FAU Erlangen-Nürnberg

Department Germanistik & Komparatistik

Lehrstuhl für Korpus- und Computerlinguistik

Grundlagen der Computerlinguistik III

Wintersemester 2022/2023

Prof. Dr. Stephanie Evert

Named Entity Recognition in juristischen Urteilen mit traditionellen maschinellen Lernverfahren

Projektbericht für „Grundlagen der Computerlinguistik III“

Xinyao Lu

Matrikelnummer: 23008498

xinyao.lu@fau.de

Computerlinguistik/Germanistik

3. Fachsemester

Abgabe: 28. März 2023

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Zusammenfassung

Als Seminarprojekt für die Lehrveranstaltung Grundlagen der Computerlinguistik III an der FAU Erlangen-Nürnberg wird das „Sub-task B: Legal Named Entities Extraction“ vom „SemEval-2023, Task 6: LegalEval: Understanding Legal Texts“ ausgewählt.[1]

Die Datensätze dieser Aufgabe sind englischsprachige juristische Urteile in Indien. Im Projekt werden nur traditionelle maschinelle Lernverfahren verwendet. Trotzdem hat das Projekt auch ein befriedigendes Ergebnis (75% strenger f1-score) mit mindestens 47% Recall bei allen Hauptkategorien (Support >= 50) im Developing Datensatz bekommen.

Der gesamte Code dieses Projekts in verschiedenen Modulen mit der Pipeline und ihrer Visualisierung kann im GitHub Repository „ LegalNER_GdCL_III_Projekt “ gefunden werden.[2]

2 LegalNER: Projektbeschreibung

2.1 Aufgaben der allgemeinen Named Entity Recognition

Named Entity Recognition ist ein grundlegender Bereich des Natural Language Processings, der viele fortschrittliche semantische Analysen des Texts, beispielsweise Informationsextraktion, ermöglicht. Die sogenannten Named Entities sind „Objekte in der realen Welt, z. B. eine Person, ein Ort, eine Organisation, ein Produkt, etc., die mit einem Eigenname referenziert werden können.“[3]

Die Aufgabe eines Named Entity Recognition Programms (kurz: NER) ist, solche Einheiten aus unterschiedlichen Textarten zu erkennen und in Kategorien zu ordnen. Da die Verteilung von den Named Entities in unterschiedlichen Textarten sehr unterschiedlich sein kann und unterschiedliche Klassifikationen in spezifischen Anwendungen angefordert werden, sind die allgemeinen NER Modelle nicht immer geeignet für alle Aufgaben. Dadurch entsteht die Notwendigkeit, spezifische Programme für bestimmte Aufgaben zu entwickeln.

2.2 Spezielle Aufgaben der LegalNER

NER in den juristischen Unterlagen (LegalNER) ist eine dieser Aufgaben, die ein spezifisches Programm auffordern. „Named Entities in juristischen Texten sind etwas anders und feiner als normale verwendete Named Entities ... “[4] Viele Kategorien in der LegalNER wie „JUDGE“ (Richter), „PETITIONER“ (Kläger), „RESPONDENT“ (Angeklagter) werden in meisten allgemeinen NER Programmen nicht voneinander unterschieden oder sind vernachlässigt wie „STATUTE“ (Bestimmung), „PRECEDENT“ (Präzedenzfall). Außerdem werden komplexerer Satzbau und eine große Menge von Fachbegriffen in den juristischen Unterlagen häufig verwendet.

Juristische Urteile bestehen normalerweise aus zwei Teilen, „preamble“ (Präambel) und „judgement“ (Urteil). „Die Präambel eines Urteils enthält formatierte Metadaten wie Namen der Parteien, des Richters, der Rechtsanwälte, das Datum, das Gericht usw. Der folgende Text wird ‚Urteil‘

[1] Link zum Task: https://sites.google.com/view/legaleval/home#h.fbpoqsn0hjeh

[2] Link zum Repository: https://github.com/cometbridge1998/LegalNER_GdCL_III_Projekt

[3] https://en.wikipedia.org/wiki/Named_entity: „In information extraction, a named entity is a real-world object, such as a person, location, organization, product, etc., that can be denoted with a proper name.“

[4] [Named Entity Recognition in Indian court judgments]https://aclanthology.org/2022.nllp-1.15) (Kalamkar et al., NLLP 2022): „Named Entities in legal texts are slightly different and more fine-grained than commonly used named entities like Person, Organization, Location etc.“

genannt. “[5] Im Datensatz vom shared-task werden Präambeln und Urteile getrennt gespeichert. Die Verteilung der Named Entities in der Präambel ist meistens viel dichter als im darauf folgenden Urteil. Deswegen sind die Datensätze für Urteile durch Auswahl der „dicht besetzten“ Sätze „intensiviert“. In diesem Projekt werden nur die Datensätzen für Urteile verwendet.

2.3 Erkennen und Einordnen: doppelte Aufgaben der NER

Die Schwierigkeit einer NER Aufgabe liegt darin, dass sie nicht nur die richtige Klassifikation, sondern auch die genaue Erkennung der Entities (nicht länger, nicht kürzer) benötigt. Allerdings ist es für ein traditionelles Lernverfahren notwendig, Texte in Token zu splitten (Tokenisierung) und demnächst in eine Matrix, die als Eingabe für das maschinelle Lernen dient, umzuwandeln. Eine Entity entspricht normalerweise mehrere Tokens. Für einige besonders lange Arten von Entities wie „PRECEDENT“ und „STATUTE“, sind Entities mit mehr als 10 Tokens nicht selten.

Wegen der unterschiedlichen Längen der Entities entsteht die „BIO“ Tagging Norm für NER Aufgaben. Denjenigen Tokens, die sich am Anfang von einer Entity befinden, wird ein „B-“ („Beginning“) Label gegeben. „I-“ Labels („Insider“) markieren alle Tokens nach dem ersten Token in einer Entity. „O“ („Outsider“) Labels entsprechen allen Tokens, die sich in keiner Entity befinden. Z. B.:

„... of Hongkong Bank ...“

[„o“, „B-ORG“, „I-ORG“]

Im Datensatz gibt es keine Überschneidung von den Labels. Jedes Token oder Zeichen befindet sich in maximal einer Entity.

3 Vorbereitung zum maschinellen Lernen

3.1 Tokenisierung

Die vom Shared Task angebotenen Datensätze sind in JSON Objekten gespeichert. Die annotierten Labels sind mit „String Slices“ angegeben. Z. B.:

„... Hongkong Bank ...“

{ „value“: { „start“: 90, „end“: 103 }, „text“: „Hongkong Bank“, „labels“: [„ORG“] }

Der erste Schritt dieses Projekts ist, alle Texte mit ihren Annotationen in ein Dataframe, das fortschrittliche Verarbeitungen vereinfacht, umzuwandeln. Jede Zeile im Dataframe entspricht einem Token in den Texten. Jede Spalte speichert eine Eigenschaft der Tokens wie „SentenceNr“, „Label“ (nach der „BIO“ Norm) etc.

Ein kleines Problem bei der Umwandlung des Textes in das Dataframe ist, dass die Norm der Tokenisierung nicht vernachlässigt werden darf, besonders wenn die Satzzeichen eine Rolle zur semantischen Interpretation des Satzes spielen. Wenn z. B. ein Punkt am Ende eines Aussagesatzes steht, soll er als getrenntes Token behandelt werden. Im Gegensatz dazu, handelt es sich bei dem Punkt um Kennzeichen für eine Abkürzung, soll er mit den vorangehenden Buchstaben ein ganzes Token bilden.

Das Projekt hat für die Tokenisierung den „TreebankWordTokenizer“ vom Paket „nltk“ gewählt.[6] Die Qualität der Tokenisierung wird durch den Vergleich zwischen dem Goldstandard

[5] [Named Entity Recognition in Indian court judgments](https://aclanthology.org/2022.nllp-1.15) (Kalamkar et al., NLLP 2022): „The preamble of a judgment contains formatted metadata like names of parties, judges, lawyers, date, court etc. The text following the preamble till the end of the judgment is called ”judgment”.“

[6] Link zum Dokument vom Tokenizer: https://www.nltk.org/api/nltk.tokenize.TreebankWordTokenizer.html

nach Annotation und den „detokenisierten“ Texten, die Wiederzusammenstellung der Texte in den Entities nach der Tokenisierung, überprüft. Es wird also verglichen, ob die Texte der Entities (inkl. Satzzeichen) bei der Tokenisierung geändert werden. Nach der Überprüfung gibt es insgesamt nur 7 Entities im gesamten Trainings- und Developingsdatensatz, die wegen der Tokenisierung leicht geändert werden. Vier Änderungen behandeln die Löschung eines Bindestrichs. Durch diese geringe Menge von Fehlern sollte kein spürbarer Effekt bei der endgültigen Evaluation entstehen. Spezifische Reports über alle Tokenisierungsfehler können in „tokenizing_report_train(dev).txt“ aus dem Projektsrepository gefunden werden.

3.2 Part-of-Speech Tagging

„Unter Part-of-speech-Tagging (POS-Tagging) versteht man die Zuordnung von Wörtern und Satzzeichen eines Textes zu Wortarten (englisch part of speech).“[7] Weil die Entities häufig bestimmten Satzteilen entsprechen, werden POS-Tags in das Dataframe hinzugefügt.

Zwei POS Taggers werden vom Projekt benutzt. Der erste Tagger ist der Standardtagger von nltk (ein voraus trainierter PerceptronTagger). Dieser Tagger gibt einen POS-Tag von jedem Token zurück. Dadurch wird null-Wert bei seltenen Tokens im Dataframe vermieden. Der zweite Tagger ist ein mit „Penn treebank“ konfigurierter TreeTagger. Ein Vorteil vom TreeTagger liegt daran, dass er neben POS-Tags zusätzlich die „Lemmas“ (Stammformen) von Tokens liefert. Der Zusatz der Lemmas in das Dataframe sollte nach Vorstellung des Entwicklers die Störung von verschiedenen Flexionsformen beseitigen und dadurch die Stabilität des Modells fördern. Im Gegensatz zum Standardtagger von nltk liefert TreeTagger keinen Tag und Lemma für fremde Tokens. Für solche Tokens, die keinen Tag oder Lemma vom TreeTagger bekommen, sind deren leeren Zellen mit „0“ gefüllt. Spezifische Einzelheiten über Installierung und Konfiguration des TreeTaggers sind im File „README.md“ vom Projektrepository angegeben.

3.3 Feature Matrix

Feature Matrix soll zusätzliche Informationen neben den schon vorhandenen Token, POS-Tag und Lemma, besonders die „Kontext Informationen“ dem maschinellen Lernmodell liefern. Mit „Kontext“ wird gemeint, die umgebenden Tokens und ihre Eigenschaften. Die Auswahl der „Features“ muss normalerweise empirisch bestimmt werden, d. h. Beste Feature Matrix muss durch Ausprobieren unterschiedlicher Kombinationen festgestellt werden.

Um die Textdataframe in eine für maschinelles Lernverfahren geeignete Matrix umzuwandeln, werden drei „CountVectorizers“ verwendet, die jeweils mit Tokens, POS-Tags und Lemmas im Trainingsdatensatz initialisiert sind.

Die Dataframe wird mit „SentenceNR“ (Nummer der Sätze) gruppiert. Als Kontextinformationen eines Tokens werden nur Tokens im selben Satz berücksichtigt. Als Kontextinformationen von den an der Satzgrenze liegenden Tokens werden „Paddings“, statt Tokens in den vorangehenden/folgenden Sätzen, gegeben. Z. B.

Für das vorletzte Token „copy“ im Satz: „ ... on that photo copy . “,

L1 Token is „photo“. L2: „that“, R1: „.“ und R2: „ · “ (Paddingszeichen).

Nach systematischem Vergleich hat die Feature Matrix mit den folgenden Spalten bei einem „LinearSVC“ Modell aus dem Paket scikit-learn bestes Ergebnis erreicht:

Token, POS-Tag und Lemma vom Token selbst und von deren L1, L2, R1, R2 Nachbartokens.

[7] https://de.wikipedia.org/wiki/Part-of-speech-Tagging

Affixe und weitere Eigenschaften von den Tokens können das Ergebnis nicht verbessern. Sie führen nur zur höheren Überanpassung an den Trainingsdatensatz. Der Versuch, mit einer dynamischen Feature Matrix das Modell mit (vorhersagten) Labels von vorangehenden Tokens zu bieten, hat das Ergebnis nur wesentlich verschlechtert.

3.4 Model Selection

Um das möglich beste Ergebnis zu erreichen, werden zwei Modelle von traditionellem Lernverfahren mit „Parameter Finetuning“ (Ausprobieren unterschiedlicher Parameterkombinationen) verwendet.

Das LinearSVC Modell aus scikit-learn hat mit folgende Parametern einen strenge f1-score von 67% bekommen. (Evaluationskriterien werden in „4.1 Zwei Methoden zur Evaluation“ erklärt.)

{„C“: 0.35, „class_weight“: None, „loss“: „hinge“, „penalty“: „l2“}

Das „sklearn_crfsuite“ Modell ist eine Erweiterung des scikit-learn Pakets. Mit der Stärke, die Abfolge der Labels zu lernen, ist das Modell besonders für NER Aufgaben geeignet. Das Modell bietet vergleichbare Funktionen wie scikit-learn, die die Anpassung zu anderen Programmteilen erleichtern. Mit der „word2features“ Funktion des sklearn_crfsuite Modells wird eine Standard Feature Matrix erzeugt. Mit dieser Standard Matrix und den Hyperparametern

{„c1“: 0.173, „c2“: 0.203}

hat das Modell ein wesentlich besseres Ergebnis als das LinearSVC Modell erreicht: strenger f1-score: 75%.

4 Evaluation und Fehleranalyse

4.1 Zwei Methoden zur Evaluation

Da jede Zeile in der Eingabematrix einem einzeln Token (statt einem Entity) entspricht, ist die vorhergesagte Labels (y_predict) auch eine Liste von Labels für einzeln Tokens. Die einfache Evaluation auf der Token Ebene (Wie viele Tokens bekommen die richtige Labels?) darf nicht als das endgültige Ergebnis eines NER Modells zählen. Wie in „2.3 Erkennen und Einordnen: doppelte Aufgaben der NER“ hingewiesen wird, muss eine ernsthafte Evaluation auf der Entity Ebene auch die beiden Aufgaben eines NER Modells bewerten. Nach diesem speziellen Anspruch werden auch zwei Methoden zur Evaluation im Projekt definiert.

Klassifizierungsreport zeigt die Genauigkeit bei der Klassifikation der Entities. D. h. Für diejenige Tokens, die mit richtigen Längen (span) erkannt werden, wie viele davon werden mit richtigen Labels eingeordnet. Diese Methode reflektiert lediglich die Genauigkeit bei der Einordnung der Entities, beispielsweise, handelt „Hongkong Bank“ sich um eine „ORG“ (organization) oder eine „GPE“ (geopolitical entity)?

Recognitionsreport zeigt die Genauigkeit sowohl bei der Erkennung als auch bei der Einordnung der Entities von einem NER Modell. Um als eine richtige Vorhersage in diesem Report zu zählen, muss die Entity nicht nur mit vollständiger Länge erkannt als auch in richtigem Kategorie eingeordnet werden. Der f1-score vom Recognitionsreport wird auch vom Organisator vom shared-task als „strict f1-score“ (strenger f1-score) genannt.

Wenn man die zwei Reports vom gleichen Modell vergleicht, hat der Klassifizierungsreport immer einen höheren f1-score („harmonic mean“ von precision und recall) aber niedrigeren support (Support zeigt die Anzahl der beteiligten Entities in der Evaluation). Das Phänomen ist leicht erklärbar. Denn der Klassifizierungsreport bewertet nur die Genauigkeit der einzelnen Aufgabe

	precision	recall	f1-score	support
CASE_NUMBER	0.99	0.96	0.97	70
COURT	0.99	1.00	1.00	130
DATE	1.00	1.00	1.00	208
GPE	0.97	0.97	0.97	128
JUDGE	0.50	1.00	0.67	7
ORG	0.81	0.83	0.82	58
OTHER_PERSON	0.88	0.81	0.84	217
PETITIONER	0.30	0.38	0.33	8
PRECEDENT	0.79	0.85	0.82	68
PROVISION	0.99	1.00	1.00	197
RESPONDENT	0.00	0.00	0.00	2
STATUTE	0.99	0.99	0.99	180
WITNESS	0.69	0.69	0.69	45
accuracy			0.93	1318
macro avg	0.76	0.81	0.78	1318
weighted avg	0.93	0.93	0.93	1318

Abbildung 1: Klassifizierungsreport vom LinearSVC Modell

	precision	recall	f1-score	support
CASE_NUMBER	0.58	0.55	0.57	121
COURT	0.82	0.73	0.77	178
DATE	0.90	0.94	0.92	222
GPE	0.64	0.68	0.66	182
JUDGE	0.32	0.88	0.47	8
ORG	0.41	0.30	0.35	159
OTHER_PERSON	0.75	0.64	0.69	276
PETITIONER	0.21	0.33	0.26	9
PRECEDENT	0.37	0.33	0.35	177
PROVISION	0.76	0.76	0.76	258
RESPONDENT	0.00	0.00	0.00	5
STATUTE	0.81	0.80	0.80	222
WITNESS	0.48	0.53	0.50	58
micro avg	0.68	0.65	0.67	1875
macro avg	0.54	0.58	0.55	1875
weighted avg	0.68	0.65	0.67	1875

Abbildung 2: Recognitionsreport vom LinearSVC Modell

„Einordnung“, während der Recognitionsreport die doppelten Aufgabe bewertet. Je höher der support im Klassifizierungsreport eines Modells ist, desto mehr Entities werden vollständig erkannt.

Die beiden Modelle erreichen über 93% gewichteten f1-score in den Klassifizierungsreports. Allerdings sind die f1-scores in den Recognitionsreports mit Abstand niedriger. Dadurch kann man feststellen, dass die meisten Fehlers aus der unvollständiger Erkennung der Längen von Entities entstehen. In einfachen Worten, solange ein Entity vollständig gefunden würde, wäre die Einordnung viel leichter.

4.2 Zwei Methoden zur Evaluation

4.3 Confusion Matrix

Confusion Matrix ist eine im scikit-learn eingebaute Funktion, um die falschen Entscheidungen eines Modells herauszufinden. Da eine Confusion Matrix von allen 13 Kategorien (13*13) schwer in einer Abbildung zeigen lässt, werden die Labels in drei Hauptklassen klassifiziert:

	precision	recall	f1-score	support
CASE_NUMBER	0.97	0.94	0.96	82
COURT	1.00	0.99	1.00	148
DATE	0.99	1.00	0.99	201
GPE	0.95	0.93	0.94	105
JUDGE	1.00	1.00	1.00	8
ORG	0.96	0.97	0.97	76
OTHER_PERSON	0.86	0.89	0.88	218
PETITIONER	0.38	0.43	0.40	7
PRECEDENT	0.99	0.97	0.98	109
PROVISION	1.00	1.00	1.00	225
RESPONDENT	0.00	0.00	0.00	3
STATUTE	1.00	1.00	1.00	183
WITNESS	0.61	0.54	0.57	50
accuracy			0.95	1415
macro avg	0.82	0.82	0.82	1415
weighted avg	0.95	0.95	0.95	1415

Abbildung 3: Klassifizierungsreport vom crfsuite Modell

	precision	recall	f1-score	support
CASE_NUMBER	0.79	0.64	0.71	121
COURT	0.89	0.83	0.85	178
DATE	0.95	0.91	0.93	222
GPE	0.68	0.54	0.60	182
JUDGE	0.80	1.00	0.89	8
ORG	0.66	0.47	0.55	159
OTHER_PERSON	0.78	0.71	0.74	276
PETITIONER	0.30	0.33	0.32	9
PRECEDENT	0.68	0.60	0.64	177
PROVISION	0.92	0.87	0.89	258
RESPONDENT	0.00	0.00	0.00	5
STATUTE	0.87	0.82	0.85	222
WITNESS	0.59	0.47	0.52	58
micro avg	0.81	0.72	0.76	1875
macro avg	0.68	0.63	0.65	1875
weighted avg	0.80	0.72	0.75	1875

Abbildung 4: Recognitionsreport vom crfsuite Modell

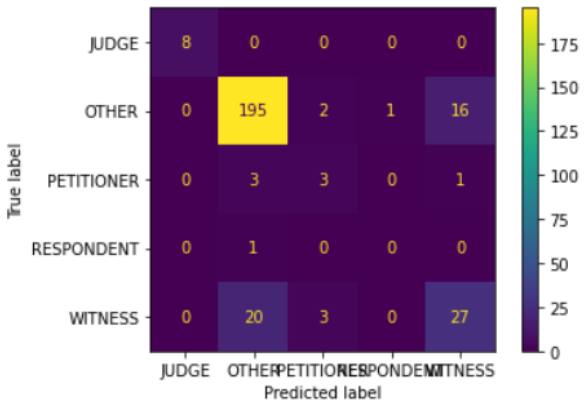

Abbildung 5: Confusion Matrix von der Klasse natural person

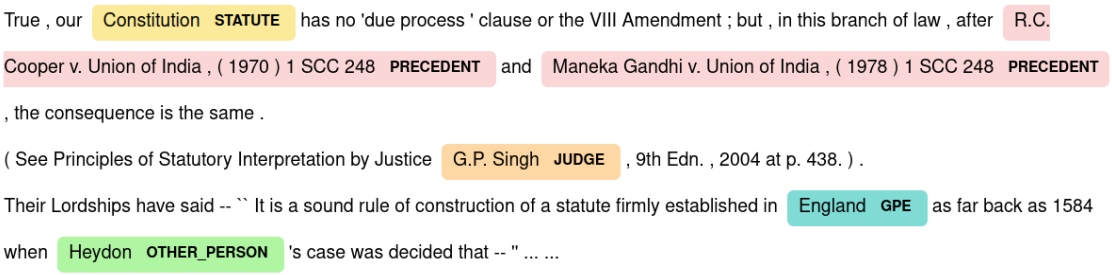

Abbildung 6: Visualisierungsbeispiel

„natural_person“: [„JUDGE“, „OTHER_PERSON“, „PETITIONER“, „RESPONDENT“, „WITNESS“]

„formats“ = [„CASE_NUMBER“, „PRECEDENT“, „PROVISION“, „STATUTE“, „DATE“]

„juridical_person“ = [„COURT“, „GPE“, „ORG“]

Die Confusion Matrix von jeder Klasse oben kann allein gezeigt werden. Abbildung 5 zeigt die Confusion Matrix von der Klasse „natural_person“ mit dem crfsuite Modell.

4.4 Fehleranalyse

4.4.1 versteckter Fehler: Personen in den Präzedenzfällen

4.4.2 natural_person: eine unausgewogene Klasse

5 Schlussbemerkung

Literatur

[1]

Versicherung der selbstständigen Anfertigung

Der Unterzeichnete versichert, dass er die vorliegende schriftliche Hausarbeit selbstständig verfasst und keine anderen als die von ihm angegebenen Hilfsmittel benutzt hat. Die Stellen der Arbeit, die anderen Werken dem Wortlaut oder dem Sinne nach entnommen sind, wurden in jedem Fall unter Angabe der Quellen (einschließlich des World Wide Web und anderer elektronischer Text- und Datensammlungen) kenntlich gemacht. Dies gilt auch für beigegebene Zeichnungen, bildliche Darstellungen, Skizzen und dergleichen.

Erlangen, 28. März 2023
Unterschrift des Verfassers der Seminararbeit:
Xinyao Lu