Klyueva Olga
CIS, LMU München
19.07.1917

Michael Strohmayer: Machine-learning basierte automatische OCR-Korrektur

Mit Hilfe von der OCR-Engine wurden Bilder von Texten in maschine-kodierte Texte konvertiert. Dabei können Fehler entstehen. Der Profiler nimmt alle Wörter, die in der OCR-Ausgabe stehen und erzeugt die unterschiedlichen Interpretationen mit Wahrscheinlichkeiten.
Die Aufgabe der Bachelorarbeit besteht darin, eine Software zu entwickeln, die einen maschine-learning Klassifikator trainiert und anhand von ausgegebenen Profiler-Dump entscheidet, welcher Korrekturvorschlag von einem Wort korrekt ist.
Die verwendeten Dokumente sind zwei Kräutertexte: "Paradiesgärtner" und "Curiöse botanicus". Diese Texte haben ground truth Werte: die falschen Token sind händig nachkorrigiert und der korrekte Korrekturvorschlag präsentiert.
Als nächstes wurde die Ausgabe des Profilers gezeigt. Der OCR-Token "Leter" wurde korrigiert und der korrigierte Token "Leder" eingetragen. Es wurde ein Levenstein distance und ein Konfidenzwert berechnet. Der Konfidenzwert für dieses Beispiel ist relativ niedrig, weil das System zwei Patterns verwendet hat und sie miteinander ersetzen musste. Das System hat einen anderen besseren Korrekturvorschlag gefunden.
Um dieses Problem zu beseitigen, wurden drei Basisfeatures genommen, um maschine-learning Systeme zu trainieren: Konfidenzwert, Levenstein distance und Häufigkeit. Um die Häufigkeit zu erhalten, wurden Frequenzlisten erzeugt. Noch ein weiteres Feature war die Längendifferenz zwischen dem OCR-Token und dem Korrekturvorschlag.
Es wurden zwei maschne-learning Bibliotheken verwendet:

1. Libsvm
2. Scikit-learn

Eines der aufgetretenen Probleme ist das Performance Problem bei großen Datenmengen. Es wurden die Internetdaten in einem Python Dictionary geändert. Es wurden eigene Klassen für jeden Dateneinsatz erzeugt.
Als nächstes wurde Evaluation präsentiert. Durch Crossevaluation wurden gute Ergebnisse geliefert. Es wurden zwei Klassifikatoren miteinander verglichen. Der Naiv Bayes Klassifikator war sehr schnell. Der Libsmn Klassifikator hat hochwertige Ergebnisse geliefert.
Es gab zwei mögliche Einstellungsmöglichkeiten bei dem Libsmn Klassifikator: one-class und multi-class. Das multi-class Modell ist, wenn mehrere Erwartungswerte möglich sind. Das multi-class Modell hat 34 Sekunden gebraucht um zu trainieren und vier Sekunden zu predicten. Für das one-class Modell wurden die Dokumente umgewandelt: falscher Korrekturvorschlag war null und richtiger Korrekturvorschlag war eins. Das one-class Modell benötigt 30 Sekunden zu trainieren und 3 Sekunden zu predicten.
Grundsätzlich reicht die binäre Klassifikation, wenn es falsche oder richtige Korrekturvorschläge gibt. Während des Experiments wurde festgestellt, dass mit dem multi-class Modell bessere Ergebnisse erzielt wurden.
Der Naiv Bayes Klassifikator benötigt eine Sekunde zum Trainieren und weniger als eine Sekunde zum Predicten. Dafür waren die Ergebnisse schlechter. F-Score war 50 Prozent.
Es wurden in der Bachelorarbeit folgende Evaluationsmetriken verwendet: Precision, Recall, Accuracy, F1.
Einer der häufigsten Fehlklassifikatoren war beispielsweise, wenn der Profiler einen falschen Konfidenzwert geliefert hat. Der richtige Korrekturvorschlag wurde oft mit dem niedrigen Konfidenzwert bewertet, was bei der Maschine-Learning zu Fehlern geführt hat.
Der andere häufigste Fehlklassifikator war, wenn kein Korrekturvorschlag in dem ground truth Korpus vorhanden war. Der Grund war, dass der OCR ein Wort erkannte, wo kein Wort gestanden hat oder wenn man nicht weiß, was das Wort bedeutet hat, deswegen waren keine Korrekturvorschläge vorhanden.
Als nächstes wurde die Zusammenfassung präsentiert. Insgesamt ist die automatische Nachkorrektur von OCR-Dokumenten sinnvoll und liefert gute Ergebnisse. Es wurde 99,5 Prozent Accuracy erreicht.
Zum Schluss wurden weitere mögliche Schritte erwähnt:

1. Hinzufügung von zusätzlichen Features.

3. Kombination von beiden Klassifikatoren (Naiv Bayes und Libsvm)