Named Entity Classification

Madita Huvar, Phillip Richter-Pechanski, Sanaz Safdel

27. Februar 2017

Inhaltsverzeichnis

1 Einführung

Named Entity Recognition ist seit den 1990er Jahren ein aktives Forschungsfeld. Es stellt die Grundlage für weitere Forschungsfelder dar. Zum Beispiel im Bereich Semantic Annotation, Question Answering, Opinion Mining und viele mehr.

1.1 Was sind Named Entities?

Named Entities sind Phrasen, die Namen von Personen, Organisationen, Währungen, usw. enthalten. In unserem Projekt werden sie wie folgt dargestellt: BILD

2 Unser Projekt

2.1 Ziel

Typischerweise werden Named Entity Recognition und Named Entity Classification zusammen betrachtet und nur wenige Untersuchungen beschäftigen sich nur mit Named Entity Classification. In unserem Projekt konzentrieren wir uns auf Named Entity Classification und stellen vor allem die Frage, welchen Einfluss Feature Selection auf die Klassifikationsergebnisse eines Named Entity Klassifizierers haben. Wir verwenden hierzu einfache syntaktische und lexikalische Features, die in fast allen Forschungsarbeiten in ähnlicher Form genutzt werden.

2.2 Organisation

Zur Organisation unseres Projekts benutzen wir GitHub. Zusätzlich haben wir eine WhatsApp Gruppe, in der wir uns besprechen und unsere Treffen planen können. Diese finden ca. ein bis zwei Mal pro Woche statt. Unsere Ergebnisse halten wir auf einer speziell angelegten Wikiseite fest.

3 Tools

Für unser Projekt verwenden wir folgende Tools:

- Python 3.4+
- Scikit Learn (als Klassifizierer)
- liac-arff
- matplotlib
- Weka (zur Korpusanalyse)

- GitHub
- ICL-Wiki

4 Korpus

Wir verwenden für unser Projekt das OntoNotes Korpus 2012. In diesem sind englische Nachrichtentexte des 'The Wall Street Journal' enthalten. Es existieren bereits ein Development-/Trainings- und Testset. TABELLE ANZAHL

5 Korpusreader

Für die Extraktion der Named Entities haben wir einen Korpusreader erstellt. Dieser Reader extrahiert alle Named Entities, inklusive POS-Tags der einzelnen Token, Phrasenart, Kontextwörter (ne-1, ne+1) und ordnet ihnen Klassen zu. BEISPIEL

6 Korpusklassenbalancierung

Die Anzahl der Named Entites im Trainingsset ist für die einzelnen Klassen sehr unterschiedlich. Die zehn Klassen mit der geringsten Anzahl Named Entities, werden aus dem balancierten Korpus entfernt. Die Klassen NORP und GPE werden zusammengefasst, da sie semantisch ähnlich sind. Die Klassen MONEY, PERCENT und CARDINAL werden ebenfalls zusammengefasst, da sie alle numerische Klassen sind. Daraus ergeben sich folgende neue Korpusklassen: XXXXXXX Die Verteilung der Named Entities auf die Klassen ist nun wesentlich ausgeglichener, wie in Tabelle XXXXX zu sehen.

7 Baselineklassifizierer

Für unseren Baselineklassifizierer verwenden wir nur das Feature 'Unigram', welches die Vorkommenshäufigkeit, der Unigramme in der Named Entity,welche mindestens fünfmal im Trainigskorpus vorkommen, beschreibt.

8 Erweitertes Featureset

Für unseren Named Entity Klassifizierer, haben wir weitere Features hinzugefügt. Diese sind in Tabelle XXXXXX beschrieben. Insgesamt hatten wir eine Anzahl von 1716 Features.

9 Klassifizierertyp

Zur KLassifizierung der Named Entities wir eine Support Vector Maschine mit linearem Kernel verwendet. SVM XXXXXXFOLIEN Unsere Featurevektoren haben sehr viele Features, daher verwenden wir den linearen Kernel. Mapping in höheren Featurespace eines nicht-linearen Kernels bringt kaum Klassifiziereungsverbesserungen. Alernativ haben wir Decisiontree getestet, dieser hatte allerdings mit allen Featurekombinationen tendenziell schlechtere Evaluationsergebnisse. Zudem trainiert der SVM deutlich schneller. ROC KURVE + MATRIX

10 Featureselection

Insgesamt wurden elf Features eingesetzt. Um die Performance der einzelnen Features zu testen, wurde die Potenzmenge des Featuresets gebildet. Schließlich wurde der Klassifizierer auf allen 1013 Teilmengen durchgeführt. Für die Evaluation entscheidend waren alle Teilmengen, die die Featured Unigram und Context enthalten und mindestens drei Features besitzen. Die Accuracy ohne diese Features lag nur bei ¡69. Wohingegen die Accuracy nur mit den Features Unigram und Context bei 87.42 lag. Wie in ABBILDUNG XXXXX zu sehen erreichen wir die höchste Accuracy ab sieben Features. Jedoch schon ab vier Features gibt es kaum noch Verbesserungen der Accuracy. Die Features, die zur Erhöhung der Accuracy beitragen sind: POS, isallcaps, isinwiki, isnp und containsdigit. Die beste Accuracy bei möglichst kleinem Featureset erreichen wir mit Unigram, Context, POS, und isallcaps.

11 Evaluation

tbd

12 Probleme und Lösungsvorschläge

Ein mögliches Problem ist, dass das OntoNotes Korpus bereits automatisch annotiert ist und dadurch bereits vor unserer Verarbeitung Klassifikationsfehler im Testset vorhanden sind. Um dieses Problem zu beheben, müsste man das Korpus Handannotieren, was jedoch sehr aufwändig ist. Die Klassifizierung der Klasse PERSON könnte möglicherweise durch die Generierung weiterer PERSON-Instanzen verbessert werden. Ein weiterer Punkt, den man genauer betrachten könnte, ist das Feature 'Context', dieses bezieht um Moment auch Satzzeichen mit ein. Man könnte testen, ob eine Verbesserung erzielt wird, wenn man das Feature auf alphanumerische Strings beschränkt.

13 Zusammenfassung

In unserem Projekt haben wir herausgefunden, dass mehr Features nicht zwangsläufig bessere Ergebnisse liefern. Außerdem scheint die Dimensionalität der Features Einfluss auf die Klassifikationsergebnisse zu haben. Hochdimensionale Features wie 'Unigram' oder 'Context' tragen maßgeblich zu besseren Ergebnisse bei. Auch das Zusammenfassen von Klassen, die sich ähneln verbessert die Ergebnisse.

14 Anhang