SAKI SS19 Homework 2

Author: Jannis Wolf

Program code: https://github.com/JannisWolf/Resume_NER

Summary

Ein elektronischer Bewerbungsprozess, welcher heute in den meisten großen Firmen den Standard markiert, eröffnet die Möglichkeit diese Bewerbungen mit Methoden des Machine Learnings zu verarbeiten. Ein Szenario beschreibt das Vorsortieren von Bewerbungen nach gewollten Voraussetzungen die mit Hilfe von Named Entity Recognition (NER) erkannt werden können. Im Folgenden wird die Pipeline zum Trainieren eines NER Modelle beschrieben und anschließend die Ergebnisse evaluiert.

Der benutzte Datensatz umfasst 701 annotierte Bewerbungen. Als Entitäten wurden Abschluss (Degree), vorherige Jobbeschreibung (Designation) und Fähigkeiten (Skills) als aussagekräftige Charakteristiken eines potentiellen Bewerbers gewählt. Nach Filterung des Datensatz wurde mit der Teilmenge von 444 Bewerbungen, die jede der drei Entitäten mindestens einmal enthielt weitergearbeitet. Das Natural Language Processing Framework Flair benötigt für das Training eines NER Modells die Eingabedaten im BILOU-Format, in welches der Datensatz durch eine Helperfunktion von Spacy konvertiert wurde.

Flair Modelle beschränken den Kontext von Sprache auf einzelne Sätze (Sentences). Daher musste noch eine sinnvolle Einteilung in einzelne Sentences gefunden werden. Hier wurden wie im natürlichen Sprachgebrauch die Sentences durch einen Punkt getrennt. Dies stellt sicher, dass auch Bullet Points, welche beispielsweise häufig in Bewerbungen Skills aufzählen, nicht in mehrere Sentences spaltet. In dem vorliegenden Datensatz verhinderte dieses Vorgehen auch die meisten I-Tokens (vor allem der Entität Skills) nicht in verschieden Sentences zu spalten.

Das Framework von Flair stellt eine Reihe an vortrainierten Embeddings zur Verfügung die kombiniert werden können. Klassische WordEmbeddings können mit kontextualen Embeddings gestackt trainiert werden. Die Kombination, welche die beste Resultate erzielte, bestand aus WordEmbeddings des Glove Algorithmus und forward und backward FlairEmbeddings, die auf einem Korpus von einer Milliarde Wörter bidirektional trainiert wurden.

Für das Training des Modells wurde der Datensatz schließlich in 75% / 25% Training / Test Set aufgeteilt. Außerdem werden 7 % des Trainingssets als Validationset verwendet, dessen Score optimiert wird. Durch die Möglichkeit die variable Learning Rate des von Flair trainierten Neuronalen Netz zu überwachen stoppt das Training automatisch bei Erreichen einer zu kleinen Learning Rate. Die Ergebnisse werden im Folgenden Evaluationsteil beschrieben.

Evaluation

Für die Evaluierung der Ergebnisse wurde das F1-Maß berechnet wie auf verschiedenen Wettbewerben von NER Modellen (wie zum Beispiel der Computational Natural Language Learning CoNLL Challenge). Dafür wird die Precision berechnet, welche angibt wie viele gelabelte Entitäten tatsächlich diese Entitäten sind und der Recall, der berechnet wie viele Entitäten das Modell im Korpus findet und tatsächlich richtig bestimmt.

Das F1-Maß kombiniert diese als harmonischen Mittelwert von Precision und Recall.

```
Precision = True Positives / (True Positives + False Positives)
Recall = True Positives / (True Positives + False Negatives)
F1 Score = 2*(Recall * Precision) / (Recall + Precision)
```

Die nachfolgende Tabelle zeigt den F1-Score für jede Entität. Für die Gesamtperformance wurde der Microaverage berechnet, welcher 0.40 beträgt. Eine detailreichere Aufschlüsselung findet sich im Screenshot auf der nächsten Seite.

	Skills	Degree	Designation	Micro Average
F1-Score	0.30	0.67	0.66	0.40

Zur Intepretation dieses niedrigen Wertes muss beachtet werden, dass der Microaverage die Anzahl der zu erkennenden Entitäten mit in die Berechnung einbezieht. Eine von dem NER Modell schlecht zu erkennde Entität kann den Wert gegebenfalls deutlich senken. Deutlich bessere Ergebnisse der Performance konnten mit Entitäten wie zum Beispiel Location und College Name erreicht werden. Jedoch zeigen die gewählten Entitäten sehr gut die Möglichkeiten und Beschränkungen von NER in diesem Szenario auf.

- Skills: Der Grund für den schlechten Gesamtscore ist in der Erkennung der Entität Skills zu finden. Diese erreicht einen F1-Score von 0.30 bei gleichzeitig den am meisten zu bewertenden Tokens. Es ist schnell ersichtlich, dass es sich hier um eine sehr diverse Eigenschaft handelt. Von technischen Fertigkeiten bis Soft Skills waren sehr unterschiedliche Eigenschaften im Korpus für die selbe Entität.

Weiterhin fiel während des Preprocessing des Datensatzes mit Spacy schon auf, dass bei manchen Sets an Bewerbungen die Entität Skill nur unzureichend gelabelt wurde. Während in einigen nur Skills im freien Text gelabelt waren, waren in anderen nur Skills die in Bullet Points gelistet waren gelabelt, wodurch kein sinnvolles Training möglich ist.

Trotzdem wäre mit einem viel größeren Datensatz deutlich bessere Ergebnisse möglich. Fertigkeiten eines Bewerbers aus einem frei geschriebenen Text herauszufinden wäre für eine Recruiting Abteilung dann sehr wertvoll. Mit dem begrenzten Datensatz der hier verwendet wurde, ist dies aber nur sehr begrenzt möglich.

- Degree: Im Gegensatz dazu kann die Entität für den Degree passabel gut erkannt werden, mit einem Score von 0.67. Das Vokabular für Degree ist sehr begrenzt und es stellt sich die Frage, ob mit Hilfe einfacherer Suchalgorithmen und Algorithmen wie Stemming, bekannten Abschlüssen und deren Abkürzungen genauso oder noch erfolgreicher erkannt werden könnten. Im

- Designation: Die Erkennung der Entität Designation erreichte einen ähnlichen F1-Score von 0.66. Dieses Ergebnis kann durchaus positiver eingeordnet werden als Degree, trotz einem ähnlichen Score. Der Trend zu immer neuen, länger und komplizierteren Berufsbezeichnung kann schwer ohne Machine Learning geparst werden. Ein NER Modell, welches hier durch Erlernen des Kontexts gute Ergebnisse auf unbekannten neuen Daten liefert, könnte sehr nützlich sein, wenn man Jobbezeichnungen Bedeutung zumisst.

Abschließend fällt das Urteil, recht positiv aus. Es ist beeindruckend, wie gute Ergebnisse mit einem sehr beschränkten Datensatz und einem mächtigen NER Framework wie Flair erzielt werden konnten. Durch Formulare, in die Bewerber ihre Daten sortiert eintragen müssen, könnten in kurzer Zeit extrem viele annotierte Daten gesammelt werden. Damit könnte sich Named Entity Recognition für das Vorsortieren von Bewerbern gut eigenen.

Screenshot

```
MICRO_AVG: acc 0.2522 - f1-score 0.4028
MACRO_AVG: acc 0.3582 - f1-score 0.5043399999999999
-          tp: 408 - fp: 767 - fn: 1635 - tn: 408 - precision: 0.3472 - recall: 0.1997 - accuracy: 0.1452 - f1-score: 0.2536
Degree     tp: 75 - fp: 33 - fn: 42 - tn: 75 - precision: 0.6944 - recall: 0.6410 - accuracy: 0.5000 - f1-score: 0.6666
Designation tp: 246 - fp: 122 - fn: 134 - tn: 246 - precision: 0.6685 - recall: 0.6474 - accuracy: 0.4900 - f1-score: 0.6578
L-Degree   tp: 67 - fp: 29 - fn: 40 - tn: 67 - precision: 0.6979 - recall: 0.6262 - accuracy: 0.4926 - f1-score: 0.6601
L-Designation tp: 258 - fp: 95 - fn: 109 - tn: 258 - precision: 0.7309 - recall: 0.7030 - accuracy: 0.5584 - f1-score: 0.7167
L-Skills   tp: 63 - fp: 51 - fn: 151 - tn: 63 - precision: 0.5526 - recall: 0.2944 - accuracy: 0.2377 - f1-score: 0.3841
Skills     tp: 62 - fp: 100 - fn: 185 - tn: 62 - precision: 0.3827 - recall: 0.2510 - accuracy: 0.1787 - f1-score: 0.3032
U-Degree   tp: 30 - fp: 9 - fn: 13 - tn: 30 - precision: 0.7692 - recall: 0.6977 - accuracy: 0.5769 - f1-score: 0.7317
U-Designation tp: 11 - fp: 6 - fn: 38 - tn: 11 - precision: 0.6471 - recall: 0.2245 - accuracy: 0.2000 - f1-score: 0.3334
U-Skills   tp: 59 - fp: 39 - fn: 194 - tn: 59 - precision: 0.6020 - recall: 0.2332 - accuracy: 0.2021 - f1-score: 0.3362
```

```
#designation='n Erlangen.'
designation='Senior Sales Manager in Erlangen. \nBachelor in medical engineering at the university of Erlangen. \nI have upgradable programming skills'

degree=''
#degree='Bachelor in computer science at the university of chennai'

skill=''
from flair.data import Sentence
sentence: Sentence = Sentence(designation)

trainer.model.predict(sentence)

print("Analysing %s" % sentence)
print("\nThe following NER tags are found: \n")
print(sentence.to_tagged_string())
```

```
Analysing Sentence: "Senior Sales Manager in Erlangen. Bachelor in medical engineering at the university of Erlangen. I have upgradable programming skills" - 19 Tokens

The following NER tags are found:

Senior <B-Designation> Sales <I-Designation> Manager <L-Designation> in Erlangen. Bachelor <B-Degree> in <I-Degree> medical <I-Degree> engineering <L-Degree> at the university of Erlangen. I have upgradable prog
```