dabei sehr gute Resultate. Mit den Ground Truth Daten erreichten sie einen F-Score von 0,89 bei Orten, 0,84 bei Personen und 0,60 bei Organisationen.

Sintayehu und Lehal (2020) erforschten einen graphbasierten Label Propagation Algorithmus für die Amharische Sprache. Einen annotierten Trainingsdatensatz für Amharisch gibt es nicht, dennoch existiert eine Vielzahl an nicht annotierten Daten. Um dieses NER Problem der amharischen Sprache zu lösen, wurde ein Label Propagation (LP) Algorithmus entwickelt, der vorher unannotierte Datenpunkte mit Annotationen markiert. Ein weiteres Ziel war einen Vergleich zwischen einem Erwartungs-Maximierungs-Algorithmus (EM)²⁰ und dem LP Algorithmus aufzustellen. Das Experiment zeigte, dass der NER-basierte LP Algorithmus bessere Ergebnisse lieferte als die Verwendung von wenig annotierten Daten bei der Methode der EM.

Für die Anwendung von NER im Bereich der digitalen Museen, stellten M. Zhang et al. (2020) das teilüberwachte Deep Learning Modell SCRNER (Semi-supervised model for Cultural Relics' Named Entity Recognition) vor, das mit wenig annotierten Daten und teilannotierten Daten trainiert wurde. Digitale Museen haben die Art, wie sich Menschen kulturelles Wissen aneignen, stark verändert. Online Museen generieren sehr viele Daten über kulturelle Relikte. In den letzten Jahren konnten Forscher durch den Einsatz von Deep Learning Modellen NER implementieren. Leider fehlt es oft an annotierten Daten im Bereich der Kulturdenkmäler und das macht es für Deep Learning Modelle schwer gute Ergebnisse zu erzielen, da diese auf annotierten Daten

²⁰Ein EM ist ein Algorithmus der mathematischen Statistik, der mit einem zufällig gewählten Modell startet und abwechselnd die Zuordnung der Daten zu Teilen des Modells und die Parameter des Modells an die neueste Zuordnung verbessert (vgl. „EM-Algorithmus – Wikipedia“, 09.11.2020).

angewiesen sind. Aus diesem Grund entwickelten M. Zhang et al. (2020) das SCRNER Modell. Die Forscher hatten das Ziel verschiedene Entitäten von Kulturdenkmälern, wie deren Namen, die Dynastie der Denkmäler (Zeit der Herstellung/Erbauung), Ausgrabungsorte und Museumssammlungen zu erkennen. Die Ergebnisse der Experimente zeigen, dass das Modell, das mit wenig annotierten Daten trainiert wurde, die besseren Ergebnisse bei Accuracy und dem F_1 -Score lieferte als die Baseline-Ansätze.

Riedl und Padó (2018) untersuchten wie man ein NER-Modell für deutschsprachige Texte bauen kann, das sowohl auf aktuelle als auch auf historische Texte anwendbar ist. Sie stellten dafür zwei der am besten funktionierenden NER-Modell gegenüber: das linear-chain CRF (Conditional Random Fields) System und ein auf BiLSTM (Bidirectional Long Short-Term Memory Networks) basierendes System. Getestet wurden die Modelle anhand zwei großen aktuellen Korpora und zwei kleinen historischen Korpora.

Im Projekt „Semantic Blumenbach“ (Wettlaufer et al., 2015) lag das Ziel darin, die Verbindungen zwischen Johann Friedrich Blumenbachs (1752-1840) Texten über Naturgeschichte und den physikalischen Objekten, die er studiert und gesammelt hatte, zu erkennen und sichtbar zu machen. Für das Projekt wurde eine Teilstichprobe von zwölf TEI-tite-kodierten Texten aus 1000 Texten, die von Blumenbach publiziert wurden, ausgewählt. Das *Handbuch der Naturgeschichte*, wurde zwischen 1779 und 1830 in zwölf deutschsprachigen Editionen veröffentlicht und war eines von Blumenbachs erfolgreichsten Werken. Es wurde auch in mehrere andere Sprachen übersetzt, aber für das Projekt wurden nur die deutschen Editionen ausgewählt. Ausgehend von der 6. Edition wurde eine Liste von Personen, Orten und Fachbegriffen erstellt. Zur Vorbereitung wurde das NER-Tool SynCoPe²¹, das vom Deutschen Textarchiv (DTA) entwickelt

²¹Jurish, B. and Thomas, Chr. (2013). Named Entity Recognition (NER) im Deutschen Textarchiv. Computerlinguistisch gestützte Identifikation von Personen- und Ortsnamen in den Korpora des DTA www.deutschestextarchiv.de/files/DTAE-NER_vortrag-2013-03-06.pdf.

2. Forschungsfeld Named Entity Recognition

wurde, verwendet. Dabei ergab sich ein erstes Set von Begriffen und Namen, die dringend manuelle Korrektur benötigten (50% Fehlerquote). Zusätzlich wurden noch Listen von anderen zeitgenössischen Quellen integriert. Personen und Orte wurden semi-automatisch mit Hilfe der Tools CERL und Getty Thesauri identifiziert. Das Resultat war schlussendlich zufriedenstellend, obwohl es auf die deutschsprachigen Editionen des Buches limitiert war. Precision und Recall lagen über 90%. und es stellte sich heraus, dass die Ergebnisse für die späteren Editionen fast genauso gut waren, wie die mit den optimierten Listen. Nach der NER wurde manuell nochmal nachgebessert, indem man offensichtliche Fehler in Bezug auf mehrdeutige Begriffe korrigierte.

3. Maschinelles Lernen

Machine Learning oder Maschinelles Lernen (ML) ist eine Form von künstlicher Intelligenz (KI), vgl. 3.2. Dabei lernt ein System von Daten und nicht durch explizite Programmierungen (vgl. Hurwitz & Kirsch, 2018, S. 4). Beim maschinellen Lernen nehmen die Algorithmen Trainingsdaten auf, um genauere Modelle anhand dieser Daten zu erstellen. Anfangs wird ein Algorithmus trainiert, um danach im erstellten Modell eine Ausgabe generieren zu können. Wird nach diesem Training eine Eingabe im Modell vorgenommen, erhält man eine Ausgabe. KI und ML Algorithmen sind nicht neu. Die Erforschung von KI geht zurück in die 1940er Jahre. Warren McCulloch und Walter Pitts publizierten 1943 das erste Konzept einer vereinfachten Gehirnzelle, um eine KI zu konstruieren. Dieses Konzept wurde *McCulloch-Pitts (MCP) Neuron* genannt (McCulloch & Pitts, 1943). Arthur Lee Samuels, ein Forscher der International Business Machines Corporation (IBM), entwickelte in den 1950er Jahren eines der ersten maschinellen Lernprogramme - ein selbstlernendes Programm für das Spielen von Schach. Frühe Entwicklungen im maschinellen Lernen legten ihren Schwerpunkt auf symbolische Repräsentationen von gelerntem Wissen, wie Produktionsregeln, Entscheidungsbäumen und logischen Formeln. Nouvel (2016, S. 89) erklärt den grundlegenden Wandel von symbolischen Systemen zu datengesteuerten Systemen Nouvel (2016, S. 89) (vgl. Abbildung 3.1): Bei symbolischen Systemen interagieren die Entwickler hauptsächlich mit dem Modell. Die Daten werden nur für Visualisierungen oder Evaluierungen verwendet. Bei datengesteuerten Systemen verwenden die Entwickler die Daten selbst, wie beispielsweise mithilfe eines Annotationsprozesses (Nouvel, 2016, S. 89). Die allgemeine Struktur dieser Modelle ist starr und vordefiniert, da der Mensch keinen Einfluss darauf nimmt. Nouvel (2016, S. 89) merkt hier aber an, dass erfahrene Entwickler in der Lage sein können, Lernalgorithmen zu modifizieren.

3. Maschinelles Lernen

Die Parameter werden automatisch an die vorhandenen Daten angepasst und das Modell kann dann mit der erschlossenen Konfiguration verwendet werden.

Über die Jahrzehnte wurden KI-Techniken immer breiter genutzt. In den letzten Jahren lag der Fokus auf der Verteilung von Rechenmodellen und billiger Rechenleistung und Speicherung. KI, ML und Deep Learning sind alles Begriffe, die meistens mit der Diskussion von großen Datenmengen (Big Data), Analysen und fortgeschrittene Technologien einhergehen (vgl. Hurwitz & Kirsch, 2018, S. 12). Datengesteuerte (data-driven) Methoden sind im Laufe der letzten Jahrzehnte immer beliebter geworden. Sie bieten Problemlösungen für KI-Systeme. Diese Methoden versuchen mathematische, statistische und kognitive Theorien zu erarbeiten und bestimmen die Parameter eines Modells, indem sie die Daten beeinflussen. Das funktioniert so ähnlich wie bei dem menschlichen Lernprozess. Die Datenaufbereitung ist sehr wichtig für datenge-

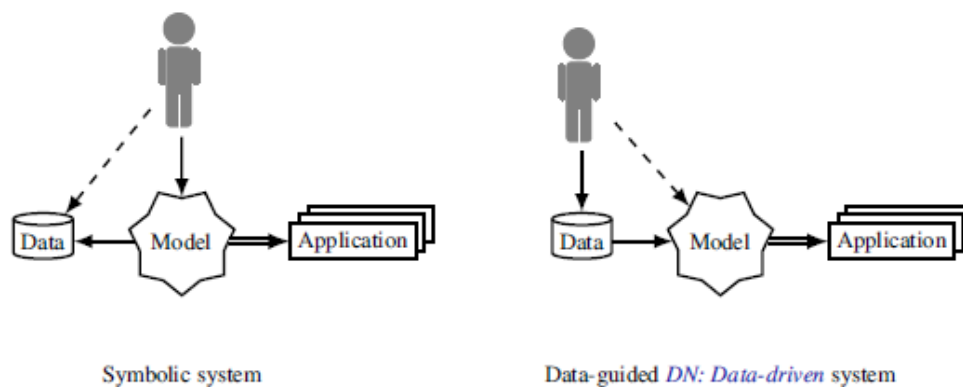


Abbildung 3.1.: Vergleich von symbolischen und datengesteuerten Systemen (Nouvel, 2016, S. 89)

steuerte Systeme. Das gewählte Annotationsformat, die Datenmenge, die dem System für den Lernprozess zur Verfügung gestellt wird, und die Qualität der Daten haben einen großen Einfluss auf die Genauigkeit, Reichweite und die Robustheit des Systems. Diese Modelle sind außerdem abhängig vom Genre der zur Verfügung gestellten Daten. Der Texttyp, der für den Lernprozess

3. Maschinelles Lernen

verwendet wird, bestimmt die Anwendbarkeit der Daten, d.h. wenn ein Modell auf Zeitungsartikel trainiert wird, wird das Ergebnis wahrscheinlich keine so guten Resultate bei historischen Romanen liefern. Beim maschinellen Lernen wird also vorausgesetzt, dass die richtigen Daten auf den Lernalgorithmus angewendet werden.

Es gibt mehrere Methoden des ML, denn je nachdem welche Art von Problem gelöst werden soll, gibt es verschiedene Ansätze, die auf dem Typ und dem Umfang der Daten basieren. Abbildung 3.2 zeigt Methoden des maschinellen Lernens. Für die DH sind vor allem das überwachte und das unüberwachte Lernen von großer Relevanz. Der praktische Teil der Arbeit basiert auf überwachten Lerntechniken, daher wird der Fokus hauptsächlich auf diese Methode gelegt. Trotzdem werden auch andere Methoden kurz angesprochen und erklärt, um einen generellen Überblick zu geben und die Unterschiede zwischen den Methoden zu verdeutlichen.

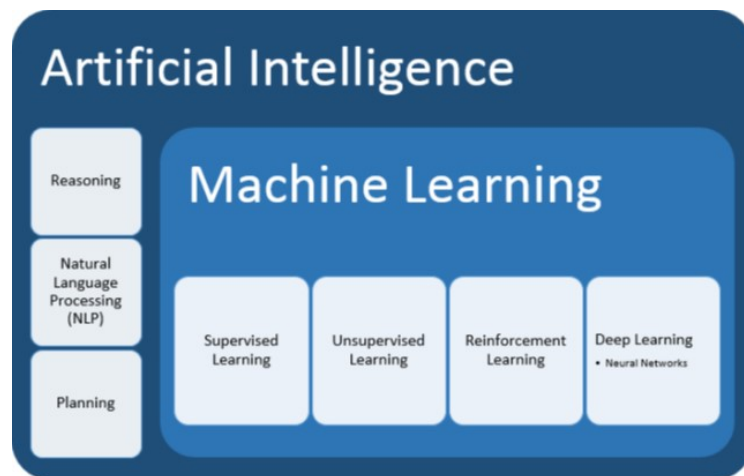


Abbildung 3.2.: KI ist die übergeordnete Kategorie, die maschinelles Lernen und NLP beinhaltet (Hurwitz & Kirsch, 2018, S. 13)

3.1. Überwachtes Lernen (supervised Learning)

Beim überwachten Lernen lernt der Algorithmus Muster und Zusammenhänge anhand von Trainingsdaten. Diese Trainingsdaten, werden als „Ground Truth“ bezeichnet und bestehen meist aus einer großen Menge an Datensätzen. Dabei wird das System so trainiert, dass es NEs anhand spezifischer Eigenschaften erkennt. Der Computer hat die Aufgabe einen Zusammenhang zwischen den Eigenschaften der NEs und den Klassen, die das System von einem vorannotierten Korpus lernt, zu erkennen. Die Grundlage dieses Prozesses liegt auf der Trennung von Lern-, Evaluations- und Anwendungsphase und damit auch auf der Trennung der Trainings- und Evaluationsdaten und den Anwendungsdaten (Jannidis et al., 2017, S. 289). Das Modell soll Vorhersagen über ungesehene oder zukünftige Daten machen können. Überwachte Lernalgorithmen werden also auf Daten angewandt, die zuvor manuell von Menschen vorannotiert wurden, um den Algorithmus so zu trainieren, dass er versteht, welche Merkmale wichtig sind. Folglich sind maschinelle Lernmethoden nicht nur von der spezifischen Technik oder dem Implementationsdetails abhängig, sondern auch von den Eigenschaften, die verwendet werden. Die meisten hochleistungsfähigen Tools

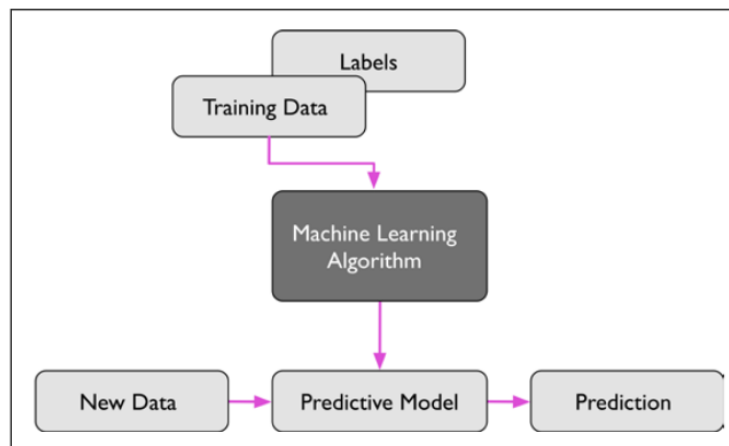


Abbildung 3.3.: überwachtes Lernen (Raschka & Mirjalili, 2017, S. 3)

verwenden nicht-semantische Eigenschaften, wie Sprache, Lemmata, reguläre Ausdrücke, Präfixe, etc. Die hohe Rechenleistung, die benötigt wird, um beispielsweise tiefe syntaktische und semantische Eigenschaften oder große und komplexe Korpora zu analysieren, beschränkt NERC-Systeme auf orthographische, morphologische und oberflächliche syntaktische Eigenschaften (vgl. Jonnalagadda et al., 2010, S. 224). Zu den überwachten Lernalgorithmen, die einen wichtigen Beitrag für die NER geleistet haben, zählen das Hidden Markov Modell (HMM), Entscheidungsbäume, Maximum Entropy Modelle (ME), Support Vector Machines (SVM) und Conditional Random Fields (CRF) (vgl. Sharnagat, 2014, S. 2).

Beim überwachten Lernen geht es hauptsächlich um zwei Problemstellungen, die Klassifikation und die Regression. Im Bereich des kulturellen Erbes sind die bekanntesten Regressionsmethoden die Lineare Regression und die Logistische Regression (vgl. Fiorucci et al., 2020, S. 103). Diese Methoden kommen beispielsweise zum Einsatz bei der Erfassung der Menge und der Art der Steinwerkzeugproduktion (Shott & Habtzghi, 2016), der Entwicklung eines Modells für die Beschreibung der wichtigsten historischen Faktoren, die die Verwendung verschiedenen Keramikarten im Laufe der Zeit beeinflusst haben könnten (García Rivero et al., 2016) und der Vorhersage von archäologischen Fundorten, die nur auf Teilwissen über antike Siedlungen beruhen (Wachtel et al., 2018).

3.1.1. Klassifikation

Überwachtes Lernen mit einzelnen Klassenbezeichnungen wird auch als Klassifikation bezeichnet. Bei der Klassifikation liegt das Ziel darin, die einzelnen Instanzen voneinander als Klassen zu unterscheiden, um hinzukommende Instanzen den richtigen Klassen zuweisen zu können. Bei maschinellen Lernalgorithmen geht es immer darum, eine mathematische Funktion $f : X \rightarrow Y$ zu lernen (vgl. Frochte, 2020, S. 21). Dabei soll einem Element aus X genau ein Element Y

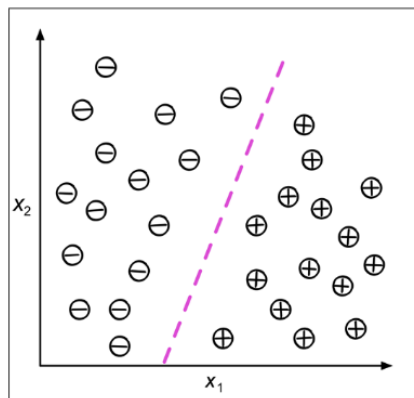


Abbildung 3.4.: binäre Klassifikation (Raschka & Mirjalili, 2017, S. 4)

zugeordnet werden. Ein typisches Beispiel dafür ist die binäre Klassifikation. Dabei lernt der Algorithmus Regeln, um zwischen zwei verschiedenen Klassen zu unterscheiden (z.B.: Spam-Mails oder keine Spam-Mails). Aber Klassifikationen müssen nicht immer binär sein. Ein typisches Beispiel für mehrere Klassen ist die Erkennung von handschriftlichen Zeichen (Raschka & Mirjalili, 2017, S. 3). Dabei wird ein Trainingsdatensatz gesammelt, der aus verschiedenen Handschrift-Beispielen eines jedes Buchstabens des Alphabets besteht. Gibt der Benutzer also ein neues handschriftliches Zeichen ein, würde das Modell den richtigen Buchstaben im Alphabet mit einer bestimmten Wahrscheinlichkeit erkennen und vorhersagen können. Abbildung 3.4 zeigt das Konzept der binären Klassifikation mit 30 Trainingsdaten, 15 Trainingsdaten sind als negative Klassen (Minus-Symbol) und 15 als positive Klassen (Plus-Symbol) gekennzeichnet. Der Datensatz ist zweidimensional, d.h. dass jedes Beispiel mit zwei Werten verknüpft wird: x_1 und x_2 . Ein maschineller Lernalgorithmus kann nun eine Regel erlernen, die beide Klassen voneinander trennen kann und neue Daten einer der beiden Klassen zuordnen kann.

3.1.2. Regression

Die zweite große Problemstellung des überwachten Lernens ist die Regression. Dabei handelt es sich um die Vorhersage von kontinuierlichen Ausgaben, die auch als Regressionsanalyse bezeichnet wird:

In regression analysis, we are given a number of predictor (explanatory) variables and a continuous response variable (outcome or target), and we try to find a relationship between those variables that allows us to predict an outcome (Raschka & Mirjalili, 2017, S. 3).

Auch bei der Regression wird eine Funktion gelernt, hierbei geht es um einen kontinuierlichen Bereich, also beispielsweise um das Erlernen eines optimalen Drehwinkels oder die Höhe eines Kreditrahmens (vgl. Frochte, 2020, S. 23). Raschka und Mirjalili (2017, S. 5) nennen als Beispiel für Regression die Vorhersage von studentischen Testergebnissen. Wenn es eine Verbindung zwischen der Zeit, die ein Studierender für das Lernen vor einem Test und dem Testergebnis gibt, dann kann man diese Daten für das Training benutzen. Das Modell würde die Lernzeit verwenden, um die zukünftigen Testergebnisse der Studierenden vorhersagen zu können. Abbildung 3.5 bildet das Konzept einer linearen Regression ab. Wenn eine vorhergesagte Variable x und eine Antwort-Variable y gegeben sind, wird eine gerade Linie durch diesen Daten gezogen, die die Distanz zwischen den Beispielpunkten und der Linie minimiert (vgl. Raschka & Mirjalili, 2017, S. 5).

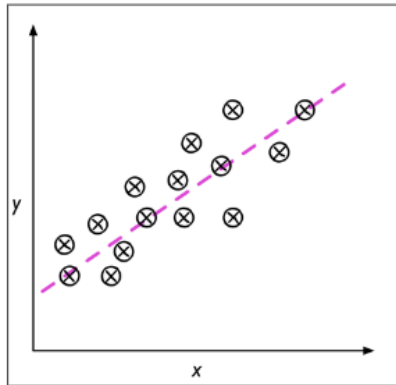


Abbildung 3.5.: lineare Regression (Raschka & Mirjalili, 2017, S. 5)

3.1.3. Wichtige Algorithmen

Es gibt verschiedenen Algorithmen, die beim ML von Bedeutung sind. Zum einen gibt es Klassifizierungsalgorithmen. Ein Klassifizierer ist eine Funktion, die den Input in Form einer Gruppe von Eigenschaften bekommt. Der Output besteht aus Klassenlabels, die mit diesen Eigenschaften assoziiert werden. Klassifizierer sind also Algorithmen, die eine Gruppe von Objekten in verschiedene Kategorien einteilen. Das Ziel dieses Algorithmus ist diese Funktion von einem annotierten Trainingsdatensatz zu lernen (Pustejovsky & Stubbs, 2012). Daher zählt dieser Algorithmus zu den überwachten Lerntechniken. Weiters zählen auch die Entscheidungsbäume (decision trees) zu den Klassifizierungsalgorithmen, dabei werden „Fragen“ an das Korpus gestellt. Die Fragen, die gestellt werden, können sich auf jegliche Information des Datensatzes beziehen. Außerdem zählen noch Maximum Entropie Klassifizierer, Support Vector Machines (SVM) und K-nearest neighbor Methoden dazu.

Eine weiter wichtige Aufgabe des ML ist das Lernen von Sequenzen bestimmter Kategorien. Diese Klasse von Algorithmen werden Sequenzklassifizierer genannt (Pustejovsky & Stubbs, 2012). Dabei handelt es sich um Algorithmen, die eingesetzt werden, wenn die Daten aus Sequenzen von Ein-

heiten zusammengesetzt sind, wie Buchstaben, Morpheme, Wörter, Sätze usw. Sequenzmodellierung wählt immer das beste Label für jedes Element in einer Sequenz. Einer dafür am besten geeigneten Algorithmen ist das Hidden Markov Modell (HMM) (Pustejovsky & Stubbs, 2012). Außerdem zählen noch das Maximum Entropy Markov Model (MEMM) und Conditional Random Fields (CRF) Modelle zu den Sequenzmodellen.

Hidden Markov Modell

Die Theorie von Hidden Markov Modellen (HMM) geht auf die späten 1960er und frühen 1970er Jahre zurück (L. R. Rabiner, 1989, S. 258). In der natürlichen Sprachverarbeitung werden Wörter in Sätzen mit Markierungen der jeweiligen Wortart ausgezeichnet. Dieser Vorgang wird als Part-of-Speech (POS) Tagging bezeichnet. Das Ergebnis ist ein annotierter Text wie in Abbildung 3.6. POS Tagging ist oft ein wichtiger Schritt für die Vorverarbeitung von weiteren NLP-Prozessen. Mithilfe von POS Auszeichnungen kann man die Information der Wörter innerhalb der Sprachstruktur erkennbar machen. Eine der bekanntesten Methoden für diese Annotation und Segmentierung ist die Anwendung von HMMs (vgl. Wallach, 2004, S. 1). Lafferty et al. (2001) definieren HMMs folgendermaßen:

HMMs und stochastische Grammatiken sind generative Modelle, die gemeinsamen Beobachtungs- und Labelsequenzen eine gemeinsame Wahrscheinlichkeit zuweisen; die Parameter werden typischerweise trainiert, um die gemeinsame Wahrscheinlichkeit von Trainingsbeispielen zu maximieren.

Donald	Trump	hat	seinen	Gegner	kritisiert.
PROPN	PROPN	AUX	DET	NOUN	VERB

Abbildung 3.6.: POS Tagging mit Spacy

HMMs sind die frühesten Modelle, die für die Lösung von NER-Problemen für

das Englische (Bikel et al., 1999) entwickelt wurden. Das HMM ist ein generatives Modell, das eine gemeinsame mögliche Verteilung $p(X,Y)$ definiert. Um eine gemeinsame Verteilung zu definieren, müssen generative Modelle alle möglichen überwachten Sequenzen aufzählen (Wallach, 2004, S. 1). Dieser Prozess ist für die meisten Bereiche sehr hartnäckig und schwierig zu bewältigen, wenn die überwachten Elemente nicht als isolierte Einheiten repräsentiert werden und unabhängig von den anderen Elementen in einer überwachten Sequenz vorkommen. Das überwachte Element kann zu jedem gegebenen Zeitpunkt nur direkt vom Zustand oder der Markierung zu dieser Zeit abhängig sein (Wallach, 2004, S. 2).

HMM ist eine unüberwachte Trainingsmethode, die dem Markov Prozess folgt. Der Markov Prozess ist in zwei Komponenten gespalten: in einen *beobachtbaren* und einen *versteckten* Teil. Prozesse in der echten Welt produzieren im Allgemeinen sichtbare Ausgaben, die als Signale bezeichnet werden können (L. R. Rabiner, 1989, S. 257). Diese Signale können diskret (Zeichen eines Alphabets) oder kontinuierlich (Sprachproben, Temperaturmesswerte, Musik, etc.) sein. Die Signalquelle kann stationär (statistische Eigenschaften verändern sich nicht mit der Zeit) oder nicht stationär (Signaleigenschaften ändern sich mit der Zeit) sein. Die Signale können rein oder durch eine andere Signalquelle, wie Lärm, gestört sein. Ein Problem von großem Interesse ist die Kennzeichnung solcher Real-World Signale innerhalb eines Signalmodells (L. R. Rabiner, 1989, S. 257). Das HMM ist ein stochastisches Signalmodell, das vor allem in der Syntaxanalyse im Bereich des NLP oder in der Bioinformatik verwendet wird (vgl. Sugomori et al., 2017, S. 33).

Maximum Entropy Markov Modelle

Maximum Entropy Markov Modelle (MEMM) zeichnen sich dadurch aus, „dass die einzelnen Beobachtungen x_1 im Gegensatz zu den HMM aus willkürlichen, sich überlappenden Merkmalen bestehen“ (Jungermann, 2006, S. 26). Es gibt

zwei Unterschiede zwischen HMMs und MEMMs (Jungermann, 2006, S. 27) :

1. Komplexe, überlappende Merkmale. HMMs ordnen einem Label y jeweils nur eine Beobachtung zu. Die Tatsache, dass das beobachtete Wort großgeschrieben, ein Nomen und an einer bestimmten Stelle des Textes ist, meist aussagekräftiger, als eine Identität. Überlappende Merkmale dienen einer detaillierten Beschreibung der Beobachtung.
2. Bedingte Wahrscheinlichkeit. Bei MEMMs wird die Labelsequenz in Abhängigkeit von der gegebenen Beobachtungssequenz bestimmt. Bei HMMs wird eine kombinierte Wahrscheinlichkeit aus Beobachtungs- und Labelsequenzen gebildet, was jedoch nicht intuitiv ist.

MEMMs sind genauso wie HMMs gerichtete Modelle und ihre bedingten Wahrscheinlichkeiten beruhen auf dem Maximum Entropy-Prinzip. Bei diesem Prinzip wird nicht über alle möglichen Beobachtungssequenzen iteriert, sondern die überlappenden Merkmale werden korrekt behandelt (vgl. Jungermann, 2006, S. 27).

Conditional Random Fields

Conditional Random Fields (CRFs) wurden von Lafferty et al. (2001) als ein statistisches Tool für Mustererkennung und Machine Learning eingeführt. CRFs sind ein probabilistisches Framework für die Markierung und Segmentierung sequentieller Daten. Ein CRF ist eine Form eines ungerichteten grafischen Modells. Der größte Vorteil von CRFs gegenüber HMMs ist ihre an Bedingungen geknüpfte Natur (Wallach, 2004, S. 2). Außerdem haben CRFs kein Label Bias Problem, eine Schwäche des Maximum Entropy Markov Modells und anderen bedingten Markov Modellen, die auf gerichteten graphischen Modellen basieren. CRFs übertreffen sowohl MEMMs als auch HMMs bei einer Reihe von Sequenzkennzeichnungsaufgaben in der realen Welt (Lafferty et al., 2001).

Eines der am besten funktionierenden Modelle für deutsche NER ist das linear-chain CRF (Riedl & Padó, 2018). Dieses Modell bildet die Basis für

viele bekannte Systeme (vgl. Benikova et al., 2015) und kann leicht sprach- und bereichsspezifisches Wissen von Lexika oder Ortsverzeichnissen übernehmen (Riedl & Padó, 2018, S. 120). STANFORDNER²²(Finkel & Manning, 2009) ist eines der auf CRF basierenden Systeme. STANFORDNER stellt Modelle für mehrere Sprachen zur Verfügung. Das Standardmodell für Deutsch wurde anhand der GERMAN CoNLL 2003 Daten (Tjong Kim Sang, Erik F. & de Meulder, 2003) trainiert.

Support Vector Machines

Support Vector Machines (SVMs) (dt. „Stützvektormaschinen“) ist eine überwachte Lernmethode für binäre Klassifikation oder Regression. SVMs dienen zur Klassifikation von Objekten. Das Ziel der SVM ist das Finden einer Hyperebene in einem N-dimensionalen Bereich (N steht für die Anzahl der Merkmale), die die Datenpunkte eindeutig klassifiziert (Gandhi, 07.06.2018). Datenpunkte, die



Abbildung 3.7.: mögliche Hyperebenen (Gandhi, 07.06.2018)

auf beiden Seiten der Hyperebene liegen, können unterschiedlichen Klassen zugeordnet werden. Die Dimension der Hyperebenen hängt von der Anzahl der Merkmale ab, d.h. wenn die Anzahl der eingegebenen Merkmale 2 ist, dann besteht die Hyperebene aus einer Linie, wenn sie 3 ist, dann wird die Hyperebene zweidimensional (Gandhi, 07.06.2018, vgl.). Stützvektoren sind

²²<https://nlp.stanford.edu/software/CRF-NER.shtml>

Datenpunkte, die näher an der Hyperebene liegen und deren Position und Ausrichtung beeinflussen. Durch diese Stützvektoren kann der Spielraum des Klassifikators maximiert werden. Werden Stützvektoren gelöscht, wird die Hyperebene verändert, das sind die Punkte, die beim Aufbau eines SVM helfen (Gandhi, 07.06.2018, vgl.).

Anwendungsbereiche von SVMs im Cultural Heritage Sektor sind beispielsweise die automatische Dokumentlayout-Analyse von mittelalterlichen Manuskripten (Yang et al., 2017) und die Authentifizierung von Kunstwerken mithilfe von hyperspektraler Bildgebung (engl. Hyperspectral Imaging, HSI) und Signalverarbeitungstechniken zur Identifizierung und Klassifizierung von Pigmenten (Polak et al., 2017).

Supervised Deep Neural Networks

Deep Neural Networks (DNNs) wurden in den letzten Jahren sehr erfolgreich für unterschiedliche Anwendungen in den Bereichen Computer Vision und NLP eingesetzt. DNNs sind fähig High-Level-Merkmale aus Daten zu lernen, was die manuelle Erstellung von Merkmalen ersetzt. In den DH mangelt es oft an großen annotierten Datensätzen, daher versucht man den Transfer-Learning-Ansatz anzuwenden. Hierbei werden die letzten Schichten eines trainierten Netzwerks auf dem Ziel-Datensatz abgestimmt (vgl. Fiorucci et al., 2020, S. 105).

Cultural Heritage in Verbindung mit Computer Vision sind in den DH ein großes Anwendungsgebiet des ML. Computer Vision ist ein Fachgebiet, das sich mit dem Einsatz von Berechnungen beschäftigt, um ein hochgradiges Verständnis von Bildern zu bekommen (Wevers & Smits, 2019). Jüngste Beiträge zur digitalen Werkanalyse sind die Untersuchung von Lernmethoden für Ähnlichkeitsmetriken, wie beispielsweise die Vorhersage des Stils, des Genres und des Künstlers eines Gemäldes (Saleh & Elgammal, n. d.), die Erkennung von gefälschten Kunstwerken durch eine Strichanalyse (Elgammal et al., 2018) oder die künstlerische Übertragung eines Stils (Xu et al., n. d.).

3.2. Unüberwachtes Lernen (unsupervised Learning)

Unüberwachtes Lernen wird dann verwendet, wenn man eine große Menge von unannotierten oder unstrukturierte Daten hat. Mit unüberwachten Lern-techniken ist es möglich, die Struktur der Daten zu erkennen und wichtige Informationen zu extrahieren ohne die Anleitung einer bekannten Variable (vgl. Raschka & Mirjalili, 2017, S. 7). Social-Media Anwendungen wie Twitter, Instagram und Snapshot verfügen beispielsweise über große Mengen nicht vorannotierten Daten („Machine Learning“, 2020). Beim unüberwachten Lernen werden Algorithmen verwendet, die die Daten anhand der Muster oder Cluster klassifizieren. Dabei wird ein iterativer Prozess durchgeführt, der die Daten automatisch analysiert. Unüberwachtes Lernen unterscheidet sich in zwei Punkten vom überwachten Lernen. Erstens gibt es keine Trennung zwischen den verschiedenen Phasen, daher werden die Datensätze auch nicht getrennt. Das Verfahren operiert auf einem einzigen Datensatz. Außerdem werden keine Klassen vorgegeben. Beim unüberwachten Lernen werden Regelmäßigkeiten, Korrelationen und andere Zusammenhänge zwischen Merkmalen für die Bildung von gut abgegrenzten Gruppen innerhalb der Daten genutzt. (vgl. Jannidis et al., 2017, S. 289). Der Algorithmus wird auf Daten angewandt, die nicht annotiert wurden und lernt mithilfe komplexer Prozesse und Muster von selbst, welche Eigenschaften und Merkmale wichtig sind. Ein großes Problem des unüberwachten Lernens liegt in den Anforderungen einer großen Menge an Merkmalen. Um ein gutes Modell zu erlernen, ist ein starkes Set an Merkmalen und ein großes annotiertes Korpus nötig. Viele Sprachen haben keine so große Menge an vorannotierten Korpora. Um dieses Problem zu lösen, wurden unüberwachte Lerntechniken für die NER entwickelt (Sharnagat, 2014, S. 8).

3.2.1. Clustering

Clustering ist eine Datenanalysetechnik, die es uns erlaubt, einen Stapel an Informationen in Untergruppen (Cluster) zu organisieren, ohne vorher zu wissen, zu welcher Gruppe die Informationen gehören (vgl. Raschka & Mirjalili, 2017, S. 7). Jeder Cluster, der während der Analyse auftaucht, definiert eine Gruppe von Objekten, die ein gewisses Maß an Ähnlichkeit aufweisen, sich aber von anderen Objekten in andern Clustern unterscheiden. Deswegen wird Clustering auch manchmal als unüberwachte Klassifikation bezeichnet (vgl. Raschka & Mirjalili, 2017, S. 7). Frochte (2020) nennt ein sehr anschauliches Beispiel von

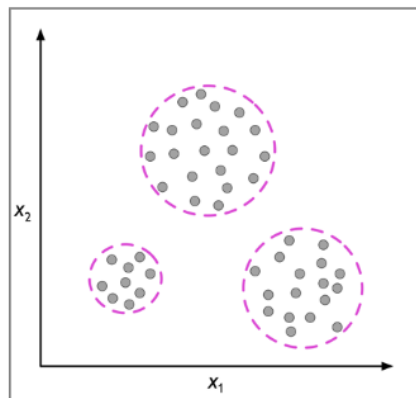


Abbildung 3.8.: Clustering (Raschka & Mirjalili, 2017, S. 7)

vier Pflanzen-Skizzen (vgl. Abbildung 3.9). Wenn man diese vier Zeichnungen seinen Mitmenschen vorlegt und ihnen die Aufgabe stellt, daraus zwei Gruppen zu bilden - also eine Gruppe mit drei Pflanzen und eine Gruppe mit einer und zwei Gruppen mit jeweils zwei Elementen - werden verschiedene Antworten zurückkommen. Ob man jetzt die tote Pflanze aussortiert oder den Bambus rausnimmt, da er kein Gehölz ist, ist egal, da alle Antworten richtig wären. Jede Pflanze oder jeder Datensatz hat verschiedene Merkmale. Beim Clustering werden diese Merkmale unterschiedlich gewichtet und dementsprechend gruppiert.

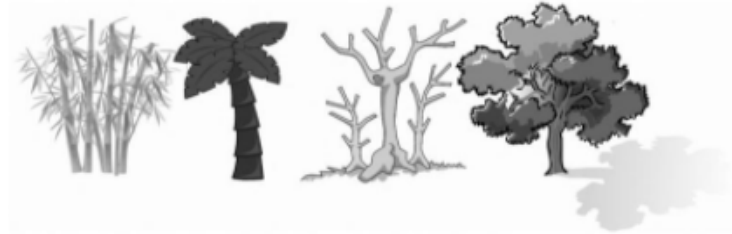


Abbildung 3.9.: Skizzen verschiedener Pflanzen (Frochte, 2020, S. 26)

Clustering wird in den DH beispielsweise eingesetzt bei der Konstruktion eines Codebuchs visueller Wörter zur chronologischen Klassifizierung antiker Gemälde (Chen et al., 2017), der Gruppierung von Gemälden nach künstlerischem Stil (Gultepe et al., 2018) oder der Bestimmung maximaler Brenntemperaturen von antiker Keramik (Kostadinova-Avramova et al., 2018).

3.3. Teilüberwachtes Lernen (semi-supervised Learning)

Teilüberwachtes Lernen liegt zwischen überwachtem und unüberwachtem Lernen. Die meisten teilüberwachten Lernstrategien basieren entweder auf der Erweiterung von überwachten oder unüberwachten Lernen (vgl. Sharnagat, 2014, S. 8). Teilüberwachte Lernalgorithmen werden aus einer Kombination von annotierten und nicht annotierten Daten trainiert. Das ist meist dann sinnvoll, wenn man nicht genug annotierte Daten für ein akkurates Modell und keine Möglichkeiten und Ressourcen hat, um an mehr Daten zu gelangen. Die Algorithmen starten meist mit einer kleinen Menge eines annotierten Grunddatensets und einer großen Menge eines nicht annotierten Korpus. Mit jeder Iteration werden mehr Annotationen generiert und gespeichert. Wenn ein bestimmter Schwellenwert erreicht wird, wird die Iteration gestoppt (Sharnagat, 2014, S. 8). Das Annotieren von riesigen Datenmengen für das überwachte

Lernen ist meist sehr teuer und zeitintensiv. Durch teilüberwachtes Lernen kann man die Genauigkeit erhöhen und dabei Zeit und Geld sparen. Die wichtigste Technik für teilüberwachtes Lernen wird *bootstrapping* genannt und beinhaltet einen kleinen Teil von Überwachung, wie beispielsweise eine kleine Gruppe von Seeds, um einen Lernprozess zu starten (vgl. Nadeau & Sekine, 2007, S. 5).

Auch in den DH kommen teilüberwachte Algorithmen zum Einsatz, so haben Carraggi et al. (2018) ein teilüberwachtes visuell-semantisches Modell für das crossmodale Retrieval von Bildern und Bildunterschriften entwickelt. Sie trainierten zwei Autoencoder für visuelle und textuelle Daten. X. Shen et al. (2019) stellten eine Methode zum Auffinden von Kunstwerken vor, die doppelte visuelle Elemente aufweisen.

3.4. Bestärkendes Lernen (Reinforcement Learning)

Beim bestärkenden Lernen wird ein System (Agent) entwickelt, dass die Performance anhand von Interaktionen mit der Umgebung verbessert (Raschka & Mirjalili, 2017, S. 6). Eine Definition für den Begriff des „Agents“ lieferten 1996 Franklin und Graesser:

Ein autonomer Agent ist ein System, welches sich in einer Umgebung befindet und [zugleich] Teil von ihr ist, das diese Umgebung wahrnimmt und im Lauf der Zeit auf sie einwirkt, um so mit Blick auf die eigenen Ziele das zu beeinflussen, was er in der Zukunft wahrnimmt. (Franklin & Graesser, 1996)

Dabei erhält das System kontinuierlich Rückmeldungen als Belohnungs- oder Bestrafungssignale. Im Gegensatz zum überwachten Lernen ist dieses Feedback nicht der korrekte Ground Truth Marker oder Wert, sondern ein Messwert der Belohnungsfunktion, wie gut die Wirkung bewertet wurde. Durch diese Abfolge von Belohnung und Bestrafung soll ein möglichst gute Strategie für

die Problemlösung erlernt werden. Laut Lorenz (2020) gehört das bestärkende Lernen zu den sich aktuell am dynamischsten entwickelnden Forschungsgebieten des maschinellen Lernens. Die vorher genannten Methoden benutzen Algorith-

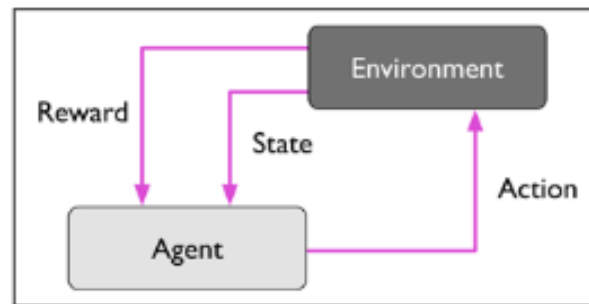


Abbildung 3.10.: bestärkendes Lernen (Raschka & Mirjalili, 2017, S. 6)

men, die aus Daten lernen, das bestärkende Lernen jedoch funktioniert durch Ausprobieren. Das ist oft nützlich für die Robotik, für die DH ist es jedoch (noch) nicht wirklich relevant (LaTeX Ninja'ing and the Digital Humanities, 2020) und auch die Literaturrecherche zeigte, dass diese Methode aktuell nicht zum Einsatz kommt.

3.5. Deep Learning

Deep Learning ist eine Methode des maschinellen Lernens, „die neuronale Netzwerke in aufeinanderfolgenden Schichten integriert, um in iterativer Weise von Daten zu lernen“ („Machine Learning“, 2020). Deep Learning ist besonders gut geeignet, um Muster in unstrukturierten Daten zu erkennen. Neuronale Netzwerke sollen wie das menschliche Gehirn funktionieren. Diese Modelle verwenden „Neuronen“ in Schichten. Jedes Neuron bekommt eine Eingabe, wird berechnet, und transferiert das Ergebnis auf andere Neuronen. Die erste Schicht bekommt den Aufgabeninput, welcher dann durch das ganze Netzwerk wandert. Bei der letzten Schicht erfolgt die Ausgabe des vorhergesehenen

Ergebnisses des Netzwerks. Der größte Vorteil neuronaler Netzwerke ist ihre Fähigkeit automatisch die optimale repräsentative Gruppe von Merkmalen für die gegebene Aufgabe zu berechnen ohne sich auf manuell ausgewählte Merkmale zu verlassen (vgl. Suissa et al., 2020, S. 2). Auch in den DH kommen neuronale Netzwerke zum Einsatz. Ein paar Beispiele:

„Transkribus“²³. Mit Transkribus kann man Textmaterial automatisch transkribieren lassen, wie beispielsweise handschriftliche Dokumente, oder in Fraktur gedruckte Texte. Eingesetzt werden dafür eigens trainierte ML-Modelle, die bestimmte Schrifttypen in verschiedenen Sprachen erkennen. Transkribus bietet verschiedene Tools für die automatische Texterkennung an, wie KI-unterstützte HTR, Bilderkennung (Layout-Analyse) und Strukturerkennung. Außerdem ist es mit Transkribus möglich ein eigenes Recognition-Modell zu trainieren.

Iyyer et al. (2016) beschäftigten sich mit der Veränderung von fiktiven Beziehungen zwischen zwei Charakteren über Zeit (z.B.: von besten Freunden zu Feinden). Dafür entwickelten sie ein unüberwachtes neuronales Netzwerk, das mit wörterbuchbasiertem Lernen arbeitet und interpretierbare, exakte Beziehungskurven generiert.

Suissa et al. (2020, S. 2) entwickelten eine Methode für das automatische Generieren von sprach- und aufgabenspezifischen Trainingsdaten, um die Ergebnisse des Neuronalen Netzwerks für die OCR-Nachkorrektur zu verbessern. Das Evaluierungskorpus basierte auf hebräischen Zeitungen des JPress Projekts und bestand aus 150 zufällig ausgewählten OCRten Artikeln. Die Ergebnisse zeigten, dass die Leistung des neuronalen Netzwerks für die OCR-Nachkorrektur stark vom Genre und dem Bereich der Trainingsdaten abhängt. Außerdem übertrafen die NN, die mit den vorgeschlagenen Methoden trainiert wurden, andere moderne NN für die OCR-Nachkorrektur und komplexe Rechtschreibprüfungen.

²³<https://readcoop.eu/transkribus/?sc=Transkribus>

4. NER und die Anwendung auf historische Texte

Jeder der ein NER-System benutzen und anwenden will, wird sich anfangs zwei Fragen stellen:

1. Welche Entitäten möchte ich auswählen?
2. Wie möchte ich diese Entitäten kennzeichnen?

Die Antworten auf diese beiden Fragen werden je nach Projekt unterschiedlich ausfallen. Bei der ersten Frage geht es darum, die relevanten Kategorien der Entitäten auszuwählen. Dafür ist es wichtig, zu wissen, welche und wie viele NE-Kategorien man für die jeweiligen Texte verwenden will. Die zweite Frage beschäftigt sich mit den dafür verfügbaren Annotationskonventionen. Dabei sollte man sich mit den verschiedenen Annotationsmöglichkeiten auseinandersetzen und sich schlussendlich auf eine einheitliche beschränken. Hat man sich entschieden und ein NER-Modell gefunden, das für die jeweilige Fragestellung passend erscheint, sollte die Qualität der Ergebnisse auch bewertet werden, denn nicht jedes Modell funktioniert gleich gut. Die Bewertung von NER-Systemen erfolgt durch den Vergleich zwischen der per Hand angefertigten Annotationen und der Ausgaben des NER-Systems.

4.1. Annotation der Trainingsdaten

Die Annotation des Trainingsmaterials ist der erste wichtige Schritt, um ein Trainingskorpus zu erstellen. Je besser und genauer die Auszeichnungen für den **Gold Standard** gemacht werden, desto besser ist das Trainingsergebnis. Dieser Gold Standard ist die finale Version der annotierten Daten und wird für

das ML verwendet. Es gibt mehrere Möglichkeiten Texte so zu annotieren, dass sie maschinenlesbar werden. Beispielsweise haben sich Pustejovsky und Stubbs in *Natural language annotation for machine learning*(2012) genauer mit der Annotation von natürlicher Sprache für das ML befasst. Aus der Vielzahl der Annotationsmöglichkeiten möchte ich drei hervorheben, die in den DH stark vertreten sind und auch für den praktischen Teil der Masterarbeit relevant waren.

Die International Organization for Standardization²⁴ (ISO) ist verantwortlich für die Erstellung von weltweiten Standards, um die Kompatibilität zwischen Systemen zu gewährleisten. Es gibt auch ISO Standards für linguistische Annotationen und für gewisse Spezifikationstypen, wie ISO-TimeML (Standard für die zeitliche Information in einem Dokument) und ISO-Space (Standard für die Repräsentation von Orten, örtliche Konfigurationen und die Veränderung natürlicher Sprache) (vgl. Pustejovsky & Stubbs, 2012, S. 80). Zu diesen ISO-Standards gehört auch die Standard Generalized Markup Language (SGML)²⁵. SGML ist eine Metasprache, die Texte standardisieren und maschinenlesbar machen sollte. Die bekanntesten auf SGML basierenden Sprachentwicklungen sind HTML und eXtensible Markup Language (XML). XML gilt als Standard für die Auszeichnung von Texten. XML dient als erweiterbare Auszeichnungssprache oder Metabezeichnungssprache zur Definition von weiteren Markup-Sprachen (Kurz, 2015). Auf der MUC-7 (Chinchor & Robinson, 1998) wurde in Form der SGML annotiert. Das Markup hatte folgende Form:

```
<ELEMENT-NAME ATTR-NAME="ATTR-VALUE"...>text-string</ELEMENT-NAME>
```

Beispiel:

```
<ENAMEX TYPE="PLACE">Paris</ENAMEX>
```

```
<ENAMEX TYPE="ORGANIZATION">Universität Graz</ENAMEX>
```

²⁴<https://www.iso.org/home.html>

²⁵siehe <https://www.w3.org/TR/WD-html40-970708/intro/sgmltut.html>

In der Praxis hat XML heute SGML in vielen Bereichen abgelöst. XML wurde im Jahre 1998 von der XML Working Group des World Wide Web Consortiums (W3C) entwickelt. Man wollte die Verwendung von SGML konkretisieren und durch Regeln festlegen.

In den DH werden die meisten Texte mithilfe der auf XML basierenden Anwendung der Text Encoding Initiative (TEI)²⁶ annotiert. Die TEI ist ein Konsortium, das seit 1987 einen gemeinsamen Standard für die Repräsentation von digitalen Texten entwickelt und bereitstellt. Die TEI-Richtlinien²⁷ wurden speziell für Annotationsmethoden von maschinenlesbaren Texten in den Geisteswissenschaften, Sozialwissenschaften und der Linguistik entwickelt. Seit 1994 werden die TEI-Richtlinien von Bibliotheken, Museen, Publishern und Wissenschaftlern verwendet, um Texte für die Forschung, Lehre und Archivierung online aufzubereiten. Die TEI ist eine Nonprofit-Organisation, die aus akademischen Institutionen, Forschungsprojekten und Wissenschaftlern der ganzen Welt besteht. Der XML-Editor Oxygen²⁸ eignet sich besonders gut für Auszeichnungen mit der TEI. Ein Beispiel einer Annotation mit TEI könnte folgendermaßen aussehen:

```
<persName>
  <forename type="first">Franklin</forename>
  <forename type="middle">Delano</forename>
  <surname>Roosevelt</surname>
</persName>
```

Ein bekannter Standard beim Tagging von NEs ist das IOB-Schema (Ramshaw & Marcus, 1995), vgl. Tabelle 4.1. Ratinov und Roth (2009) zeigten, dass Systeme, die anhand des BILOU-Schemas trainiert wurden bessere Resultate lieferten, als solche, die BIO-Tagging benutzten. Tabelle 4.2 zeigt, wie ein Satz mit dem IOB-Schema kodiert wird.

²⁶<https://tei-c.org/>

²⁷<https://tei-c.org/guidelines/>

²⁸<https://www.oxygenxml.com/>

4. NER und die Anwendung auf historische Texte

- I Token befindet sich innerhalb (inside) einer Entität
- O Token befindet sich außerhalb (outside) einer Entität
- B Token steht am Anfang (begin) einer Entität
- L Token ist das letzte (last) Token einer mehrteiligen Entität
- U Entität besteht nur aus einem einzelnen Token (single-token unit)

Tabelle 4.1.: IOB-Schema

Token	Tag
Joe	B
Biden	I
ist	O
Präsident	O
der	O
USA	B

Tabelle 4.2.: Beispiel für IOB-Tagging

Außerdem gibt es noch Standards, die von einer Community entwickelt wurden. Beispielsweise entstehen solche Community-Standards, wenn Korpora bereitgestellt werden, die einen großen Einfluss auf die Literatur hatten. Sobald diese Korpora zur Verfügung gestellt und häufig genutzt werden, wird die Annotation, die dafür benutzt wurde, zum Standard.

Hilfe für die Annotation von Texten bieten Auszeichnungstools, die für Anwendungen des maschinellen Lernens entwickelt wurden. Beispielsweise ermöglicht das kostenpflichtige Annotationstool Prodigy²⁹ eine schnelle und flexible Annotation mithilfe einer webbasierten Annotationsanwendung. Prodigy wurde speziell für ML-Modelle entwickelt. Aber auch Open-Source Programme wie doccano (Hiroki Nakayama et al., 2018) oder Inception (Jan-Christoph Klie et al., 2018) sind für Textannotationen eine gute Wahl.

²⁹<https://prodi.gy/>

4.2. Trainings-, Validierungs- und Testdaten

Beim maschinellen Lernen benötigt man einen Trainingsdatensatz, um ein Modell richtig trainieren zu können. Unüberwachte Methoden brauchen keine Beispiele, da man sie direkt mit den Eingabedaten trainieren kann. Beim überwachten Lernen hingegen sind Beispieldaten, also vorannotierte Daten notwendig. Meistens wird das Korpus dabei in zwei Teile geteilt: das Entwicklungskorpus und das Testkorpus. Das Entwicklungskorpus wird danach wiederum unterteilt in: Trainingsdaten und Validierungsdaten. Je nach Aufgabe kann die Aufteilung auch anders aussehen. Eine mögliche Aufteilung des Datensatzes könnte folgendermaßen aussehen (vgl. Pustejovsky & Stubbs, 2012, S. 29):

- 70% Trainingsdaten
- 20% Validierungsdaten
- 10% Testdaten

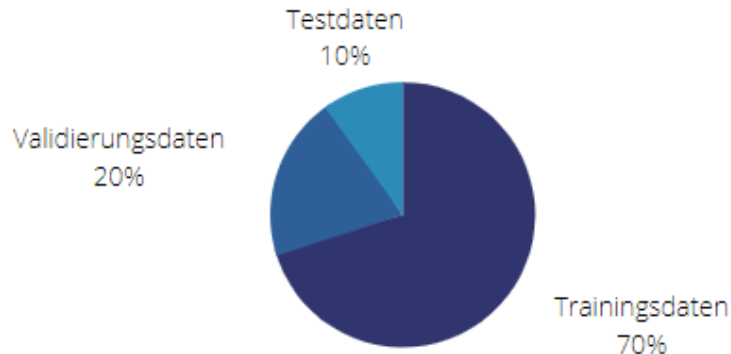


Abbildung 4.1.: mögliche Aufteilung von Trainings-, Validierungs- und Testdaten

Ein Trainingsdatensatz besteht aus vorannotierten Beispielen, die für das Lernen der Muster und Zusammenhänge in den Daten benötigt werden. Aus diesen Daten soll der Algorithmus lernen und ein Modell erstellen. Wie gut die Vorhersagen des Algorithmus bei neuen Instanzen sind, wird anhand der

Generalisierungsfähigkeit gemessen (vgl. Alpaydin, 2014, S. 39). Man benötigt dafür Zugang zu den Daten außerhalb des Trainingsdatensatzes. Dies wird simuliert, indem man den Datensatz in zwei Teile unterteilt (vgl. Alpaydin, 2014, S. 40): einen Trainings- und einen Validierungsdatensatz. Man benötigt aber noch einen dritten Datensatz: die Testdaten. Die Testdaten beinhalten Beispiele, die nicht im Training oder in der Validierung verwendet wurden. Die gleiche Training/Validierung Aufteilung darf allerdings nicht erneut verwendet werden, denn nachdem diese benutzt wurde, werden die Validierungsdaten Teil der Trainingsdaten (vgl. Alpaydin, 2014, S. 40).

4.3. Evaluierungsmethoden

Wenn man ein NER-System trainiert, wird das System in den meisten Fällen anhand von „Precision“, „Recall“ und dem „F-Score“ bewertet. Dabei wird evaluiert, wie genau der Algorithmus den Datensatz annotiert hat. Diese Technik wird die „Precision and Recall Metrik“ genannt (Pustejovsky & Stubbs, 2012, S. 30), vgl. Tabelle 4.3. Für jede Annotation eines Elements, wird der Datensatz in zwei Unterkategorien geteilt: einer wird als *relevant* für die Annotation gekennzeichnet, der andere ist nicht relevant. Unter relevant „wird ein Verhältnis zwischen dem Informationsbedürfnis des Nutzers und den ausgegebenen Dokumenten“ (Klinke, 2017, S. 269) verstanden. Der Precision-Wert bezieht sich auf die prozentuale Anzahl der richtig erkannten Ergebnisse des Systems. Der Recall-Wert bezieht sich auf die prozentuale Anzahl der vom System insgesamt korrekt erkannten Entitäten. Im Folgenden werden die Evaluierungsmetriken (Stepanyan, 2020, S. 2) genauer erläutert:

- **False Positive (FP):** Entitäten, die vom NER-System zurückgegeben werden, aber nicht in der Ground Truth auftauchen. Dabei handelt es sich um nicht relevante Ergebnisse.
- **False Negatives (FN):** Entitäten, die nicht vom NER-System zurück-

4. NER und die Anwendung auf historische Texte

		Predicted Labeling	
		positive	negative
Gold Labeling	positive	true positive (tp)	false negative (fn)
	negative	false positive (fp)	true negative (tn)

Tabelle 4.3.: Precision und Recall Metrik (Pustejovsky & Stubbs, 2012, S. 30)

gegeben wurden, aber in der Ground Truth auftauchen.

- **True Positives (TP)**: Entitäten, die vom NER-System zurückgegeben werden und auch in der Ground Truth enthalten sind.
- **True Negatives (TN)**: Entitäten, die vom NER-System korrekt ignoriert werden, wenn sie irrelevant sind (vgl. Pustejovsky & Stubbs, 2012, S. 172).
- **Precision**: Anzahl der prognostizierten Strings, die exakt mit den Strings der Evaluierungsdaten übereinstimmen.

$$precision = \frac{|\{relevanteEntitäten\} \cap \{zurückbekommeneEntitäten\}|}{|\{zurückbekommeneEntitäten\}|}$$

verkürzte Schreibweise:

$$precision = \frac{\#TP}{\#(TP + FP)}$$

- **Recall**: Anzahl der Entitäten in den Evaluierungsdaten, die exakt an der gleichen Stelle vorkommen, wie in den Vorhersagen.

$$recall = \frac{|\{relevanteEntitäten\} \cap \{zurückbekommeneEntitäten\}|}{|\{relevanteEntitäten\}|}$$

verkürzte Schreibweise:

$$recall = \frac{\#FP}{\#(TP + FN)}$$

- **F-Score:** harmonischer Mittelwert von Precision und Recall

$$F = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

4.4. NER Systeme und Tools für die DH

Für die DH ist es oft sehr herausfordernd mit aktuellen NER Systemen und Tools zu arbeiten, da man es in den meisten Fällen nicht mit standardisierten und modernen Sprachen zu tun hat. Meist arbeitet man mit seltenen oder alten Sprachen, kaum strukturierten Daten oder anwendungsspezifischen Klassifikationsaufgaben (vgl. Erdmann et al., 2019, S. 1). Wenn ein Korpus ein oder auch mehrere dieser Merkmale aufweist, kann man ihn meist nicht automatisch mit standardisierten NER Modellen annotieren. Es gibt zwar DH Initiativen wie beispielsweise das Pelagios Netzwerk (<https://pelagios.org/>), das geographische Daten von historischen Quellen sammelt, jedoch benötigen solche Projekte eine sehr gute Finanzierung (Simon et al., 2016). Linguistische Tools werden normalerweise anhand der Schwierigkeit der Analyse kategorisiert (CLARIN-D/SfS-Uni. Tübingen, 2012). Abbildung 4.2 zeigt eine vereinfachte Hierarchie linguistischer Einheiten, Unterdisziplinen und Tools. Sie dient zum besseren Verständnis, wie hierarchische Abhängigkeiten zwischen den Analysestufen linguistische Theorien und Tools beeinflussen. Die Abbildung zeigt auch, dass die NER eindeutig ein hohes Analyseniveau aufweist. Anzumerken ist hier, dass Tools, die weiter unten in der Hierarchie stehen sehr oft von höheren Stufen abhängig sind. So können beispielsweise phonetische und phonologische Analysen morphologische und syntaktische Analysen beeinflussen. Außerdem sind viele Tools sprachspezifisch, d.h. dass nicht jedes Tool für jede Sprache

4. NER und die Anwendung auf historische Texte

	Discipline	Units and categories	Tools
↑ Higher levels of analysis	Pragmatics	Discourse types	Emotion analyzers
	Discourse theory	Genres	Rhetorical coherency
	Rhetoric	Classes of speech act	analyzers
	Speech act theory	Emotions	Named entity recognizers
↓ Lower levels of analysis	Semantics	Predicates	Word sense disambiguators
		Logical representations	Semantic role analyzers
		Word senses	Coreference and anaphora tools
	Syntax	Sentences	Parsers
		Phrases	Chunkers
		Words	
	Morphology and lexical analysis	Words	Stemmers
		Prefixes and suffixes	Lemmatizers
		Singular and plural	Part-of-speech taggers
		Conjugations	
		Declensions	
	Phonetics and phonology	Sounds	Speech recognition
		Phonemes	Spectrograms/Sonograms
		Syllables	
		Intonational categories	

Abbildung 4.2.: Hierarchie linguistischer Tools (CLARIN-D/SfS-Uni. Tübingen, 2012)

gleich gut geeignet ist (vgl. CLARIN-D/SfS-Uni. Tübingen, 2012). WebLicht ist eine webbasierte Anwendung für automatische Annotation von Textkorpora, das sowohl verschiedene Services für die Verarbeitung von Daten anbietet als auch eine anwenderfreundliche grafische Oberfläche für die Verkettung von verschiedenen linguistischen Anwendungen hat. WebLicht bietet auch mehrere Pipelines für deutschsprachige NER an, vgl. Tabelle 4.4.

Laut Erdmann et al. (2019) ist das bekannteste NER System unter Geisteswissenschaftlern Stanford NER, da es mehrere vortrainierte Modelle in verschiedenen Sprachen und eine Möglichkeit für die Erstellung von eigenen Modellen bietet. Die Arbeiten von Erdmann et al. (2019) (NER System für Latein) und Sprugnoli (2018) (neuronaler Zugang zur Identifikation von Ortsnamen in historischen Texten) zeigen, dass man existierende Modelle verbessern kann, um sie für relevante Daten im DH-Bereich wie Literatur, Geschichte oder Kulturerbe anzupassen. Beispielsweise ist der Humanity Entity Recognizer

4. NER und die Anwendung auf historische Texte

Service	Quelle
Person Name Recognizer	BBAW: Berlin-Brandenburg Academy of Sciences and Humanities
Sticker Named Entity	SfS: Uni-Tuebingen
Sticker2-German	SfS: Uni-Tuebingen
German Named Entity Recognizer	SfS: Uni-Tuebingen
German NER	IMS: Universität Stuttgart
OpenNLP Externally Trained Named Entity Recognizer	SfS: Uni-Tuebingen

Tabelle 4.4.: WebLicht NER Services für das Deutsche

(HER)³⁰ (Erdmann et al., 2019) ein NER-Tool, das für digitale Geisteswissenschaftler entwickelt wurde. Es dient zur automatischen Identifizierung von Entitäten in großen Textkorpora, dabei können verschiedene Entitätstypen, Sprachen, Stile und Textstrukturen verarbeitet werden (Erdmann et al., 2019). Mit diesem Toolkit ist es möglich, ein eigenes NER Modell zu erstellen. Im Anhang befindet sich zusätzlich eine Tabelle ausgewählter NER Tools, die in den DH eingesetzt werden.

4.5. Probleme der NER bei historischen Texten

Im Folgenden werde ich näher auf die Probleme eingehen, die bei der NER von historischen Texten auftreten.

4.5.1. Digitalisierung von historischen Texten

Die Digitalisierung der Dokumente ist ein wichtiger Schritt für die Verarbeitung von historischen Texten. Moderne Texte sind meist in digitalen Formaten verfügbar, egal ob als reiner Text oder als Korpus. Bei historischen Texten und Dokumenten sieht diese Situation allerdings anders aus. Zwar haben viele Kul-

³⁰github.com/alexerdmann/HER

turinstitutionen, wie Bibliotheken, Museen oder Archive historische Dokumente mittlerweile digitalisiert, allerdings unterscheiden sich die Digitalisierungsmethoden sowie das digitalisierte Material oft voneinander (vgl. Piotrowski, 2012, S. 25). Es gibt mehrere bekannte Digitalisierungsprojekte, die beispielsweise auch Daten für die NER zur Verfügung stellen. *Europeana*³¹ ist ein Portal, das mit tausenden europäischen Archiven, Bibliotheken und Museen zusammenarbeitet. Für die deutsche Sprache ist vor allem das *Deutsche Textarchiv*³² erwähnenswert, da es einen disziplin- und gattungsübergreifenden Grundbestand deutschsprachiger Texte zwischen dem frühen 16. und dem frühen 20. Jahrhundert bereitstellt. Das Hauptziel des DTA lag in der Erstellung eines Kernkorpus. Dieser umfasst um die 1500 Texte aus den Jahren 1600 bis 1900. Außerdem integriert das DTA weitere Texte aus dem Zeitraum von der Mitte des 15. bis ca. zur Mitte des 20. Jahrhunderts als DTA-Erweiterungen. Das Korpus ist frei zugänglich, durchsuchbar und die Texte stehen unter der freien Creative Commons-Lizenz CC BY-SA 4.0 zum Download zur Verfügung. Das *Projekt Gutenberg*³³ ist eine amerikanische Non-Profit Digitalisierungsinitiative und wurde 1971 gegründet. Hier werden frei zugängliche elektronische Bücher zur Verfügung gestellt. Das Projekt fokussiert sich auf bekannte nicht durch Urheberrecht geschützte Titel. Gutenberg-DE³⁴ ist die weltweit größte kostenlose deutschsprachige Volltext-Literatursammlung und wurde 1994 gegründet. *Google Books*³⁵ arbeitet mit Bibliotheken auf der ganzen Welt zusammen und nimmt deren Sammlungen auf. Für die Aufbewahrung und Erhaltung ihrer Bücher bekommen die Bibliotheken eine digitale Kopie jedes gescannten Buches aus ihrer Sammlung. Im Oktober 2019 gab Haimin Lee anlässlich des 15. Jubiläums von Google Books bekannt, dass Google bereits mehr als 40

³¹www.europeana.eu

³²<http://www.deutschestextarchiv.de/>

³³www.gutenberg.org

³⁴<https://www.projekt-gutenberg.org/>

³⁵<https://books.google.at/>

Millionen Bücher in über 400 Sprachen digitalisiert hatte³⁶. Erwähnenswert sind außerdem Perseus³⁷, deren Sammlung die Geschichte, Literatur und Kultur der griechisch-römischen Welt umfasst, und die Sammlung von 75 historischen Korpora von CLARIN³⁸.

Trotzdem gibt es genügend historische Texte, die noch digitalisiert werden müssen, bevor sie für NER-Anwendungen geeignet sind. Abgesehen von der Beschaffung der reinen Texte nennt Piotrowski (2012) noch zwei weitere wichtige Aspekte, die für die Anwendung von NLP auf historische Texte von Interesse sind: Die Konvertierung von einem Medium in ein anderes und die Verwendung von NLP Ressourcen und Tools während der Beschaffung und Vorbereitung der Texte. Die Konvertierung in ein neues Medium ist nicht verlustfrei und es schleichen sich dabei Fehler ein. Der Digitalisierungsprozess und die Qualität der Resultate haben direkten Einfluss auf die weitere Verwendung der Texte. Es gibt kein allgemeines Rezept für die Digitalisierung historischer Texte, es kommt meist auf die Eigenschaften der Dokumente und die Projektbedürfnisse an.

4.5.2. Nested Entities

Unter Nested Entities versteht man Entitäten, die ein oder mehrere Entitäten beinhalten, wie beispielsweise *Universität Graz* oder *Bank von China*. Beide Entitäten bezeichnen Organisationen mit darin verschachtelten Orten. Viele Arbeiten, die sich mit NER beschäftigen, haben verschachtelte Entitäten fast ignoriert und stattdessen den Fokus auf äußere Entitäten gelegt (Finkel & Manning, 2009). Finkel und Manning (2009) gehen davon aus, dass das nicht aus ideologischen sondern praktischen Gründen so gehandhabt wurde. Die oft verwendeten NER-Korpora CoNLL (Tjong Kim Sang, Erik F., 2002), MUC-6

³⁶<https://www.blog.google/products/search/15-years-google-books/>

³⁷<http://www.perseus.tufts.edu/hopper/collections>

³⁸<https://www.clarin.eu/resource-families/historical-corpora>

und MUC-7 wurden alle auf der obersten Ebene annotiert, d.h. meist wurde der längste String oder nur die äußerste Entität gekennzeichnet. Das GENIA Korpus enthält Nested Entities, aber das JNLPBA 2004 (Kim et al., 2004), das das Korpus verwendet, entfernte alle verschachtelten Entitäten wieder. Leider sind Nested Entities in so gut wie jedem Text zu finden. So sind 17% der Entitäten des GENIA Korpus in anderen Entitäten eingebettet, im ACE Korpus beinhalten 30% der Sätze verschachtelte Entitäten oder Erwähnungen (vgl. Katiyar & Cardie, 2018, S. 1). Auch in historischen Korpora finden sich oft verschachtelte Entitäten. Vor allem bei Personennamen trifft man häufig auf dieses Problem. So finden sich beispielsweise in der Entität *Herzog Otten von Luneburg* nicht nur der Personennamen *Otten* sondern auch der Ort *Luneburg* und der Titel *Herzog*. Hier stellt sich allerdings die Frage, ob *Luneburg* in diesem Fall möglicherweise auf eine Herrschaft verweist, was nicht das selbe ist wie ein Ort. Hierbei muss man bei der Annotation selbst entscheiden, wie man solche Fälle gerne auszeichnen möchte. Auch sollte man sich überlegen, ob man Titel miteinbeziehen will oder nicht. Wichtig hierbei ist schlussendlich nur die Einheitlichkeit der Annotation.

4.5.3. ältere Sprachstufen / Nicht-standardisierte Sprachen

Jede moderne Sprache hat historische Vorläufer. Viele historische Sprachstufen der aktuellen Sprachen sind dadurch gekennzeichnet, dass sie keine standardisierte Sprache oder Orthographie besitzen. Historische Texte haben drei Grundmerkmale (vgl. Piotrowski, 2012, S. 12). Erstens haben sie eine unterschiedliche Schreibweise, d.h. ihre Orthographie unterscheidet sich in den meisten Fällen von modernen Sprachen. Orthographische Konventionen ändern sich im Laufe der Zeit, außerdem gibt es immer wieder Rechtschreibreformen, wie beispielsweise die Reform der deutschen Rechtschreibung 1996.

Zweitens ist eine nationale standardisierte Orthographie eine relativ neue Entwicklung. Die deutsche Orthographie wurde 1901 standardisiert, davor gab

4. NER und die Anwendung auf historische Texte

es keine Regeln für eine korrekte Schreibweise. Stattdessen gab es verschiedene Schulen der Rechtschreibung oder geschriebene Dialekte und eine viel größere Reichweite akzeptabler Schreibweisen. Daher ist die Schreibweise in historischen Texten - selbst wenn sie aus der selben Periode stammen - sehr unterschiedlich. Dieses Phänomen wird *Schreibvarianz* genannt. In der Linguistik hängen die unterschiedliche Schreibweise und Schreibvarianz mit der diachronen und synchronen Varianz zusammen. Piotrowski (2012) zeigt anhand der Abbildung 4.3 den Prozenwert der auftretenden Schreibvarianten des deutschen Wortes *wird* in einem diachronen Korpus. Um 1530 treten alle vier Schreibweisen auf, das deutet auf eine synchrone Varianz hin. Die diachrone Varianz kann man anhand der Zeitachse erkennen, hier wird die Schreibweise *wird* generell akzeptiert.

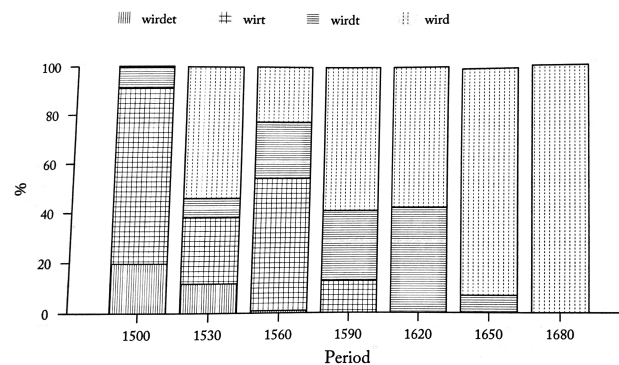


Abbildung 4.3.: Synchrone und diachrone Schreibvariation des deutschen Wortes *wird* (Piotrowski, 2012, S. 13)

Als drittes Merkmal nennt Piotrowski (2012) die Ungewissheit. Im NLP arbeitet man mit digitalisierten Texten, aber die meisten Texte sind nicht in digitaler Form erstellt worden. Die Transkription kann automatisch oder manuell ausgeführt werden. Die wichtigen Punkte hier sind aber, dass jede Transkription Interpretation erfordert und Fehler und Artefakte mitbringt. Wenn man NLP auf historische Texte anwendet, arbeitet man nie mit dem originalen Texten, sondern immer mit einer Form der Interpretation.

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

In diesem Kapitel geht es um die Anwendung und Evaluierung von NER auf deutsche Reiseberichte des 17. Jahrhunderts. In dieser Arbeit wurde öfter angesprochen, dass die Anwendung von NER-Tools auf nicht-standardisierte Sprachen meist keine guten Ergebnisse liefert, da diese für moderne Texte programmiert wurden und werden. Das Ziel dieser Arbeit war herauszufinden, inwieweit man diese Tools auf deutschsprachige historische Texte anwenden kann und welche Ergebnisse sie liefern. Zusätzlich habe ich mich mit der Anwendung von NER mit spaCy genauer auseinandergesetzt. Dabei wurde ein kleines NER-Modell trainiert und dieses auf einen Reisebericht aus dem Jahre 1650 von Johann Georg Metzger angewendet. In den folgenden Abschnitten werden die verwendeten Daten vorgestellt, welche Tools und Methoden angewendet wurden und danach die Ergebnisse evaluiert. Alle verwendeten Daten und der dazugehörige Code befinden sich auf Github unter: https://github.com/jackymore/NER_historical_texts (More, 2021)

5.1. Daten

5.1.1. Testdaten

Für die Testdaten wurde ein Reisebericht von 1649 verwendet: *Die Internuntiaturs des Johann Rudolf Schmid zum Schwarzenhorn (1649): Reisebericht, Instruktionen, Korrespondenz, Bericht* (Metzger, 2019). Das Textkorpus basiert auf Übernahmen von Passagen vorhergehender Reiseberichte, von Berichten und Relationen der Internuntius, Johann Rudolf Schmid zum Schwarzenhorn,

sowie auf Aufzeichnungen und Beobachtungen von Johann Georg Metzger selbst. Der Reisebericht bildet sehr anschaulich die Internuntiaturs des Johann Rufolf Schmid zum Schwarzenhorn nach Konstantinopel zwischen dem 27. Dezember 1648 und dem 29. September 1649 ab. Dabei werden Eindrücke des Autors mit zeitgenössischem, historischem, naturkundlichem und ethnographischem Wissen miteinander verwoben (Huemer, 2019). Der Text wurde bereits im Zuge des Editionsprojekts *Quellen zur habsburgisch-osmanischen Diplomatie in der Neuzeit* (<http://gams.uni-graz.at/context:dipko>) mit TEI ausgezeichnet. Die Daten sind im XML-Format vorhanden und stehen zum Download zur Verfügung. Personen und Orte wurden in folgender Form ausgezeichnet:

```
<rs ref="#id" type="place|person">PERSON/ORT</rs>
```

Beispiel:

```
<rs ref="#johgeomet1" type="person">Iohann Georg von Metzger</rs>
```

5.1.2. Trainingsdaten

Für den Trainingskorpus wurde der Reisebericht „Offt begehrte Beschreibung Der Newen Orientalischen Rejse“ aus dem Jahr 1647 von Adam Olearius aus dem Textkorpus des Deutschen Textarchivs³⁹ entnommen. Der Text wurde mit TEI/XML inklusive dem att.linguistic ausgezeichnet. Dabei handelt es sich um eine Kombination von wort- bzw. tokenbezogenen Informationen aus der automatischen linguistischen Analyse. Dieses Format vereint strukturelle und semantische Annotationen aus dem DTA-Basisformat(DTABf) mit linguistischen Informationen zu Satzsegmentierung (<s> + eindeutige XML-ID) und zu den einzelnen Wörtern bzw. Tokens (<w> + eindeutige XML-ID). Die Informationen umfassen Angaben zur Grundform (@lemma), der Wortart(@pos, notiert gemäß dem im DTA verwendeten STTS-Tagset) sowie der durch CAB

³⁹Version vom 27. Juli 2020: DTA-Kernkorpus und Ergänzungstexte im Format TEI P5 inkl. att.linguistic (5305 Texte, 2,5G)

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

orthographisch normierten Schreibweise (@norm) (Thomas, 2018). Vgl. dazu folgendes Beispiel aus dem Reisebericht „Offt begehrte Beschreibung Der Newen Orientalischen Rejse“:

Was Persien nun ist / der Reusse-Tartarn Land.

```
<s xml:id="sbf_5" prev="#sbf_4">
  <w xml:id="w1aac" lemma="was" pos="PWS" norm="Was">Was</w>
  <w xml:id="w1aad" lemma="Persien" pos="NE" norm="Persien">
    Persien</w>
  <w xml:id="w1aae" lemma="nun" pos="ADV" norm="nun">nun</w>
  <w xml:id="w1aaf" join="right" lemma="sein"
    pos="VAFIN" norm="ist">ist</w>
  <w xml:id="w1ab0" join="left" lemma="/" pos="$(" norm="/">/
</w>
  <w xml:id="w1ab1" lemma="d" pos="ART" norm="der">der</w>
  <w xml:id="w1ab2" lemma="reussen-tartarn" pos="ADJA"
    norm="Reussen-Tartarn">Reussen-Tartarn</w>
  <w xml:id="w1ab3" join="right" lemma="Land"
    pos="NN" norm="Land">Land</w>
  <w xml:id="w1ab4" join="left" lemma="." pos="$." norm=".">.
</w>
</s>
```

Der gesamte Trainingsdatensatz bestand aus 1128 Sätzen, die NEs beinhalteten, und 2693 NEs (751 Ortsnamen, 1942 Personennamen).

5.2. NER Workflow für Standardmodelle

5.2.1. Auswahl der Tools und Methoden

Wenn man NLP Tools mit Python verwendet, kommen meist spaCy⁴⁰ oder NLTK (Natural Language Toolkit)⁴¹ zum Einsatz. Das NLTK bietet eine einfache Oberfläche für über 50 Korpora und lexikalischen Ressourcen, sowie Bibliotheken für Klassifizierung, Tokenisierung, Stemming (Stammformreduktion), Tagging und Parsing. Es kommt vor allem im akademischen Bereich zum Einsatz. SpaCy hingegen wird eher für wirtschaftliche Aufgabenstellungen oder zur Verarbeitung und Analyse von Big Data eingesetzt. SpaCy wird in der Wirtschaft öfter eingesetzt, weil es sehr gut skaliert und man schnell mehrere Aufgaben hintereinander lösen kann. Es weist eine hohe Accuracy bei Standardmodellen auf und liefert sehr gute Resultate beim Parsen und Taggen von Texten (Mattingly, 2020). SpaCy bietet eine große Anzahl an verschiedensten NLP Techniken, dazu zählen Tokenisierung, POS Tagging, Dependency Parsing, Lemmatisierung, NER, EL, SBD (Sentence Boundary Detection), Similarity, Textklassifikation, regelbasiertes Matching, Training und Serialisierung. Eine gute Übersicht der Funktionalitäten von spaCy und NLTK im Vergleich bietet Tabelle 5.1. Die Versuche in dieser Arbeit wurden mit spaCy v2.3 gemacht. Informationen und Anleitungen zu v2.x, können unter folgendem Link gefunden werden: <https://v2.spacy.io/usage/spacy-101>. Im Februar 2021 erschien spaCy v3.0, das mit verschiedensten Verbesserungen und Änderungen im Vergleich zum Vorgänger einhergeht. SpaCy v3.0 bietet mittlerweile Support für über 70 verschiedene Sprachen, Geschwindigkeit auf dem neuesten Stand der Technik, ein produktionsfertiges Trainingssystem, 58 trainierte Pipelines für 18 Sprachen, Support für eigene Modelle in PyTorch, TensorFlow und anderen Frameworks. Das NER-System von spaCy RoBERTa

⁴⁰<https://spacy.io/>

⁴¹<https://www.nltk.org/>

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

	SPACY	NLTK
Programming language	Python	Python
Neural network models	✓	✗
Integrated word vectors	✓	✗
Multi-language support	✓	✓
Tokenization	✓	✓
Part-of-speech tagging	✓	✓
Sentence segmentation	✓	✓
Dependency parsing	✓	✗
Entity recognition	✓	✓
Entity linking	✓	✗
Coreference resolution	✗	✗

Tabelle 5.1.: spaCy vs. NLTK (vgl. <https://v2.spacy.io/usage/facts-figures#comparison>)

(2020) (trainiert anhand des CoNLL-2003 Korpus) verfügt über eine Accuracy von 91,6. Die verschiedenen spaCy Modelle können als Python Pakete installiert werden. Außerdem bietet spaCy drei trainierte NE-Pipelines für das Deutsche: `de_core_news_sm`, `de_core_news_md` und `de_core_news_lg`. Tabelle 5.2 zeigt die Evaluierung der drei Modelle. Für die vorliegende Arbeit wurde das Modell `de_core_news_sm` verwendet. Das Modell hat zwar den niedrigsten F-Score, hat aber eine kleinere Downloadgröße und benötigt daher auch nicht so viel Speicherplatz. Die Trainingsdaten des Modells stammen von TIGER Corpus⁴² und WikiNER⁴³.

⁴²<https://www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/tiger/>

⁴³https://figshare.com/articles/dataset/Learning_multilingual_named_entity_recognition_from_Wikipedia/5462500

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

Modell	Precision	Recall	F-Score	NE-Labels
de_core_news_sm v2	83,92	82,78	83,35	LOC, MISC, ORG, PER
de_core_news_sm v3	0,83	0,81	0,82	LOC, MISC, ORG, PER
de_core_news_md v2	85,31	84,61	84,96	LOC, MISC, ORG, PER
de_core_news_lg v2	86,28	85,94	86,11	LOC, MISC, ORG, PER

Tabelle 5.2.: Evaluierung der deutschen spaCy Modelle (vgl. <https://spacy.io/models/de>)

5.2.2. Anwendung auf Texte des 17. Jahrhunderts

Für die Anwendung der deutschen Standardmodelle von spaCy wurde das Jupyter Notebook⁴⁴ `ner_with_spacy.ipynb` verwendet. Da die Testdaten im XML-Format vorlagen, wurden diese mit dem Modul „Beautiful Soup“⁴⁵ durchsucht. Beautiful Soup ist eine Python Library, die verwendet wird, um Daten aus HTML oder XML Dateien zu extrahieren. Mit diesem Modul ist es möglich, sich durch die Dateien zu navigieren, sie zu durchsuchen und zu modifizieren. Da für die NER nur der Textteil der XML/TEI Daten von Relevanz ist, wurde mithilfe des Moduls nur der Textteil eingelesen. Danach wurde das deutsche Standardmodell von spaCy `de_core_news_sm` (v2) geladen und alle gefundenen NEs mithilfe des Moduls `pandas`⁴⁶ zur besseren Übersicht tabellarisch dargestellt, vgl. Abbildung 5.1. Es wurden insgesamt 3694 Entitäten gefunden. Die ersten 20 Ergebnisse im Vergleich zu den annotierten Testdaten zeigt Abbildung 5.2.

5.2.3. Evaluierung

Das Modell erkannte zwar einige Entitäten richtig, vor allem bei Orten erzielte es gute Ergebnisse. Bei Personennamen hingegen funktionierte es nicht ganz

⁴⁴„Jupyter Notebook“ ist eine Open-Source-Web-Applikation für Dokumente, die Code, Visualisierungen und Texte enthalten und damit eine übersichtliche Dokumentation einzelner Codeabschnitte ermöglicht (vgl. „Project Jupyter“, 08.02.2021)

⁴⁵<https://pypi.org/project/beautifulsoup4/>

⁴⁶<https://pandas.pydata.org/>

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

```
In [12]: #using the text inside the element body
text = soup.body.get_text()

In [17]: #loading standard nlp model for spacy
nlp = spacy.load("de_core_news_sm")
#nlp = spacy.load("test")
doc = nlp(text)

In [18]: # printing all found entities
data = []
for ent in doc.ents:
    data.append([ent.text, ent.label_])

df = pd.DataFrame(data, columns=['Entity', 'Label'])
df
```

Abbildung 5.1.: Workflow spaCy Standardmodell

so gut. Die Evaluierung des Modells bestätigt diese Erkenntnisse. Mit einem F-Score von 42.04 bei der Erkennung von Ortsnamen und einem F-Score von 9.80 bei Personennamen weist das Modell eindeutig eine bessere Accuracy bei Ortsnamen auf. Im Vergleich wurde das etwas größere deutsche Modell `de_core_news_md` getestet. Damit konnte die Accuracy ein wenig verbessert werden. Beide Modelle lieferten aber vor allem bei verschachtelten Entitäten meist nicht den vollen Personennamen, sondern unterschieden zwischen Person- und Ortsnamen. Außerdem wurden viele Wörter mit dem falschen Label gekennzeichnet oder erst gar nicht erkannt. Die Anwendung des Modells `de_core_news_sm` zeigte einen sehr niedrigen F-Score von 20.21, während `de_core_news_md v2` einen geringfügig höheren F-Score von 24,33 aufwies, vgl. Tabelle 5.3. Außerdem wurde noch das deutsche Standardmodell⁴⁷ von

Modell	Precision	Recall	F-Score	NE-Labels F-Score
de_core_news_sm v2	20,44	19,98	20,21	LOC: 42,04, PER: 9,80
de_core_news_md v2	24,38	24,29	24,33	LOC: 49,59 PER: 13,26

Tabelle 5.3.: Evaluierung des deutschen Standardmodells von spaCy bei der Anwendung auf den frnhd. Goldstandard

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

	Entity	Label
0	herren kreishauptmanns	PER
1	freyherrn von Stiebar	PER
2	Kröllendorf	LOC
3	Wien	LOC
4	Iohann Leopold Metzger	PER
5	kaiserlich-königlicher hofrath	PER
6	Iohannes Georgius	PER
7	Mezburg	PER
8	Johann Georg von Metzburg	PER
9	Metzburg	LOC
10	Niederösterreichischer	PER
11	bankodeputazions rath	PER
12	oberösterreichischer	MISC
13	Georg Mezger von Breysgaw	PER
14	Scydack	PER
15	Meiran	PER
16	Georg von Metzger	PER
17	Freyburg	LOC
18	Georg von Metzger	PER
19	regiments rath	PER

(a)

	0	1
0	freyherrn von Stiebar zu Kröllendorf	{'entities': [(0, 36, 'PER')]}
1	Wien	{'entities': [(0, 4, 'LOC')]}
2	Iohann Leopold Metzger freyherr von Metzburg	{'entities': [(0, 44, 'PER')]}
3	Iohannes Georgius von Mezburg	{'entities': [(0, 29, 'PER')]}
4	Mezburg	{'entities': [(0, 7, 'PER')]}
5	Iohann Georg Mezger von Breysgaw	{'entities': [(0, 32, 'PER')]}
6	Breysgaw	{'entities': [(0, 8, 'LOC')]}
7	Iohann Georg von Metzger	{'entities': [(0, 24, 'PER')]}
8	Freyburg	{'entities': [(0, 8, 'LOC')]}
9	Constantinopel	{'entities': [(0, 14, 'LOC')]}
10	Leopold I	{'entities': [(0, 9, 'PER')]}
11	Iohann Georg Metzger von Metzburg	{'entities': [(0, 33, 'PER')]}
12	Metzburg	{'entities': [(0, 8, 'LOC')]}
13	Brunn	{'entities': [(0, 5, 'LOC')]}
14	Carl Ioseph freyherr von Metzburg	{'entities': [(0, 33, 'PER')]}
15	Christoph Augustin freyherr von Gratz	{'entities': [(0, 37, 'PER')]}
16	Gratz	{'entities': [(0, 5, 'LOC')]}
17	Franz Leopold freyherr von Neapel	{'entities': [(0, 33, 'PER')]}
18	Neapel	{'entities': [(0, 6, 'LOC')]}
19	Koppenhagen	{'entities': [(0, 11, 'LOC')]}

(b)

Abbildung 5.2.: Vergleich spaCy Standardmodell und annotierte Testdaten

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

NLTK getestet. Das Modell basiert auf der Arbeit von Mnaal Faruqui und Sebastian Padó. Es wurde anhand des CoNLL 2003 Korpus trainiert. Da für eine umfangreiche Evaluierung die Daten des Goldstandards ein IOB-Format benötigten, konnte das Modell nicht anhand von Precision, Recall und F-Score bewertet werden. Aber auch mit einem groben Blick über die Ergebnisse kann angenommen werden, dass hier die Ergebnisse sehr ähnlich wie bei den spaCy aussehen, vgl. Abbildung 5.3. Die Personen- und Ortsnamen wurden zum Teil erkannt, aber auch hier traten vor allem bei der Erkennung von verschachtelten Entitäten Probleme auf. So wurde zwar der Name *Iohann Leopold Metzger* richtig als Person identifiziert, jedoch wurde beispielsweise in Zeile 2 *Kröllendorf* als Organisation gekennzeichnet, anstatt der Bezeichnung *freyherrn von Stiebar zu Kröllendorf*.

```
(PERSON Wieder/NNP)
(ORGANIZATION Kröllendorf/NNP)
(GPE Wien/NNP)
(PERSON Iohann/NNP Leopold/NNP Metzger/NNP)
(PERSON Iohannes/NNP Georgius/NNP)
(PERSON Wappen/NNP)
(PERSON Johann/NNP Georg/NNP)
(PERSON Es/NNP)
(GPE Wappens/NNP)
(PERSON Geschlecht/NNP)
(PERSON D/NNP)
(PERSON Iohann/NNP Georg/NNP Mezger/NNP)
(PERSON Tschenzi/NNP)
(PERSON Meiran/NNP)
(PERSON Nota/NNP)
(PERSON Iohann/NNP Georg/NNP)
(GPE Freyburg/NNP)
(PERSON Iohann/NNP Georg/NNP)
(GPE Freyburg/NNP)
(PERSON Iohann/NNP Georg/NNP)
(GSP Constantinopel/NNP)
(PERSON Leopold/NNP)
(PERSON Iohann/NNP Georg/NNP Metzger/NNP)
(ORGANIZATION Constantinopel/NNP)
```

Abbildung 5.3.: NLTK Ergebnisse

⁴⁷<https://stanfordnlp.github.io/CoreNLP/download.html>

5.3. NER Workflow für das Training eines eigenen Modells

Wie in Kapitel 3 gezeigt wurde, gibt es für die NER verschiedenen Algorithmen und Methoden des maschinellen Lernens. Es wäre leider nicht möglich sie alle zu testen, denn das würde den Rahmen dieser Masterarbeit sprengen. Der Algorithmus für das Training eines eigenen NER Modells in dieser Arbeit basiert auf der überwachten Lernmethode. Abbildung 5.4 bildet den Ablauf des Trainings des NER Modells ab. Der folgende Teil geht näher auf die Einzelnen Schritte ein. Vor dem Training eines eigenen NER Modells ist es wichtig, 1) die Daten für den Goldstandard zu evaluieren, 2) die Daten zu annotieren bzw. vorannotierte Daten zu bereinigen, 3) wenn nötig, die Daten in ein geeignetes Trainingsdatenformat zu konvertieren. Danach kann mit dem Training begonnen und das fertige Modell evaluiert werden.

5.3.1. Auswahl des Tools

SpaCy ist stark bei der NER von Standardtexten, also Texten, die keinem spezifischen Bereich angehören. Das NLTK benötigt leider relativ viel Zeit, um Texte zu verarbeiten und ist nicht so gut für Einsteiger geeignet, daher habe ich mich bei der vorliegenden Arbeit für das Training eines eigenen Modells mithilfe von spaCy entschieden, da es außerdem im Gegensatz zu NLTK sehr gut dokumentiert und erklärt wie es funktioniert. Die verwendeten Notebooks basieren auf der Grundlage der von Peter Andorfer erstellten Notebooks für den Workshops „Information Extraction aus frühneuhochdeutschen Texten“⁴⁸ am Zentrum für Informationsmodellierung in Graz.

⁴⁸<https://informationsmodellierung.uni-graz.at/de/neuigkeiten/detail/article/workshop-information-extraction-aus-fruehneuhochdeutschen-texten>

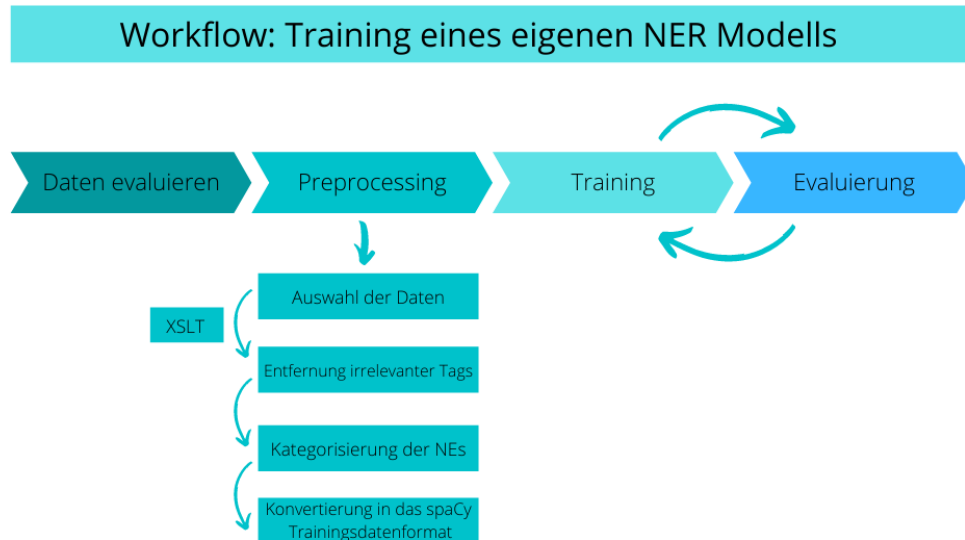


Abbildung 5.4.: Training eines eigenen NER Modells

5.3.2. Vorverarbeitung

Da spaCy ein eigenes Trainingsformat benötigt, mussten die Daten zuerst bearbeitet werden, bevor sie als Trainingsdaten verwendet werden konnten. Leider verfügen die Annotationen des DTA über keine Unterscheidung zwischen einzelnen Kategorien der NEs, wie Orte oder Personen. Die NEs werden unter dem @pos Attribut nur als „NE“ ausgezeichnet. Für das NER-Modell waren nur die Auszeichnungen von NEs in den Trainingsdaten relevant, daher wurde mithilfe von XSLT eine Transformation erstellt, um alle irrelevanten POS-Tags zu entfernen. Danach wurden die NEs per Hand mit den Tags <persName> und <placeName> gekennzeichnet. Für spaCy v2.x wurde allerdings ein bestimmtes Format der Daten benötigt, das folgendermaßen aussehen musste, vgl. <https://v2.spacy.io/usage/training#example-train-ner>:

```
TRAIN_DATA = [  
    ("Who is Shaka Khan?", {"entities": [(7, 17, "PERSON")]}),
```

```
("I like London and Berlin.",  
 {"entities": [(7, 13, "LOC"), (18, 24, "LOC")]}),  
]
```

Um die XML-Daten in das spaCy-Format zu konvertieren, wurde das Modul `acdh-spacytei`⁴⁹ verwendet und das Jupyter Notebook `step1_preprocessing_data_for_NER.ipynb`. Dabei musste die `NER_TAG_MAP` noch auf die Trainingsdaten angepasst werden, vgl. Abbildung 5.5. Beim Durchlaufen des Notebooks wurde der Text zuerst automatisch in Paragraphen und in einem weiteren Schritt in einzelne Sätze geteilt, vgl. Abbildung 5.6. Hier fällt auf, dass viele Sätze, keine NEs beinhalten, dieses Problem wurde im Notebook `step2_train_and_evaluate.ipynb` behoben. Die Sätze ohne NEs wurden entfernt, um die Trainingsdaten zu säubern, vgl. Abbildung 5.7.

map your tei encoding to NE-tags

```
NER_TAG_MAP = {  
    "persName": "PER",  
    "placeName": "LOC",  
}
```

Abbildung 5.5.: Anpassung der Tag Map

```
In [12]: ner_samples[:100]  
Out[12]: [('Daß unser GOTT ein großer Gott ist erlernen wir unter anderen aus dem Buche der vernünftigen Heiden Bibel.',  
          {'entities': []}],  
          ('Dann wenn man die zwei großen Blätter als Himmel und Erden anschaut erscheint des Schöpfers Majestät und Herrlichkeit nicht alleine aus den Wunder Lichtern des Firmaments so stets in ihrer ordentlichen Bewegung und Wirkung gehen sondern auch aus den mancherlei Gütern und Gaben so Gott über den ganzen Kreis der Erden reichlich ausgestreut.',  
          {'entities': []}],  
          ('Daß man in Betrachtung dessen billig mit dem Könige David sich verwundern und ausrufen mag:',  
          {'entities': [(52, 57, 'PER')]}),  
          ('Herr wie sind deine Werke so groß und viel !', {'entities': []}),  
          ('du hast sie alle weißlich geordnet und die Erde ist voll deiner Güter.',  
          {'entities': []}),  
          ('Weil es aber heisst:', {'entities': []}),  
          ('Omnia sunt creata', {'entities': []}),  
          ('propter hominem, & homo Omnia', {'entities': []}),  
          ('sunt creata', {'entities': []}),  
          ('propter hominem, & homo propter Deum.', {'entities': []}),  
          ('propter Deum.', {'entities': []}),  
          ('Das alle Dinge des Menschen wegen erschaffen der Mensch aber wegen des Schöpfers ;',  
          {'entities': []}),  
          ('So will Gott auch das solche Wunderwerke nicht im Verborgnen bleiben sondern von den Menschen betrachtet und Er dadurch erkannt und gelobet werde.',  
          {'entities': []})]
```

Abbildung 5.6.: Teilung der Trainingsdaten in Sätze

⁴⁹<https://pypi.org/project/acdh-spacytei/>

clean train data

e.g. remove all empty examples, use only samples with text length greater 15

```
In [9]: TRAIN_DATA = clean_train_data(TRAIN_DATA, min_ents=1, min_text_len=15, lang=[])
len(TRAIN_DATA)

Out[9]: 1044
```

Abbildung 5.7.: Säuberung der Trainingsdaten

5.3.3. Training

Das Training des Modells wurde mithilfe des Jupyter Notebooks `step2_train_and_evaluate.ipynb` durchgeführt. Dafür wurde wieder das Modul `acdh-spacyte` verwendet. Für das Training wurden die Daten mithilfe von `pandas` im Dateiformat CSV gespeichert. Mithilfe der folgenden Codezeile wurde das Training initialisiert, vgl. Abbildung 5.8. Leider stellte ich hier fest, dass die

initialize training

```
In [31]: batch_train(model='de_core_news_sm', train_data=TRAIN_DATA, output_dir='custom_model')

Loaded model 'de_core_news_sm'
730 train vs 314 test samples
```

Abbildung 5.8.: Training des NER Modells

Trainingsdaten nicht korrekt in das `spaCy`-Format konvertiert wurden, teilweise waren einige Entitäten nicht richtig zugeordnet worden, diese Fehler mussten per Hand in den Trainingsdaten behoben werden. Nach neun Iterationen konnten alle Fehler behoben und das fertige Modell gespeichert werden.

5.3.4. Evaluierung

Das neue Modell lieferte einen F-Score von 71,42, vgl. Tabelle 5.4, ein recht guter Wert für so wenig Trainingsdaten. Die Anwendung des eigenen Modells auf die Testdaten lieferte bessere Ergebnisse bei der Erkennung von einzelnen Entitäten, jedoch konnten mit dem neuen Modell keine verschachtelten Entitä-

5. Anwendung von NER-Tools auf Reiseberichte des 17. Jhds.

ten identifiziert werden, vgl. Notebook `step3_testing_custom_model.ipynb`. Das liegt vermutlich daran, dass die Trainingsdaten keine verschachtelten Entitäten beinhalteten. Um hier bessere Resultate zu erzielen, müssten in den Daten auch verschachtelte Entitäten ausgezeichnet werden.

Modell	Precision	Recall	F-Score	NE-Labels
eigenes Modell	71,63	71,34	71,42	LOC, PER

Tabelle 5.4.: Evaluierung des eigenen Modells

6. Conclusio

Diese Arbeit beschäftigte sich erstens mit der Frage, wie sich die Named Entity Recognition (NER) - ein Teilbereich des Natural Language Processing (NLP) im Laufe der Zeit entwickelt hat und welchen Einfluss sie auf das Forschungsfeld der Digitalen Geisteswissenschaften hat. Beim NLP geht es darum, gesprochene und geschriebene Sprache zu erkennen und zu analysieren. Zahlreiche Tools zur Textanalyse wurden mittlerweile entwickelt, dazu zählen beispielsweise Lemmatisierer, POS-Tagger und Parser. Durch diese digitalen Analysetechniken eröffnen sich für die Digital Humanities (DH) neue Möglichkeiten, um traditionelle qualitative Methoden mit quantitativen, computerbasierten Tools zu verbinden. Mit NLP-Methoden können große Mengen an Texten verarbeitet werden, daher dient NLP als Basis für komplexe Textanalysen. DH-Infrastrukturen wie CLARIN oder DARIAH stellen dafür Tools und Textkorpora zur Verfügung.

Wenn man sich also mit der Entwicklung von NLP und im speziellen der NER in den letzten Jahren befasst, sieht man, dass sich in diesem Bereich extrem viel getan hat. Es wurde gezeigt, dass es je nach Forschungsbereich verschiedene Definitionen und Kategorisierungen für den Begriff „Named Entity“ gibt, wobei die am häufigsten untersuchten Kategorien im NER-Bereich Personennamen, Ortsnamen und Organisationen sind. Die Arbeit hat gezeigt, dass es verschiedene Techniken und Verfahren für die NER gibt, wobei vor allem die Erkennung und Klassifizierung von Entitäten sowie die Disambiguierung und das Linking eine wichtige Rolle in der Forschung spielen. Bei meiner Recherche nach NER-Tools habe ich auch versucht, Tools für Disambiguierung und Linking zu finden und anzuwenden, jedoch gibt es für die Anwendung auf historische Texte noch weniger Möglichkeiten, denn hier sind die technischen Hürden wesentlich größer, als bei der NER.

Die Literaturrecherche hat bewiesen, dass die NER in den Geisteswissen-

6. Conclusio

schaften zwar noch immer unterrepräsentiert ist, mittlerweile aber immer mehr DH Projekte auf moderne NER Methoden und maschinelle Lerntechniken setzen. In der Literaturwissenschaft oder im Bereich digitaler Museen werden beispielsweise Visualisierungstechniken, NER oder Netzwerkanalysen eingesetzt. Es gibt mehrere Gründe, warum automatisierte Methoden in den Geisteswissenschaften nicht eingesetzt werden, wie wenig Trainingsmaterial oder die Anwendungen weichen vom eigentlichen Entwicklungsgrund ab (unterschiedliche Sprachstufen oder Textgenres) und haben dadurch höhere Fehlerquoten.

Will man ein eigenes NER Modell trainieren, kommt man an maschinellen Lernmethoden nicht vorbei. In der Arbeit wurde ein Einblick in verschiedene Methoden des ML gegeben. Je nach Anwendungsgebiet und Forschungsbereich haben andere Methoden Stärken und Schwächen. Erwähnenswert für die NER sind überwachte, unüberwachte, teilüberwachte und bestärkende Lerntechniken sowie Deep Learning. Zusätzlich wurden wichtige Algorithmen angeführt, die einen wesentlichen Beitrag zu NER Modellen liefern und ein besseres Verständnis für die Funktionsweise von NER Methoden geben. Für die Vergleichbarkeit von NER Modellen ist ein Evaluierungsprozess nötig, daher wurden wichtige Evaluierungsmetriken wie Precision, Recall und der F-Score besprochen.

Wichtig für die Arbeit mit NER und das Trainieren eigener Modelle ist die Annotation der Trainingsdaten. Es gibt verschiedenste Möglichkeiten Texte mit NEs auszuzeichnen. Die Recherche hat gezeigt, dass es nicht nur kostenpflichtige, sondern auch sehr gute Open-Source-Programme für Annotationen zur Verfügung stehen. Um ein Modell trainieren zu können ist es außerdem wichtig zu verstehen, was Trainings-, Validierungs- und Testdaten sind und wie man ein Modell evaluieren kann.

Für die DH wurden in den letzten Jahren immer wieder Modelle und Tools entwickelt, die die Anwendung von NER auf historische Texte oder große Textkorpora einfacher machen sollen. Standardmodelle sind in den meisten Fällen nicht für Anwendungsfälle in den DH geeignet. Das liegt vor allem daran, dass Standardmodelle oft an modernen Webtexten, wie beispielsweise Wikipe-

6. Conclusio

dia, trainiert wurden und werden. Probleme, die auch bei modernen Texten auftreten, wie verschachtelte Entitäten, sind auch bei historischen Texten noch immer ein Thema. Vor allem aber besitzen historische Sprachen wenig bis gar keine Standardisierung in der Orthographie, daher treten oft Komplikationen mit Modellen auf, die nicht eigens dafür trainiert wurden. Die Ergebnisse dieser Arbeit bestätigen diese Behauptung. Als Fallstudie wurde das Verhalten von NER Modellen an einem Reisebericht aus dem 17. Jh. genauer analysiert. Dabei wurden drei deutsche Standardmodelle angewendet. Dabei hat sich gezeigt, dass die deutschen Modelle von spaCy nur einen F-Score von 20,44 und 24,38 erreichten. Auch das deutsche NLTK-Modell zeigte keine vielversprechenden Ergebnisse. Um die Resultate zu verbessern, wurde der Versuch unternommen, ein eigenes Modell mithilfe von spaCy zu trainieren. Dieses Modell zeigte viel größeres Potenzial, da es einen F-Score von 71,42 erreichte, vgl. Tabelle 6.1. Bei

Modell	Precision	Recall	F-Score	NE-Labels
eigenes Modell	71,6	71,3	71,42	LOC, PER
de_core_news_sm v2	20,44	19,98	20,21	LOC, PER
de_core_news_md v2	24,38	24,29	24,33	LOC, PER

Tabelle 6.1.: Vergleich der Ergebnisse des eigenen Modells mit den Standardmodellen

diesen Versuchen handelte es sich um eine Fallstudie, die sich auf Reiseberichte beschränkt, es kann jedoch angenommen werden, dass die Ergebnisse vermutlich auch für andere Texttypen des 17. Jhds. Gültigkeit haben. Wie aber schon öfters in dieser Arbeit erwähnt wurde, benötigt man eine große Menge an ausgezeichneten Trainingsdaten, um eine gute Accuracy zu erreichen, was allerdings nur in einem größeren Projekt mit mehr Ressourcen möglich wäre, als es in der vorliegenden Arbeit der Fall war. Das TEI/XML Format scheint zudem für viele Trainingspipelines nicht gut geeignet zu sein, hier wären Auszeichnungen im IOB/BILUO-Format von größerem Vorteil. Vor allem die Konvertierung von XML-Daten in ein spezielles Trainingsdatenformat wie spaCy ist meist extrem umständlich. SpaCy beinhaltet eine automatische Konvertierung in

das eigene JSON-Trainingsformat, aber nur für Daten, die in IOB oder dem CoNLL-Format kodiert wurden. In den DH, vor allem im Bereich Digitaler Editionen, werden aber hauptsächlich TEI Daten verwendet. Im Zuge des Projekts NERDPool⁵⁰ wird bereits an einer Pipeline gearbeitet, die auch TEI Daten verarbeiten kann. Die Fertigstellung dieses Projekts wird einen großen Beitrag zur NER in der DH Community liefern. NER-Tools, die extra für die DH Community erstellt wurden, sind nicht wesentlich besser oder einfacher in ihrer Anwendung. So benötigt beispielsweise der Humanity Entity Recognizer ein Linux bzw. Mac-Betriebssystem und auch die Einarbeitungszeit ist hier nicht viel kürzer als bei spaCy. Auch WebLicht ist für die NER von größeren Texten oder Textkorpora nicht geeignet, da es eine webbasierte Anwendung ist und die Dateigröße für den Input limitiert ist.

Für die zukünftige Arbeit mit NER und Texten aus dem 17. Jahrhundert könnte das Trainingskorpus erweitert werden, indem andere Textarten inkludiert werden. Die Texte aus dem DTA bieten eine gute Trainingsgrundlage, allerdings wäre es für die NER wichtig, dass die einzelnen NEs kategorisiert und verschachtelte Entitäten ausgezeichnet werden. Von Vorteil wäre auch eine einfache Konvertierungsmöglichkeit von XML/TEI Dateien in das IOB oder CoNLL-Format. Die in dieser Arbeit gezeigten Ergebnisse lassen abschließend die Aussage zu, dass es mit spaCy möglich ist, ein eigenes Modell für frühneu-hochdeutsche Texte des 17. Jahrhunderts zu trainieren. Auf Grenzen stößt man vor allem bei der Annotation und der Vorverarbeitung der Daten, da Zeit- und Bearbeitungsaufwand sehr hoch sind. Einen Lichtblick bietet das unüberwachte Lernen, denn mit mehr Daten wird dies zu einer Option und man kann sich das händische Annotieren der Trainingsdaten damit sparen.

⁵⁰<https://nerdpool.acdh-dev.oeaw.ac.at/>

Appendix

Anhang A.

NLP-Tools

96

Tool	Autoren	Jahr	Link
spaCy : free, open-source library for advanced Natural Language Processing (NLP) in Python.	Explosion AI	2016-2020	https://spacy.io/
NLTK : Natural Language Tool-kit	Bird et. al.	2009	https://www.nltk.org/
Stanford CoreNLP : Natural language software	Manning et al.	2014	https://stanfordnlp.github.io/CoreNLP/
HER (Humanities Entity Recognizer): robust, practical, efficient Named Entity Recognition for today's digital humanist	Erdmann et al.	2019	https://github.com/alexerdmann/HER/
WebLicht : Web-Based Linguistic Chaining Tool	CLARIN-D/SfS-Uni-Tübingen	2012	https://weblicht.sfs.uni-tuebingen.de/weblichtwiki/

Tool	Autoren	Jahr	Link
TAGME	A3 lab	2014	https://tagme.d4science.org/tagme/
Entity-Fishing	Patrice Lopez	2019	https://github.com/kermitt2/entity-fishing
DBpedia Spotlight	Universität Berlin, Universität Leipzig	2011-2017	https://www.dbpedia-spotlight.org/
Watson Natural Language Understanding: The natural language processing (NLP) service for advanced text analytics	IBM	2019	https://www.ibm.com/cloud/watson-natural-language-understanding
microNER: A micro-service for Named Entity Recognition based on Docker	Wiedemann et al.	2018	https://uhh-lt.github.io/microNER/

Tabelle A.1.

Literatur

- Adel, H., Roth, B. & Schütze, H. (2016). Comparing Convolutional Neural Networks to Traditional Models for Slot Filling. <https://arxiv.org/pdf/1603.05157>. (Siehe S. 11)
- Al-Moslmi, T., Gallofre Ocana, M., L. Opdahl, A. & Veres, C. (2020). Named Entity Extraction for Knowledge Graphs: A Literature Overview: *IEEE Access*, 8, 32862–32881. *IEEE Access*, 8, 32862–32881. <https://doi.org/10.1109/ACCESS.2020.2973928> (siehe S. 20–22, 28–30, 32, 33)
- Alpaydin, E. (2014). *Introduction to Machine Learning* (Bd. Third edition). The MIT Press. <http://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=836612&site=ehost-live>. (Siehe S. 62)
- Benikova, D., Biemann, C., Santhanam, P. & Yimam, S. M. (2015). GermaNER: Free Open German Named Entity Recognition Tool. *GSCL* (siehe S. 49).
- Bikel, D. M., Schwartz, R. & Weischedel, R. M. (1999). An Algorithm that Learns What’s in a Name. *Machine Learning*, 34(1/3), 211–231. <https://doi.org/10.1023/A:1007558221122> (siehe S. 47)
- Brando, C., Frontini, F. & Ganascia, J.-G. (2016). REDEN: Named Entity Linking in Digital Literary Editions Using Linked Data Sets. *Complex Systems Informatics and Modeling Quarterly*, (7), 60–80. <https://doi.org/10.7250/csimq.2016-7.04> (siehe S. 31)
- Campos, D., Matos, S. & Luis, J. (2012). Biomedical Named Entity Recognition: A Survey of Machine-Learning Tools. *Sakurai (Hg.) – Theory and Applications for Advanced*. <https://doi.org/10.5772/51066>. (Siehe S. 15)
- Carraggi, A., Cornia, M., Baraldi, L. & Cucchiara, R. (2018). Visual-Semantic Alignment Across Domains Using a Semi-Supervised Approach. *Proceedings of the European Conference on Computer Vision (ECCV) Workshops* (siehe S. 54).

- Chen, L., Chen, J., Zou, Q., Huang, K. & Li, Q. (2017). Multi-View Feature Combination for Ancient Paintings Chronological Classification. *Journal on Computing and Cultural Heritage*, 10(2), 1–15. <https://doi.org/10.1145/3003435> (siehe S. 53)
- Chinchor, N. & Robinson, P. (1998). Appendix E: MUC-7 Named Entity Task Definition (version 3.5). *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998*. <https://www.aclweb.org/anthology/M98-1028> (siehe S. 9, 10, 58)
- CLARIN-D/SfS-Uni. Tübingen. (2012). WebLicht: Web-Based Linguistic Chaining Tool. (Siehe S. 64, 65).
- Daiber, J., Jakob, M., Hokamp, C. & Mendes, P. N. (2013). Improving Efficiency and Accuracy in Multilingual Entity Extraction. *Proceedings of the 9th International Conference on Semantic Systems*, 121–124. <https://doi.org/10.1145/2506182.2506198> (siehe S. 28)
- Dat Ba Nguyen, Johannes Hoffart, Martin Theobald & Gerhard Weikum. (2014). AIDA-light: High-Throughput Named-Entity Disambiguation. In Christian Bizer, Tom Heath, Sören Auer & Tim Berners-Lee (Hrsg.), *Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014), Seoul, Korea, April 8, 2014*. CEUR-WS.org. http://ceur-ws.org/Vol-1184/ldow2014_paper_p03.pdf. (Siehe S. 24, 32–34)
- Deltorn, J.-M. (2017). Deep Creations: Intellectual Property and the Automata. *Frontiers in Digital Humanities*, 4. <https://doi.org/10.3389/fdigh.2017.00003> (siehe S. 29)
- Dredze, M., McNamee, P., Rao, D., Gerber, A. & Finin, T. (2010). Entity Disambiguation for Knowledge Base Population. *Proceedings of the 23rd International Conference on Computational Linguistics*, 277–285 (siehe S. 23).

- Ehrmann, M. (2017). *Les Entités Nommées, de la linguistique au TAL : Statut théorique et méthodes de désambiguïsation* (Diss.). Paris Diderot University. <https://hal.archives-ouvertes.fr/tel-01639190>. (Siehe S. 14)
- Ehrmann, M., Nouvel, D. & Rosset, S. (2016). Named Entity Resources - Overview and Outlook. *Language Resources and Evaluation*. <https://hal-inalco.archives-ouvertes.fr/hal-01359441> (siehe S. 19)
- Elgammal, A., Kang, Y. & Leeuw, M. D. (2018). Picasso, matisse, or a fake? Automated analysis of drawings at the stroke level for attribution and authentication. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*. <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85060477681&partnerID=40&md5=45bcbb89c44561249c6b8616e9884fc> (siehe S. 50)
- EM-Algorithmus – Wikipedia. (09.11.2020). <https://de.wikipedia.org/wiki/EM-Algorithmus>. (Siehe S. 35)
- Erdmann, A., Wrisley, D. J., Allen, B., Brown, C., Cohen-Bodénès, S., Elsner, M., Feng, Y., Joseph, B., Joyeux-Prunel, B. & de Marneffe, M.-C. (2019). Practical, Efficient, and Customizable Active Learning for Named Entity Recognition in the Digital Humanities. Verfügbar 2. Februar 2020 unter <https://www.aclweb.org/anthology/N19-1231.pdf> (siehe S. 64–66)
- Eshel, Y., Cohen, N., Radinsky, K., Markovitch, S., Yamada, I. & Levy, O. (2017). Named Entity Disambiguation for Noisy Text. <https://arxiv.org/pdf/1706.09147>. (Siehe S. 29)
- Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S. & Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. *Artificial intelligence*, 165(1), 91–134 (siehe S. 22).
- Fang, W., Zhang, J., Wang, D., Chen, Z. & Li, M. (2016). Entity Disambiguation by Knowledge and Text Jointly Embedding. *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 260–269. <https://doi.org/10.18653/v1/K16-1026> (siehe S. 28)

- Finkel, J. R. & Manning, C. D. (2009). Nested Named Entity Recognition. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 141–150. <https://www.aclweb.org/anthology/D09-1015> (siehe S. 49, 68)
- Fiorucci, M., Khoroshiltseva, M., Pontil, M., Traviglia, A., Del Bue, A. & James, S. (2020). Machine Learning for Cultural Heritage: A Survey. *Pattern Recognition Letters*, 133, 102–108. <https://doi.org/10.1016/j.patrec.2020.02.017> (siehe S. 42, 50)
- Fleischman, M. & Hovy, E. (2002). Fine grained classification of named entities. In Unknown (Hrsg.), *Proceedings of the 19th international conference on Computational linguistics* - (S. 1–7). Association for Computational Linguistics. <https://doi.org/10.3115/1072228.1072358>. (Siehe S. 13)
- Franklin, S. & Graesser, A. C. (1996). Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents. In Jörg P. Müller, Michael J. Wooldridge & Nicholas R. Jennings (Hrsg.), *Intelligent Agents III, Agent Theories, Architectures, and Languages, ECAI '96 Workshop (ATAL), Budapest, Hungary, August 12-13, 1996, Proceedings* (S. 21–35). Springer. <https://doi.org/10.1007/BFb0013570>. (Siehe S. 54)
- Frochte, J. (2020). *Maschinelles Lernen: Grundlagen und Algorithmen in Python*. Carl Hanser Verlag GmbH & Company KG. <https://books.google.at/books?id=oOMHEAAAQBAJ>. (Siehe S. 42, 44, 52, 53)
- Frontini, F., Brando, C. & Ganascia, J.-G. (2015). Semantic Web Based Named Entity Linking for Digital Humanities and Heritage Texts. In Arnaud Zucker, Isabelle Draelants, Catherine Faron Zucker & Alexandre Monnin (Hrsg.), *First International Workshop Semantic Web for Scientific Heritage at the 12th ESWC 2015 Conference*. <https://hal.archives-ouvertes.fr/hal-01203358>. (Siehe S. 23, 25, 28, 30–33)
- Gandhi, R. (07.06.2018). Support Vector Machine — Introduction to Machine Learning Algorithms. *Towards Data Science*. <https://towardsdatasci>

- ence.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 (siehe S. 49, 50)
- García Rivero, D., Jurado Núñez, J. M. & Taylor, R. (2016). Bell Beaker and the evolution of resource management strategies in the southwest of the Iberian Peninsula. *Journal of Archaeological Science*, 72, 10–24. <https://doi.org/10.1016/j.jas.2016.05.012> (siehe S. 42)
- Goyal, A., Gupta, V. & Kumar, M. (2018). Recent Named Entity Recognition and Classification techniques: A systematic review. *Computer Science Review*, 29, 21–43. <https://doi.org/10.1016/j.cosrev.2018.06.001> (siehe S. 22)
- Grishman, R. & Sundheim, B. (1996). Message Understanding Conference-6. In J. Tsujii (Hrsg.), *Proceedings of the 16th conference on Computational linguistics* - (S. 466). Association for Computational Linguistics. <https://doi.org/10.3115/992628.992709>. (Siehe S. 9)
- Gultepe, E., Conturo, T. E. & Makrehchi, M. (2018). Predicting and Grouping Digitized Paintings by Style using Unsupervised Feature Learning. *Journal of Cultural Heritage*, 31, 13–23. <https://doi.org/10.1016/j.culher.2017.11.008> (siehe S. 53)
- Habib, M. B. & van Keulen, M. (2016). TwitterNEED: A hybrid approach for named entity extraction and disambiguation for tweet. *Natural Language Engineering*, 22(3), 423–456. <https://doi.org/10.1017/S1351324915000194> (siehe S. 30)
- Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi & Xu Liang. (2018). doccano: Text Annotation Tool for Human. <https://github.com/doccano/doccano>. (Siehe S. 60)
- Horne, R. (2020). Beyond lists: digital gazetteers and digital history. *The Historian*, 82(1), 37–50. <https://doi.org/10.1080/00182370.2020.1725992> (siehe S. 25)
- Hu, S., Tan, Z., Zeng, W., Ge, B. & Xiao, W. (2019). Entity Linking via Symmetrical Attention-Based Neural Network and Entity Structural

- Features. *Symmetry*, 11(4), 453. <https://doi.org/10.3390/sym11040453> (siehe S. 26, 29)
- Huemer, A. (2019). „Von Wien in Österreich nach Constantinopel.“ Das Reisediarium Johann Georg Metzgers aus dem Jahre 1650. Eine Einleitung. In A. Strohmeyer & G. Vogeler (Hrsg.), *Quellen zur habsburgisch-osmanischen Diplomatie in der Neuzeit. Die Internuntiaturs des Johann Rudolf Schmid zum Schwarzenhorn (1649): Reisebericht, Instruktionen, Korrespondenz, Berichte*. (Siehe S. 72).
- Hurwitz, J. & Kirsch, D. (2018). Machine Learning For Dummies, IBM Limited Edition. <https://www.ibm.com/downloads/cas/GB8ZMQZ3> (siehe S. 38–40)
- Inan, E. & Dikenelli, O. (2018). A Sequence Learning Method for Domain-Specific Entity Linking. *Proceedings of the Seventh Named Entities Workshop*, 14–21. <https://doi.org/10.18653/v1/W18-2403> (siehe S. 23)
- Iyyer, M., Guha, A., Chaturvedi, S., Boyd-Graber, J. & Daumé III, H. (2016). Feuding families and former friends: Unsupervised learning for dynamic fictional relationships. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1534–1544 (siehe S. 56).
- Jacobs, P. S. & Rau, L. F. (1990). SCISOR: Extracting Information from on-Line News. *Digital Scholarship in the Humanities*, 33(11), 88–97. <https://doi.org/10.1145/92755.92769> (siehe S. 9)
- Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho & Iryna Gurevych. (2018). The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation. *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, 5–9. <http://tubiblio.ulb.tu-darmstadt.de/106270/> (siehe S. 60)

- Jannidis, F., Kohle, H. & Rehbein, M. (Hrsg.). (2017). *Digital Humanities : eine Einführung*. Stuttgart : J.B. Metzler Verlag. <https://permalink.obvsg.at/UGR/AC12324948>. (Siehe S. 3, 41, 51)
- Jonnalagadda, S., Leaman, R., Cohen, T. & Gonzalez, G. (2010). A Distributional Semantics Approach to Simultaneous Recognition of Multiple Classes of Named Entities. In A. Gelbukh (Hrsg.), *Computational linguistics and intelligent text processing* (S. 224–235). Springer. https://doi.org/10.1007/978-3-642-12116-6_19. (Siehe S. 42)
- Jungermann, F. (2006). Named entity recognition mit conditional random fields. *Computer Science, University of Dortmund, Diplomarbeit* (siehe S. 47, 48).
- Katiyar, A. & Cardie, C. (2018). Nested Named Entity Recognition Revisited. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 861–871. <https://doi.org/10.18653/v1/N18-1079> (siehe S. 69)
- Kim, J.-D., Ohta, T., Tsuruoka, Y., Tateisi, Y. & Collier, N. (2004). Introduction to the bio-entity recognition task at JNLPBA. In N. Collier, P. Ruch & A. Nazarenko (Hrsg.), *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications - JNLPBA '04* (S. 70). Association for Computational Linguistics. <https://doi.org/10.3115/1567594.1567610>. (Siehe S. 69)
- Klinke, H. (2017). Information Retrieval. In F. Jannidis, H. Kohle & M. Rehbein (Hrsg.), *Digital Humanities : eine Einführung* (S. 268–278). Stuttgart : J.B. Metzler Verlag. (Siehe S. 62).
- Kostadinova-Avramova, M., Jordanova, N., Jordanova, D., Grigorov, V., Lesigarski, D., Dimitrov, P. & Bozhinova, E. (2018). Firing temperatures of ceramics from Bulgaria determined by rock-magnetic studies. *Journal of Archaeological Science: Reports*, 17, 617–633. <https://doi.org/10.1016/j.jasrep.2017.12.021> (siehe S. 53)

- Kripke, S. A. (1981). *Naming and necessity* (2. print). Harvard Univ. Press. (Siehe S. 13).
- Kuhn, J. (2019). Computational text analysis within the Humanities: How to combine working practices from the contributing fields? *Language Resources and Evaluation*, 53(4), 565–602. <https://doi.org/10.1007/s10579-019-09459-3> (siehe S. 1, 2, 5)
- Kurz, S. (2015). Fortgeschrittene Web-Basics. In S. Kurz (Hrsg.), *Digital humanities* (S. 3–116). Springer Vieweg. https://doi.org/10.1007/978-3-658-05793-0_2. (Siehe S. 58)
- L. R. Rabiner. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 257–286. <https://doi.org/10.1109/5.18626> (siehe S. 46, 47)
- Lafferty, J., McCall, A. & Pereira, F. C. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *International Conference on Machine Learning*. https://repository.upenn.edu/cgi/viewcontent.cgi?article=1162&context=cis_papers (siehe S. 46, 48)
- LaTeX Ninja'ing and the Digital Humanities. (2020). Machine Learning for the Humanities: A very short introduction and a not-so-short reflection. <https://latex-ninja.com/2020/10/25/machine-learning-for-the-humanities-a-very-short-introduction-and-a-not-so-short-reflection/>. (Siehe S. 55)
- Li, J., Sun, A., Han, J. & Li, C. (2018). A Survey on Deep Learning for Named Entity Recognition. <https://arxiv.org/pdf/1812.09449>. (Siehe S. 20, 22)
- Lorenz, U. (2020). Bestärkendes Lernen als Teilgebiet des Maschinellen Lernens. *Reinforcement Learning: Aktuelle Ansätze verstehen - mit Beispielen in Java und Greenfoot* (S. 1–11). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-61651-2_1. (Siehe S. 55)
- Luo, G., Huang, X., Lin, C.-Y. & Nie, Z. (2015). Joint Entity Recognition and Disambiguation. *Proceedings of the 2015 Conference on Empirical*

- Methods in Natural Language Processing*, 879–888. <https://doi.org/10.18653/v1/D15-1104> (siehe S. 30)
- Machine Learning. (2020). <https://www.ibm.com/de-de/analytics/machine-learning>. (Siehe S. 51, 55)
- Marrero, M., Urbano, J., Sánchez-Cuadrado, S., Morato, J. & Gómez-Berbís, J. M. (2013). Named Entity Recognition: Fallacies, challenges and opportunities. *Computer Standards & Interfaces*, 35(5), 482–489. <https://doi.org/10.1016/j.csi.2012.09.004> (siehe S. 15–17)
- Marshall, C. (2020). What is named entity recognition (NER) and how can I use it? Verfügbar 4. Juli 2020 unter <https://medium.com/mysuperaai/what-is-named-entity-recognition-ner-and-how-can-i-use-it-2b68cf6f545d>. (Siehe S. 11)
- Mattingly, W. . B. (2020). Named Entity Recognition Series: Lesson 01 - Introduction to NER. (Siehe S. 1, 2, 74).
- McCulloch, W. S. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4), 115–133. <https://doi.org/10.1007/BF02478259> (siehe S. 38)
- Metzger, J. G. (2019). Itinerarium oder rayss beschreibung von Wien in Österreich nach Constantinopel [...] (1650): bearbeitet von Anna Huemer, Kommentar von Lisa Brunner / Anna Spitzbart, Digitalisierung von Carina Koch / Jakob Sonnberger, unter Mitarbeit von Anna Huemer. In A. Strohmeyer & G. Vogeler (Hrsg.), *Quellen zur habsburgisch-osmanischen Diplomatie in der Neuzeit. Die Internuntiaturs des Johann Rudolf Schmid zum Schwarzenhorn (1649): Reisebericht, Instruktionen, Korrespondenz, Berichte*. (Siehe S. 71).
- More, J. (2021). NER_historcal_texts. https://github.com/jackymore/NER_historical_texts. (Siehe S. 71)
- Nadeau, D. & Sekine, S. (2007). Named Entities: Recognition, classification and use. *Linguisticae Investigationes*, 30(1), 3–26. <https://doi.org/10.1075/li.30.1.03nad> (siehe S. 8, 13, 20, 54)

- Named entity - Wikipedia. (20.08.2020). https://en.wikipedia.org/wiki/Named_entity. (Siehe S. 14)
- News Center Microsoft Deutschland. (2020). Microsoft erklärt: Was ist Deep Learning? Definition & Funktionen von DL | News Center Microsoft. <https://news.microsoft.com/de-de/microsoft-erklaert-was-ist-deep-learning-definition-funktionen-von-dl/>. (Siehe S. 22)
- Nguyen, D. B., Theobald, M. & Weikum, G. (2016). J-NERD: Joint Named Entity Recognition and Disambiguation with Rich Linguistic Features. *Transactions of the Association for Computational Linguistics*, 4, 215–229. https://doi.org/10.1162/tacl_a_00094 (siehe S. 30)
- Nie, F., Cao, Y., Wang, J., Lin, C.-Y. & Pan, R. (2018). Mention and Entity Description Co-Attention for Entity Disambiguation. *AAAI* (siehe S. 29).
- Nouvel, D. (2016). *Named entities for computational linguistics*. ISTE. <http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&AN=1140972>. (Siehe S. 8–11, 38, 39)
- Phan, C. M. (2019). *Named entity recognition and linking with knowledge base* (Diss.). Nanyang Technological University. <https://dr.ntu.edu.sg/handle/10356/136585?mode=simple>. (Siehe S. 23)
- Phan, M. C., Sun, A., Tay, Y., Han, J. & Li, C. (2017). NeuPL. In E.-P. Lim, M. Winslett, M. Sanderson, A. Fu, J. Sun, S. Culpepper, E. Lo, J. Ho, D. Donato, R. Agrawal, Y. Zheng, C. Castillo, A. Sun, V. S. Tseng & C. Li (Hrsg.), *CIKM 2017* (S. 1667–1676). ACM Association for Computing Machinery. <https://doi.org/10.1145/3132847.3132963>. (Siehe S. 29)
- Piotrowski, M. (2012). *Natural language processing for historical texts* (Bd. 17). Morgan & Claypool. (Siehe S. 2, 4, 67–70).
- Polak, A., Kelman, T., Murray, P., Marshall, S., Stothard, D. J., Eastaugh, N. & Eastaugh, F. (2017). Hyperspectral imaging combined with data classification techniques as an aid for artwork authentication. *Journal of Cultural Heritage*, 26, 1–11. <https://doi.org/10.1016/j.culher.2017.01.013> (siehe S. 50)

- Project Jupyter. (08.02.2021). <https://jupyter.org/index.html>. (Siehe S. 76)
- Pustejovsky, J. & Stubbs, A. (2012). *Natural language annotation for machine learning*. O'Reilly. <http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10766860>. (Siehe S. 45, 46, 58, 61–63)
- Ramshaw, L. A. & Marcus, M. P. (1995). Text Chunking using Transformation-Based Learning. *CoRR*, *cmp-lg/9505040* (siehe S. 59).
- Raschka, S. & Mirjalili, V. (2017). *Python Machine Learning - Second Edition* (Bd. 2nd ed). Packt Publishing. <http://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1606531&site=ehost-live>. (Siehe S. 41, 43–45, 51, 52, 54, 55)
- Ratinov, L. & Roth, D. (2009). Design Challenges and Misconceptions in Named Entity Recognition. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, 147–155 (siehe S. 59).
- Riedl, M. & Padó, S. (2018). A Named Entity Recognition Shootout for German. In I. Gurevych & Y. Miyao (Hrsg.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (S. 120–125). Association for Computational Linguistics. <https://doi.org/10.18653/v1/P18-2020>. (Siehe S. 36, 48, 49)
- Rouse, M. (n. d.). named entity. *TechTarget, 2010*. <https://searchbusinessanalytics.techtarget.com/definition/named-entity> (siehe S. 15)
- Ruokolainen, T. & Kettunen, K. (2020). Name the Name – Named Entity Recognition in OCREd 19th and Early 20th Century Finnish Newspaper and Journal Collection Data. <http://ceur-ws.org/Vol-2612/paper10.pdf> (siehe S. 34)
- Saleh, B. & Elgammal, A. (n. d.). Large-scale Classification of Fine-Art Paintings: Learning The Right Metric on The Right Feature. <https://arxiv.org/pdf/1505.00855>. (Siehe S. 50)
- Sánchez, J. A., Bosch, V., Romero, V., Depuydt, K. & de Does, J. (2014). Handwritten text recognition for historical documents in the transcriptorium project. In A. Antonacopoulos & K. U. Schulz (Hrsg.),

- Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage - DATeCH '14* (S. 111–117). ACM Press. <https://doi.org/10.1145/2595188.2595193>. (Siehe S. 4)
- Schöch, C. (2017). Quantitative Analyse. In F. Jannidis, H. Kohle & M. Rehbein (Hrsg.), *Digital Humanities : eine Einführung* (S. 279–298). Stuttgart : J.B. Metzler Verlag. (Siehe S. 21).
- Sharnagat, R. (2014). Named entity recognition: A literature survey. *Center For Indian Language Technology* (siehe S. 8, 20, 22, 42, 51, 53).
- Shen, W., Wang, J. & Han, J. (2015). Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions: *IEEE Transactions on Knowledge and Data Engineering*, 27(2), 443–460. *IEEE Transactions on Knowledge and Data Engineering*, 27(2), 443–460. <https://doi.org/10.1109/TKDE.2014.2327028> (siehe S. 23, 24, 26, 31)
- Shen, X., Efros, A. A. & Aubry, M. (2019). Discovering Visual Patterns in Art Collections with Spatially-consistent Feature Learning. <https://arxiv.org/pdf/1903.02678>. (Siehe S. 54)
- Sherzod Hakimov, H. T. Horst, Soufian Jebbara, Matthias Hartung & P. Cimiano. (2016). Combining Textual and Graph-Based Features for Named Entity Disambiguation Using Undirected Probabilistic Graphical Models. *undefined*. <https://www.semanticscholar.org/paper/Combining-Textual-and-Graph-Based-Features-for-Hakimov-Horst/8148a33273173b52458eef840449bf2155817b84> (siehe S. 29)
- Shott, M. J. & Habtzghi, D. (2016). Toward disentangling stages in mixed assemblages of flake debris from biface reduction: An experimental approach. *Journal of Archaeological Science*, 70, 172–180. <https://doi.org/10.1016/j.jas.2016.04.023> (siehe S. 42)
- Simon, R., Isaksen, L., Barker, E. & de Soto Cañamares, P. (2016). The Pleiades Gazetteer and the Pelagios Project. In M. L. Berman, R. Mostern & H. Southall (Hrsg.), *Placing Names* (S. 97–109). Indiana University Press. <https://doi.org/10.2307/j.ctt2005zq7.12>. (Siehe S. 64)

- Sintayehu, H. & Lehal, G. S. (2020). Named entity recognition: a semi-supervised learning approach. *International Journal of Information Technology*. <https://doi.org/10.1007/s41870-020-00470-4> (siehe S. 35)
- Song, M., Kim, W. C., Lee, D., Heo, G. E. & Kang, K. Y. (2015). PKDE4J: Entity and relation extraction for public knowledge discovery. *Journal of biomedical informatics*, 57, 320–332. <https://doi.org/10.1016/j.jbi.2015.08.008> (siehe S. 14)
- Sprugnoli, R. (2018). Arretium or Arezzo? A Neural Approach to the Identification of Place Names in Historical Texts. In E. Cabrio, A. Mazzei & F. Tamburini (Hrsg.), *Proceedings of the Fifth Italian Conference on Computational Linguistics CLiC-it 2018* (S. 360–365). Accademia University Press. <https://doi.org/10.4000/books.aaccademia.3627>. (Siehe S. 65)
- Stepanyan, L. (2020). Automated Custom Named Entity Recognition and Disambiguation. *International Journal of Signal Processing*, 05 (siehe S. 14, 19–22, 62).
- Sugomori, Y., Souza, A. M. F., Kaluza, B. & Soares, F. M. (2017). *Deep learning: Practical neural networks with Java : build and run intelligent applications by leveraging key Java machine learning libraries : a course in three modules*. Packt Publishing. <http://proquest.tech.safaribooksonline.de/9781788470315>. (Siehe S. 47)
- Suissa, O., Elmalech, A. & Zhitomirsky-Geffet, M. (2020). Optimizing the neural network training for OCR error correction of historical Hebrew texts. *iConference 2020 Proceedings*. <http://hdl.handle.net/2142/106546> (siehe S. 3, 56)
- Thomas, C. (2018). Neues Textformat im DTA: XML (DTABf) mit linguistischer Annotation (TEI class att.linguistic). <http://www.deutschestextarchiv.de/news/68>. (Siehe S. 73)
- Tjong Kim Sang, Erik F. (2002). Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. *COLING-02*:

- The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.
<https://www.aclweb.org/anthology/W02-2024> (siehe S. 68)
- Tjong Kim Sang, Erik F. & de Meulder, F. (2003). Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, 142–147. <https://www.aclweb.org/anthology/W03-0419> (siehe S. 49)
- van Hooland, S., de Wilde, M., Verborgh, R., Steiner, T. & van de Walle, R. (2015). Exploring entity recognition and disambiguation for cultural heritage collections. *Digital Scholarship in the Humanities*, 30(2), 262–279. <https://doi.org/10.1093/llc/fqt067> (siehe S. 13)
- Wachtel, I., Zidon, R., Garti, S. & Shelach-Lavi, G. (2018). Predictive modeling for archaeological site locations: Comparing logistic regression and maximal entropy in north Israel and north-east China. *Journal of Archaeological Science*, 92, 28–36. <https://doi.org/10.1016/j.jas.2018.02.001> (siehe S. 42)
- Wallach, H. M. (2004). Conditional random fields: An introduction. *Technical Reports (CIS)*, 22 (siehe S. 46–48).
- Wang, C., Sun, X., Yu, H. & Zhang, W. (2019). Entity Disambiguation Leveraging Multi-Perspective Attention: IEEE Access, 7, 113963–113974. *IEEE Access*, 7, 113963–113974. <https://doi.org/10.1109/ACCESS.2019.2933644> (siehe S. 29)
- Wettlaufer, J., Johnson, C., Scholz, M., Fichtner, M. & Thotempudi, S. G. (2015). Semantic Blumenbach: Exploration of Text–Object Relationships with Semantic Web Technology in the History of Science. *Digital Scholarship in the Humanities*, fqv047. <https://doi.org/10.1093/llc/fqv047> (siehe S. 36)
- Wevers, M. & Smits, T. (2019). The visual digital turn: Using neural networks to study historical images. *Digital Scholarship in the Humanities*. <https://doi.org/10.1093/llc/fqy085> (siehe S. 50)

- Xu, Z., Wilber, M., Fang, C., Hertzmann, A. & Jin, H. (n.d.). Learning from Multi-domain Artistic Images for Arbitrary Style Transfer. <https://arxiv.org/pdf/1805.09987>. (Siehe S. 50)
- Yadav, V. & Bethard, S. (2018). A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. *Proceedings of the 27th International Conference on Computational Linguistics*, 2145–2158. <https://www.aclweb.org/anthology/C18-1182> (siehe S. 8, 22)
- Yang, Y., Pintus, R., Gobbetti, E. & Rushmeier, H. (2017). Automatic Single Page-Based Algorithms for Medieval Manuscript Analysis. *Journal on Computing and Cultural Heritage*, 10(2), 1–22. <https://doi.org/10.1145/2996469> (siehe S. 50)
- Zhang, M., Geng, G. & Chen, J. (2020). Semi-Supervised Bidirectional Long Short-Term Memory and Conditional Random Fields Model for Named-Entity Recognition Using Embeddings from Language Models Representations. *Entropy*, 22(2), 252. <https://doi.org/10.3390/e22020252> (siehe S. 35, 36)
- Zhang, S. & Elhadad, N. (2013). Unsupervised biomedical named entity recognition: experiments with clinical and biological texts. *Journal of biomedical informatics*, 46(6), 1088–1098. <https://doi.org/10.1016/j.jbi.2013.08.004> (siehe S. 22)
- Zhu, G. & Iglesias, C. A. (2018). Exploiting semantic similarity for named entity disambiguation in knowledge graphs. *Expert Systems with Applications*, 101, 8–24. <https://doi.org/10.1016/j.eswa.2018.02.011> (siehe S. 28, 29)