Projektarbeit

Zürcher Hochschule für Angewandte Wissenschaften

Weiss mein Computer wie alt ich bin? [Machine Learning]

HS 2016

Hardegger Florin Kodiyan Don

Titel:	**Weiss mein Computer wie alt ich bin? [Machine Learning]**
Organisation:	Zürcher Hochschule für Angewandte Wissenschaften
Institut:	Institut für angewandte Informationstechnologie
Autoren:	Hardegger Florin, hardeflo@students.zhaw.ch Kodiyan Don, kodiydon@students.zhaw.ch
Hauptbetreuung:	Cieliebak Mark, ciel@zhaw.ch Neuhaus Stephan, neut@zhaw.ch
Nebenbetreuung:	Deriu Jan, deri@zhaw.ch
Abgabedatum:	22.12.2016

Erklärung betreffend das selbständige Verfassen einer Projektarbeit an der School of Engineering

Mit der Abgabe dieser Projektarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat.

Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.) Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Projektarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmassnahmen der Hochschulordnung in Kraft.

Datum, Ort:	Unterschriften:
.....	.....
	.....

Zusammenfassung

Das Ziel dieser Projektarbeit ist die Erarbeitung eines grundlegenden Verständnises von machine learning für die Vorbereitung zur Teilnahme am PAN author profiling-Wettbewerb 2017. Die Aufgabenstellung dieses Wettbewerbs ist es, Alter und Geschlecht einer Person, die einen Text verfasst hat, möglichst genau zu bestimmen. Die Daten für diese Arbeit wurden mit einem bestehenden Modell ausgewertet, das wir für diese Aufgabenstellung angepasst haben. Dafür haben wir verschiedene Methoden miteinander verglichen und die jeweils besten implementiert. Das resultierende Modell erzielt bessere Resultate wie jene der Teilnehmenden des Wettbewerbs im Jahr 2016. Da jedoch in einzelnen Alterskategorien eine ungenügende Anzahl Trainingsdaten zu Verfügung steht, kann unser Modell hauptsächlich die Kategorien mit ausreichender Datenbasis vorhersagen. Für die vertiefte und weiterführende Bearbeitung der Fragestellung im Rahmen einer Bachelorarbeit haben wir bereits ein System entwickelt, das mehr Trainingsdaten sammelt und eine bessere Verteilung der Daten in den Kategorien gewährleistet.

Abstract

The goal of this project is to get an understanding of machine learning in order to prepare for the 2017 PAN International Competition on Author Profiling. The task in this competition is to determine the gender and the age of an author of a given document. To fulfill this task, an existing model was used and adjusted to solve our problem. We compared different Machine Learning methods and implemented the best ones into the model. The final model has managed to achieve better results than the participants of the 2016 PAN Competition. Due to sparse training data in the age categories, our model mainly predicted the most common categories. To solve this problem, a system was developed to collect additional data, which can be used to train our model. This project will be the fundament of a bachelor thesis, where the results will be improved through the usage of additional data.

Vorwort

Bei der Suche nach einer Projektarbeit sind wir eher zufällig auf das Thema machine learning gestossen. Nach kurzer Recherche war für uns klar, dass wir uns in diese spannende Aufgabe vertiefen wollen. Datenmengen nehmen zu und die automatische Datenanalyse wird bereits in vielen Bereichen des täglichen Lebens eingesetzt (Siri, Bilderkennung) und wir noch an Bedeutung zunehmen. Wir erhoffen uns die durch diese Projektarbeit erlangten Kenntnisse auch in eigenen Projekten einsetzen zu können.

Ein herzliches Dankeschön gilt unseren Betreuer Mark Cieliebak, Stephan Neuhaus und Jan Deriu, die uns mit wertvollen Hinweisen unterstützt und begleitet haben. Weiter bedanken wir uns bei Jörg Keller für die guten Ratschläge zu der Struktur und dem Inhalt unserer Arbeit. Ein spezieller Dank geht an Tobias Huonder, Simon Müller, Martin Weilemann und Dirk von Grünnigen für die interessanten Diskussionen und den Informationsaustausch. Zu guter Letzt danken wir unseren Freundinnen und Familienmitgliederm für die Unterstützung und Geduld.

Inhaltsverzeichnis

1 Einleitung

In dieser Arbeit geht es um die Vorbereitung für die Teilnahme am *PAN*-Wettbewerb im Jahr 2017. Der PAN-Wettbewerb beinhaltet drei Aufgaben: *Authorship*, *Orginality* und *Trust*. Diese Arbeit konzentriert sich auf die Authorship-Teilaufgabe *Author Profiling*, genauer auf die *Gender Prediction* und *Age Prediction* [1]. In diesen zwei Tasks geht es um die Vorhersage von Alter und Geschlecht einer Autorin oder eines Autoren eines Dokuments. Die Autorinnen und Autoren der Dokumente sind älter als 18 Jahre und sprechen entweder Englisch, Spanisch oder Holländisch.

Der PAN-Wettbewerb in diesen Teilaufgaben findet im Jahr 2017 bereits zum fünften Mal statt. Anhand der Resultate von 2016 betrachten wir den aktuellen Stand der Technik [2].

1.1 Ausgangslage

Der PAN-Wettbewerb testet die entwickelten Systeme für verschiedenen Sprachen und Textquellen. In dieser Arbeit interessiert uns vor allem wie die Systeme in englischen Tweets abschneiden. In diesem Bereich hat das Team *Waser* am besten abgeschnitten. Es hat mit einer Genauigkeit von 20.98 % das Alter und Geschlecht der Autorin oder des Autoren eines Dokuments richtig vorhersagt. Zusätzlich erreichte das Team *Waser* auch in der Vorhersage des Alters einer Autorin oder eines Autoren das beste Resultat mit einer Genauigkeit von 38.79 %. Die beste Vorhersage des Geschlechts einer Autorin oder eines Autoren erreichte das Team *Busger et al.* mit einer Genauigkeit von 55.75 %.

Den Author Profiling-Task im Jahr 2016 gewonnen hat das Team *Busger et al.* mit einer globalen Genauigkeit von 52.58 % in allen Textquellen und Sprachen.

1.2 Zielsetzung

Das Ziel dieser Arbeit ist die Entwicklung einer Software, die Alter und Geschlecht einer Autorin oder eines Autoren eines Tweets richtig vorhersagt. In dieser Arbeit beschränken wir uns auf den englischen Datensatz und experimentieren mit verschiedenen Methoden des machine learning um eine möglichst hohe Genauigkeit zu erreichen. Die Erkenntnisse sollen dazu beitragen wichtige Erfahrungen im Umgang mit machine learning zu gewinnen. Diese Experimente dienen der Vorbereitung für die Teilnahme am PAN-Wettbewerb 2017, den wir gewinnen möchten [1].

2 Theoretische Grundlagen

Dieses Kapitel gibt eine Einführung in machine learning. Anschliessend wird näher auf die in dieser Arbeit verwendeten Methode des machine learning convolutional neural networks eingegangen.

2.1 Machine Learning und Neuronale Netze

Bei machine learning (ML) geht es ganz allgemein darum, in grossen Mengen von Daten Muster zu erkennen. In unserer Anwendung ist das die Kategorisierung von Tweets nach Alter und Geschlecht. Bekommt also ein ML-System eine grosse Menge an Tweets vorgesetzt, soll es in einer *Trainingsphase* selbstständig herausfinden, welche Eigenschaft der Tweets für bestimmte Altersklassen charakteristisch sind. Als Ergebnis der Trainingsphase entsteht ein *Modell*, in das man einen unkategorisierten Tweet hineingeben kann und aus dem man dann eine vorhergesagte Alterskategorie und ein vorhergesagtes Geschlecht erhält.

Beim *supervised learning* erhält ein ML-System in der Trainingsphase Daten, die bereits mit den korrekten Werten für Geschlecht und Alter bezeichnet sind [3]. In unserem Fall würde zu jedem Tweet also angegeben, ob er von einem Mann oder einer Frau stammt und wie alt die Autorin oder der Autor beim Verfassen der Tweets war. Im Gegensatz dazu muss beim *unsupervised learning* der ML-Algorithmus selbsständig eine Kategorisierung vornehmen [4]. Unsupervised learning kann man unter anderem verwenden um supervised learning zu unterstützen.

Neuronale Netze sind ein Versuch, die Leistung von Gehirnen beim Erkennen von Mustern nachzubilden [5]. Komponenten von Neuronalen Netzen sind also "*Neuronen*", die als *Eingaben* Zahlen zwischen 0 und 1 haben, die verschieden gewichtet sein können. Zusätzlich kann ein Neuron noch eine Ausgabesteuerung besitzen, die *bias* genannt wird. Dieser bias verstärkt oder dämpft die Ausgabe des Neurons um einen konstanten Faktor und steuert damit den höchsten möglichen *Ausgabewert*. Aus diesen Eingaben $(x_1, \ldots, x_n)$, den Gewichten $(w_1, \ldots, w_n)$ und dem bias b berechnet sich die Ausgabe $o(\mathbf{x})$ eines Neurons als

$$o(\mathbf{x}) = \left(1 + \exp\left(b + \sum_{k=1}^{n} w_k x_k\right)\right)^{-1}. \tag{1}$$

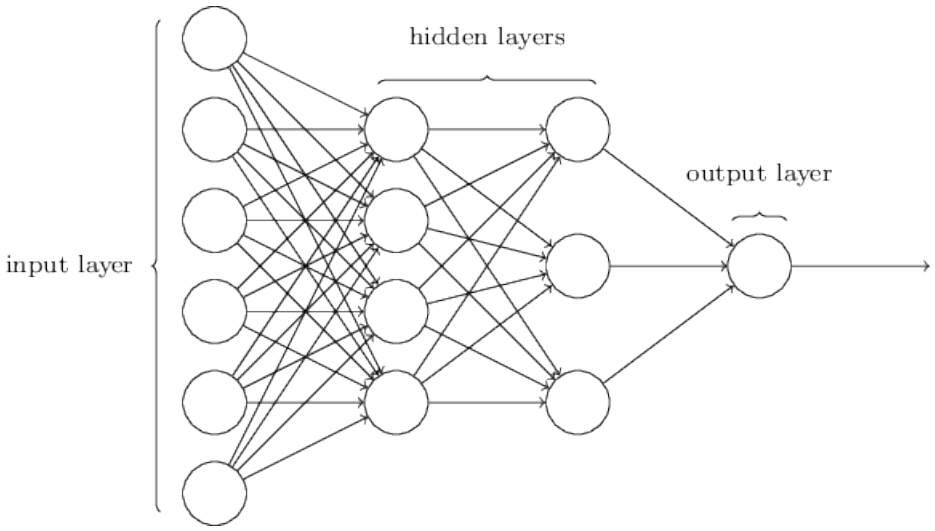

Abbildung 1: Ein Neuronales Netz mit vier Layern. Ganz links der input-Layer, ganz rechts der output-Layer, in der Mitte zwei hidden-Layer. Alle Neuronen von Layer k sind mit allen Neuronen von Layer $k+1$ verknüpft [Quelle: http://neuralnetworksanddeeplearning.com/chap1.html].

Wie in Abbildung 1 ersichtlich, werden Neuronen in Neuronalen Netzen in Ebenen angeordnet, so dass die Ausgaben der Neuronen auf Ebene k die Eingaben der Neuronen auf Ebene $k+1$ sind. Die erste Ebene heisst dann *input-Layer*, die letzte Ebene *output-Layer* und die Ebenen dazwischen heissen *hidden-Layers*. Die Eingaben des input-Layers werden dabei direkt mit den zu klassifizierenden Eingaben verknüpft. Die Ausgaben des output-Layers dienen dann der Klassifikation. Im Fall der Klassifikation von Tweets nach Geschlecht könnte also ein output-Layer zwei outputs haben: Einen für "männlich" und einen für "weiblich". Ist das Signal an einem "weiblich"-output bei einem Tweet besonders stark, ist das ein Indiz, dass das Neuronale Netz diesen Tweet als von einer Frau verfasst klassifiziert.

In der Trainingsphase des machine learning mit Neuronalen Netzen geht es nun darum, die Gewichte der Eingaben der Neuronen so zu modifizieren, dass möglichst genaue Klassifikationsergebnisse erzielt werden. In unserem Beispiel sollen also die Outputs des output-Layers bei von Frauen verfassten Tweets möglichst stark die "weiblich"-outputs ansprechen und bei von Männern verfassten Tweets möglichst stark die "männlich"-outputs.

2.2 Deep Learning und Convolution Neural Networks

Deep learning ist eine der Möglichkeiten wie machine learning umgesetzt werden kann. Von deep learning spricht man, wenn sich in einem Neuronalen Netz zwischen input-Layer und output-Layer eine grosse Anzahl weiterer Layer befindet. Diese grosse Anzahl hidden-Layers erlaubt die Durchführung von komplexen Berechnungen [6].

Es gibt verschiedene Arten wie eine deep learning-Architektur aufgebaut werden kann. Der in dieser Arbeit verwendete Ansatz wird *convolutional deep neural networks* genannt. Diese Architektur verwendet *convolution neural networks* (CNN), die im anschliessenden Kapitel näher ausgeführt werden [7].

Convolution Neural Networks sind *feedforward neural networks*, was bedeutet, dass es zwischen den einzelnen Layer kein Feedback gibt und die Daten einer Ebene k nur an die nachfolgende Ebene $k+1$ übermittelt werden [8]. Das CNN besteht aus *convolution-Layer* und *pooling-Layer*, diese können in der genannten Reihenfolge beliebig häufig in einem CNN vorkommen. Der input-Layer in einem CNN ist anders aufgebaut als die input-Layer des in Kapitel 2.1 besprochenen Neuronalen Netzen. Es liegen nicht mehr alle Neuronen auf einer vertikalen Linie sondern in einem convolutional net und kann in vereinfachter Form als Quadrat beschrieben werden.

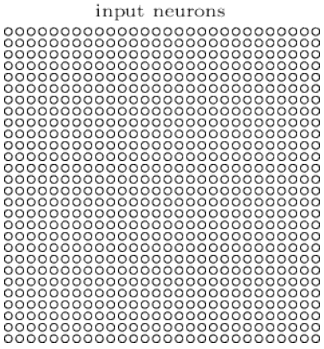

Abbildung 2: Vereinfachte Ansicht eines convolutional nets [Quelle: http://neuralnetworksanddeeplearning.com/chap1.html].

Wie bei Neuronalen Netzen sind auch hier die inputs mit dem nächsten Layer verbunden. Es führen aber nicht mehr alle inputs zu allen Neuronen des nächsten Layers. Stattdessen wird ein Bereich der inputs mit einem einzelnen Neuron des nächsten Layers verbunden. Dieser Bereich, der auf ein hidden-Neuron zeigt, wird *local receptive field* genannt.

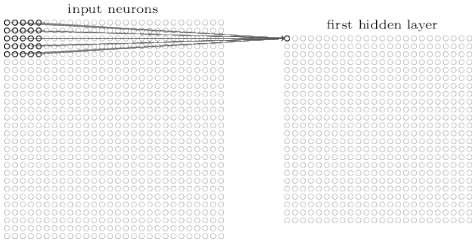

Abbildung 3: Verknüpfung des ersten Bereichs (local receptive field) des input-Layers mit dem ersten Neuron auf dem hidden-Layer [Quelle: http://neuralnetworksanddeeplearning.com/chap1.html].

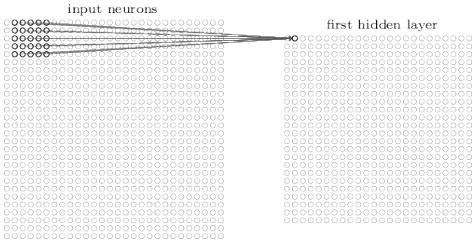

Abbildung 4: Verknüpfung des zweiten Bereichs (local receptive field) des input-Layers mit dem zweiten Neuron auf dem hidden-Layer [Quelle: http://neuralnetworksanddeeplearning.com/chap1.html].

In den Abbildungen 3 und 4 sieht man zwei Beispiele für local receptive fields. In einem convolution-Layer werden für alle Neuronen die gleichen weights und bias benutzt, somit erkennen die Neuronen des ersten hidden-Layers das gleiche feature. Um aus den ursprünglichen inputs andere features zu erkennen, müssen zusätzliche convolution-Layer hinzugefügt werden die andere weights und biases haben.

Der nächste wichtige Layer eines CNN ist der pooling-Layer. Die Funktion eines pooling-Layers ist es, Informationen zu vereinfachen. In einem Bild reicht es zum Beispiel aus zu wissen, dass in einem Bereich eine Kante vorliegt. Die exakte Position dieser Kante zu kennen, ist nicht nötig. Auf diese Weise kann die Datenmenge stark reduziert werden. Der pooling-Layer erlaubt auch in den nachfolgenden Layern komplexe Berechnungen. Meistens kommen pooling-Layer direkt nach einem convolution-Layer zum Einsatz.

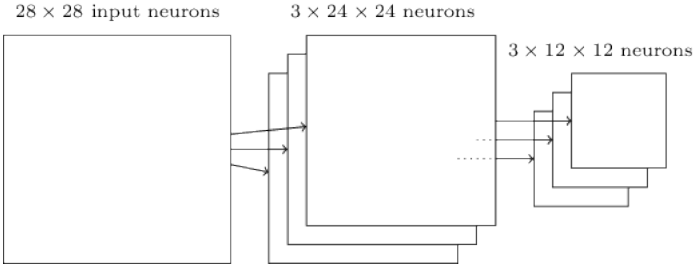

Abbildung 5: CNN mit drei features und anschliessenden pooling-Layer [Quelle: http://neuralnetworksanddeeplearning.com/chap1.html].

In Abbildung 5 wird veranschaulicht wie die Anzahl der benötigten Neuronen mit Hilfe des pooling-Layers reduziert werden kann. Dieser Ausschnitt aus einem CNN zeigt einen 28x28 Neuronen grossen input-Layer, gefolgt von drei convolution-Layers. Jeder dieser convolution-Layer steht für ein feature, das erkannt werden soll. Für jedes dieser features wird ein eigener pooling-Layer benötigt. Eine der Möglichkeiten wie ein pooling-Layer arbeitet, ist das max pooling. Im max pooling wird im betrachteten Bereich nur der höchste Wert gespeichert. Den Abschluss eines CNN machen ein oder mehrere fully connected-Layer. Die Anzahl der Neuronen im letzten Layer hängt von der Anzahl Klassen ab, die das network unterscheiden soll. Dieser Layer wird so bezeichnet, weil auch hier erneut jedes Neuron des pooling-Layers mit den einzelnen output-Neuronen verknüpft wird.

3 Daten

Die in dieser Arbeit erzielten Ergebnisse wurden alle mit den gleichen Daten erzielt. Verwendet wurde der Datensatz der im Zusammenhang der PAN-Aufgaben: Author Profiling zur Verfügung gestellt wurde [9]. Es handelt sich dabei um den Trainingsdatensatz PAN16.

Der englische Teil der Daten besteht aus 433 verschiedenen Autorinnen und Autoren. 216 der Autoren sind weiblich und 217 männlich. Die Aufteilung nach Altersgruppen sieht wie folgt aus:

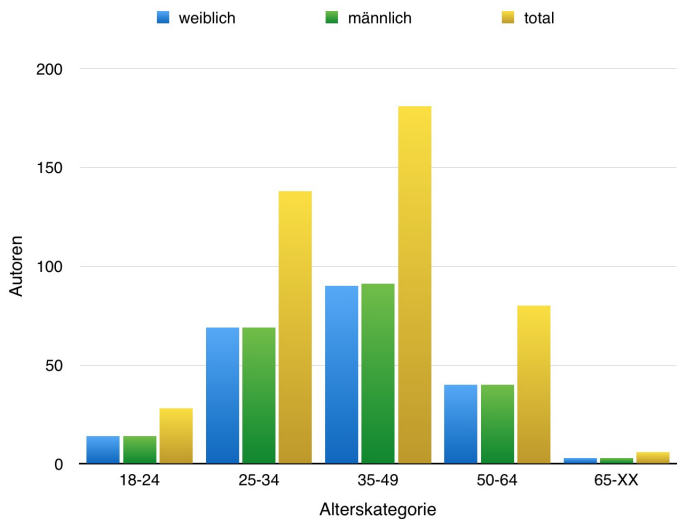

Abbildung 6: Verteilung der Autoren auf die verschiedenen Alterskategorien. [Eigene Darstellung].

Abbildung 6 zeigt eine ungleichmässige Altersverteilung. Die beiden Randbereiche sind stark untervertreten, was sich auf die Vorhersage dieser Gruppen negativ auswirkt. Es macht zum Beispiel wenig Sinn, anhand von nur sechs Autoren in einer Kategorie die verwendeten Wörter zu analysieren. Deshalb haben wir uns entschlossen für die Bachelorarbeit einen grösseren Datensatz zu erstellen und verwenden. Zu diesem Zweck haben wir einen Twitter Crawler entwickelt, der in Kapitel 3.2 genauer beschrieben wird.

Für die ersten Experimente wurden die vorgegebenen, auf Twittersprache optimierten, embeddings verwendet. Embeddings sind eine Vektorrepresentation von Wörter, die Vektoren werden mit Hilfe von *Word2Vec* erstellt. Word2Vec ist ein Neuronales Netz, das ähnliche Wörter in Vektoren mit geringem Abstand zu einander gruppiert [10]. Wie die Wort-embeddings erstellt wurden, wird in der Arbeit [11, Ch.2.2.1] beschrieben. Während ursprünglich 200M Tweets verwendet wurden, sind es bei uns 590M Tweets. Die mit Word2Vec erzeugten Vektoren werden durch die grössere Datenmenge noch genauer, was die Qualität der Wort-embeddings verbessert.

Des weiteren sind wir im Verlauf der Arbeit auf weitere Daten gestossen, die anhand der benutzten Wörter Rückschlüsse auf Alter und Geschlecht eines Autors erlauben.

3.1 Alter und Geschlecht predictive-Lexika

In diesem Abschnitt wurden Alter und Geschlecht predictive-Lexika, welche in [12] erstellt wurden, genauer betrachtet. Die Hauptquelle der Daten, die für die Erstellung der Lexika verwendet wurden, war die Facebook Applikation "MyPersonality". Beim Sammeln der Daten wurde nur nach Autorinnen und Autoren gesucht, die jünger als 65 Jahre alt sind, mindestens 1000 Wörter in deren Statusnachrichten geschrieben und Englisch als Muttersprache angegeben haben. Auf diesem Weg konnten 75394 Facebook Benutzer gefunden werden. Weitere Daten von Facebook, Twitter und Blogs wurden lediglich zu Testzwecken verwendet. Mit diesen Daten konnten zwei Lexika erstellt werden; Ein Lexikon für die Berechnung des Alters mit 10799 Wörtern und ein Lexikon für die Berechnung des Geschlechts mit 7139 Wörtern. Wie diese Daten verwendet werden, wird in Kapitel 5.5.2 beschrieben.

3.2 Twitter Crawler

Das Ziel des Twitter Crawlers ist es, möglichst viele Twitterprofile verschiedener Altersregionen zu finden. Wir wollen in einem ersten Schritt Autorinnen und Autoren aus den beiden Randbereichen finden und in einem zweiten Schritt die Autorenzahl zu erhöhen. Da wir für unsere Arbeit gelabelte Daten benötigen, müssen wir die Autoren gezielt aussuchen.

Um die Suche nach neuen Profilen zu vereinfachen, möchten wir den Prozess so weit wie möglich automatisieren. In einem ersten Schritt suchen wir über die Bing Search API Autoren, die auf Ihrem Profil Altersangaben freigegeben haben.

Abbildung 7: Screenshot eines Twitterprofils mit freigegebenen Altersinformationen [Eigene Darstellung].

In Abbildung 7 sieht man, dass dieser Twitter Benutzer sein Geburtsdatum freigeben hat (rot umrahmt in der Abbildung).

Die Suche auf Bing nach Twitter Profilen mit Altersinformationen funktioniert, weil die Twitter Profile vom Bing-Index erfasst werden. Der Query, den wir bei Bing verwenden lautet:

```
site:twitter.com "born on MMM D, yyyy"
```

Wobei D für den Tag, MMM für die ausgeschriebene englische Monatsbezeichnung und $yyyy$ für das Jahr stehen.

Mit dieser Query erhalten wir über die Bing Search API für jede Anfrage zwischen 0 und 50 Resultate. Mit dieser Methode wurden grösstenteils sinnvolle Resultate gefunden. Jedoch ist eine manuelle Überprüfung der Resultate nötig, da wir sicherstellen müssen, dass das Alter des Autors wirklich freigegeben wurde und das Geschlecht des Autors ermittelbar ist. Sobald auf diesem Weg ein Autor gefunden wurde, können wir für diesen Autor über die Twitter API seine 1000 aktuellsten Tweets herunterladen. Wir haben uns für unseren Crawler für folgendes Design entschieden.

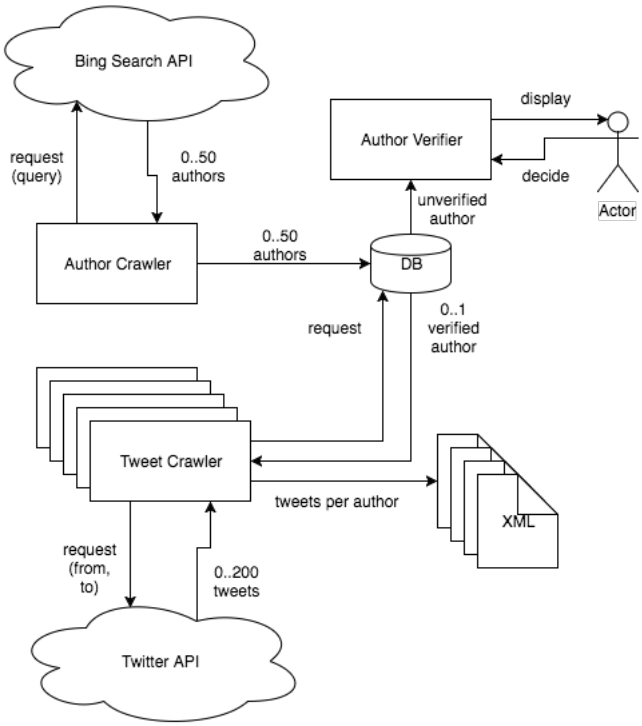

Abbildung 8: Aufbau des Twitter Crawlers [Eigene Darstellung].

Der in Abbildung 8 gezeigte Aufbau ermöglicht, dass wir Author und Twitter Crawler unabhänig voneinander betreiben können. Das wurde so geplant, weil die mittels des Author Crawlers gefunden Profilen zuerst verifiziert werden müssen.

Die Tweet Crawler erhalten aus der Datenbank ein verifiziertes Profil und speichern die 1000 aktuellsten Tweets dieses Autors in einer XML-Datei.

4 Modell

Als Vorlage wurde das Evalitalia Projekt verwendet, welches für die sentiment-Analyse optimiert worden ist [13]. Aufgrund der neuen Datenstruktur musste das Modell entsprechend vorbereitet werden. In diesem Kapitel werden die Architektur und wichtige Funktionen genauer erklärt.

4.1 Architektur

In diesem Abschnitt wird die Architektur des CNN erläutert. Die Architektur des CNN wurde aus dem Evalitalia Projekt übernommen und angepasst. Dabei wurde ein 2-Layer CNN eingesetzt.

Embedding: Der embedding-Layer erhält einen Tweet als Input. Dieser Tweet wird in Form eines Vektors mit der Länge n dargestellt. Einfachheitshalber nennen wir diesen Vektor nun "Tweet-Vektor". Der Tweet-Vektor besteht aus Zahlen und jede Zahl repräsentiert ein Wort. Die Zuordnung zwischen den Zahlen, auch "Vokabularindex" genannt, und Wörtern wird im "Vokabular" definiert. Embeddings bilden die Wörter als ein Vektor mit der Länge d ab. Diese Vektoren werden "Wortvektor" genannt und einen Wortvektor erhält man über den Vokabularindex. Die Aufgabe des embedding-Layers ist es, die Wortvektoren anhand der Vokabularindizes im Input-Tweet zu ermitteln und all diese Wortvektoren zu vereinen. Diese Vereinigung resultiert eine Outputmatrix $T \in \mathbb{R}^{n \times d}$ und wird als Input für das convolution-Layer verwendet.

Convolution-Layer: Das convolution-Layer wird mit m Filter mit einem sliding window der Länge h verwendet. Aus den Wortvektoren werden hier features generiert, welche eine Dimension von $m \times (n - h + 1)$ haben.

Max-Pooling: Nach dem convolution-Layer wird ein max pooling-Layer eingesetzt. Wie im Kapitel 2.2 beschrieben, gibt der max pooling-Layer nur den grössten Wert in einem Bereich dem folgenden Layer weiter. Diesen Bereich nennen wir "Pool". Die Länge des Pools wird mit s definiert und der stride-Wert mit s_t. Der stride-Wert bestimmt, um wie viel der Pool sich verschieben soll. Falls s_t nicht angegeben wird, wird s_t auf 1 gesetzt.

Hidden-Layer: In diesem Layer handelt es sich um einen fully connected hidden-Layer. Mit der weight-Matrix W, dem bias b und der Aktivierungsfunktion (relu) α wird anhand der Funktion $\alpha(Wx + b)$ der Outputvektor berechnet.

Softmax: Der softmax-Layer ist der output-Layer und liefert die Klasse mit der grössten Wahrscheinlichkeit als Lösung.

Layer	Parameter
Embedding	$n = 140$, $d = 52$
Convolution	$m = 200$, $h = 6$
Max Pooling	$s = 4$, $s_t = 2$
Convolution	$n = 140$, $d = 52$
Max Pooling	$s = output\ previous\ layer$
Hidden	$output\ dimension = 200$
Softmax	$output\ dimension = count(classes)$

Tabelle 1: In dieser Tabelle ist der Aufbau der Layer des CNN ersichtlich. Die Layer werden in dieser Reihenfolge und mit den angegebenen Parametern verwendet [Eigene Darstellung].

4.2 Preprocessing

Während des preprocessing werden die Tweets eingelesen und für das CNN vorbereitet. Im CNN wird mit einzelnen Tweets gearbeitet, das Ziel dieser Arbeit ist es jedoch, den Autor mehrerer Tweets zu klassifizieren. Um dieses Ziel zu erreichen, haben wir zusätzlich eine Klasse 'Author' implementiert. Diese Klasse enthält Informationen über Alter und Geschlecht des Autoren. Zusätzlich enthält sie alle zugehörigen Tweets in folgenden Formaten:

- **Plaintext:** Die Tweets werden unverarbeitet abgespeichert.
- **Tokenized:** Die Tweets werden in einzelne Wörter unterteilt und abgespeichert. Es werden alle HTML-Tags entfernt, Websiten und Benutzer-Links jeweils mit '<url>' und '<user>' ersetzt und in Kleinbuchstaben umgewandelt. Anschliessend wird der TweetTokenizer von NLTK verwendet um die einzelnen Tweets in Wörter zu unterteilen.
- **Word Indices:** Die einzelnen Wörter werden in einen eindeutigen Index umgewandelt. Diese Indizes referenzieren auf die Wortvektoren in den Wort-embeddings.

4.3 10-Fold Cross Validation

Um die genaue Leistung eines Modells zu erhalten, setzen wir k-fold cross validation ein. Bei cross validation werden die Trainingsdaten in k gleich grosse Sets aufgeteilt, wobei ein Set eine Sammlung von Tweet-Vektoren ist. Davon wird ein Set als Testset und die restlichen Sets werden als Trainingsset definiert. Mit dem Trainingsset wird jeweils ein Modell trainiert und mit dem Testset wird die Leistung des Modells überprüft. Dies wird für jeden fold gemacht, wobei immer ein anderer Set als Testset benutzt wird. In unserem Fall mussten wir beachten, dass die Tweets der Autoren im gleichen Set bleiben und nicht auf verschiedene Sets verstreut werden. Wir haben uns entschieden, dass wir $k = 10$ setzen. Dies führt zu langen Rechenzeiten, weil 10 Modelle trainiert werden müssen, liefert jedoch genauere Resultate.

4.4 Prediction

Da unser Modell mit einzelnen Tweets trainiert wird, kann es nur einzelne Tweets klassifizieren. Damit ein Autor klassifiziert werden kann, musste eine eigene Methode geschrieben werden. Dies haben wir mit der Klassifizierung der Tweets einer Autorin oder eines Autoren gelöst. Die Wahrscheinlichkeiten, die wir nach der Klassifizierung erhalten haben, wurden addiert. Die Klasse mit dem grössten Wert wurde ausgewählt. Eine andere Möglichkeit wäre gewesen, die vorhergesagte Klasse der Tweets statt der Wahrscheinlichkeiten und die Klasse mit der grössten Vorkommnis auszuwählen. Dadurch gehen jedoch wichtige Informationen über die Wahrscheinlichkeiten der vorhergesagten Klassen verloren.

5 Vorgehen

In Deep Learning gibt es verschiedenen Methoden und Algorithmen. Um diese besser zu verstehen, haben wir in einem ersten Schritt die verschiedenen Methoden und Algorithmen miteinander verglichen und analyisiert.

5.1 Early Stopping und Model Checkpoint: Accuracy vs F1 score

Early stopping und Modell-checkpoint sind Methoden, die verwendet werden um das beste Modell zu finden. Early stopping verhindert, dass sich das Modell zu sehr den Trainingsdaten anpasst. Das ist wichtig, weil ein zu stark den Trainingsdaten angepasstes Modell keine Muster in den Daten mehr erkennen kann und nur noch die Resultate der Trainingsdaten auswendig lernt.

Ein Modell-checkpoint speichert das jeweils beste Modell ab. Das beste Modell wird anhand der Genaugikeit der Ergebnisse der durchgeführten Tests gemessen. Bei der Messung wurde jeweils die Genauigkeit ($accuracy = \frac{correct\ predicted}{total}$) und der F1 score ($F1 = 2 * \frac{precision * recall}{precision + recall}$) verglichen. *Precision* ist das Verhältnis richtiger Vorhersagen zu allen Vorhersagen und *recall* das Verhältnis richtiger Vorhersagen zu der Summe richtiger Vorhersagen und vermissten Elemente dieser Klasse. Die Messung mit F1 score wurde gewählt, weil sich diese nicht nur auf die Vorhersage einer Klasse (z. B. weiblich) beschränkt.

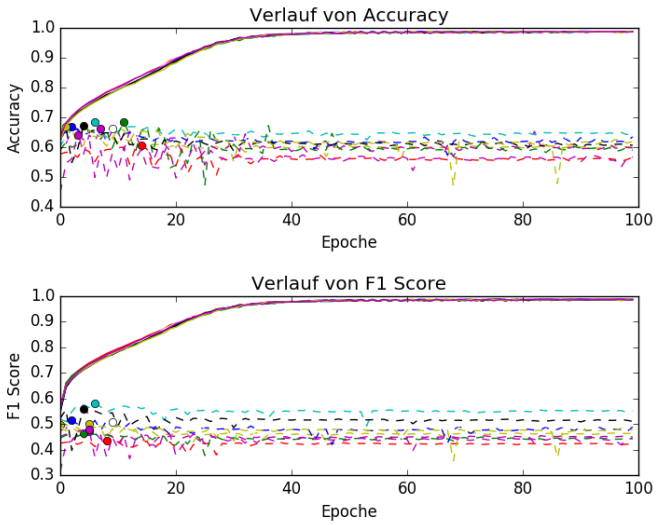

Abbildung 9: Verlauf von accuracy und F1 score beim Trainieren des Modells [Eigene Darstellung].

In Abbildung 9 ist jeweils der Verlauf von accuracy (obere Grafik) und F1 score (untere Grafik) für die zehn folds ersichtlich. Die durchgezogenen Linien stellen die Ergebnisse der Trainingsdaten und die gestrichelten Linien die Ergebnisse der Testdaten dar. Die Punkte kennzeichnen den jeweils besten Wert eines folds. Dieses Experiment wurde jeweils 100 Epochen lang trainiert um den Trainingsverlauf zu verdeutlichen.

Geschlecht Klassifizierung

Ebene	Messmethode	Korrekt geraten	Total geraten	Genauigkeit
Tweet	accuracy	181642	275471	65.94 %
Tweet	F1 score	179491	275471	65.16 %
Autor	accuracy	299	433	69.05 %
Autor	F1 score	307	433	70.9 %

Tabelle 2: Vergleich der Genauigkeit auf Tweet- und Autoren-Ebene [Eigene Darstellung].

Geschlecht und Alter Klassifizierung

Ebene	Messmethode	Korrekt geraten	Total geraten	Genauigkeit
Tweet	accuracy	81218	275471	29.48 %
Tweet	F1 score	78033	275471	28.33 %
Autor	accuracy	146	433	33.72 %
Autor	F1 score	158	433	36.49 %

Tabelle 3: Vergleich der Genauigkeit auf Tweet- und Autoren-Ebene [Eigene Darstellung].

In einem nächsten Schritt haben wir unser Modell die beiden Extremfälle klassifizieren lassen. Extremfälle, weil bei diesen zwei Klassifizierungen der grösste Unterschied an Klassen vorliegt. Bei der Geschlecht Klassifizierung in Tabelle 2 hat es zwei Klassen und bei der Geschlecht und Alter Klassifizierung in Tabelle 3 zehn Klassen. Das Modell mit jeweils der besten accuracy und dem besten F1 score wurde abgespeichert. Bei der Validierung der Modelle zeigt sich, dass das Modell, das nach accuracy ausgewählt wurde, die einzelnen Tweets nur leicht besser klassifizieren kann als das Modell, das anhand des F1 scores ausgewählt wurde. Das Modell mit dem besten F1 score liefert jedoch deutlich bessere Ergebnisse auf Autoren-Ebene. Durch die Berechnung des F1 scores kann das Modell Tweets nicht zufällig einer Klasse zuordnen. Die Wahrscheinlichkeiten sind dadurch weniger extrem ausgeprägt. Dies führt auf Tweet-Ebene manchmal zu einer knapp falschen Vorhersage. Auf Autoren-Ebene werden alle Wahrscheinlichkeiten der Tweets addiert, dadurch erhalten klar eingeschätzte Tweets einen grösseren Einfluss. Auf Autoren-Ebene haben diese falsch eingeschätzten Tweets aber durch die knappen Resultate einen kleineren Einfluss, weil die Wahrscheinlichkeiten aller zugehörtigen Tweets betrachtet werden.

Da sich ausserdem das Modell jeweils bereits nach 50 Epochen stark den Trainingsdaten angepasst hat, wurde das early stopping so definiert, dass nach 50 Epochen ohne Verbesserung abgebrochen wird. Wir haben early stopping bewusst gross gewählt, damit auch bei Änderungen am Neuronalen Netz der beste Wert gefunden werden kann.

5.2 Optimizers

Um ein Neuronales Netz zu optimieren, werden sehr oft Gradientenverfahren eingesetzt um die optimalen Parameter und Gewichte zu finden. In diesem Abschnitt wollen wir verschiedene Verfahren miteinander vergleichen. Für dieses Experiment haben wir uns für folgende vier optimizer entschieden: Adagrad, Adam, Adadelta und RMSprop. Bei diesen vier Algorithmen handelt es sich um adaptive learning rate-Methoden, welche in der Regel die besten Resultate erzielen [14]. Adaptive learning rate bedeutet, dass der optimizer selbst entscheidet, wie gross die Lernschritte sind und diese kontinuierlich anpasst. Wir verwenden die optimizer jeweils mit den von Keras vorgegebenen Standardeinstellungen.

Wie im vorherigen Kapitel ausgeführt, haben wir early stopping auf 50 Epochen gesetzt, gemessen wurde jeweils der F1 score. Im Anschluss folgen die geplotteten F1 score-Verläufe.

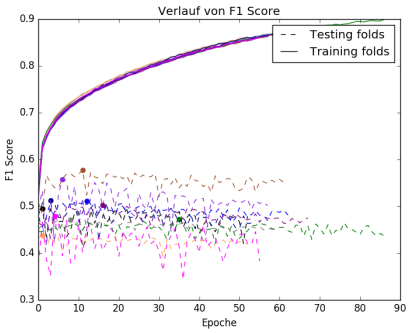

Abbildung 10: Verlauf des F1 scores mit Adadelta [Eigene Darstellung].

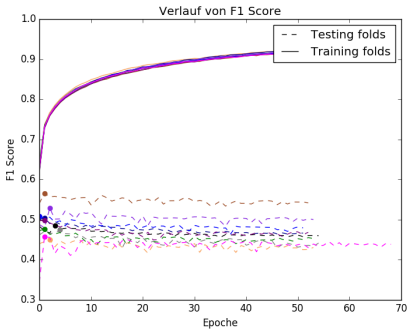

Abbildung 11: Verlauf des F1 scores mit Adagrad [Eigene Darstellung].

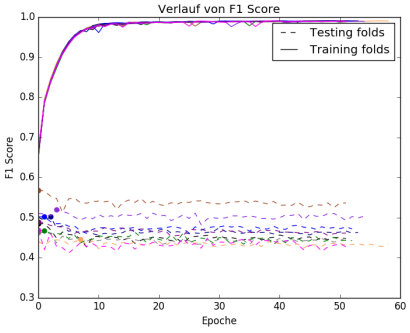

Abbildung 12: Verlauf des F1 scores mit Adam [Eigene Darstellung].

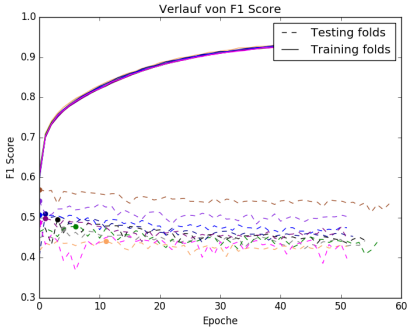

Abbildung 13: Verlauf des F1 scores mit RMSprop [Eigene Darstellung].

In den Abbildungen 10 bis 13 ist jeweils der Verlauf des F1 scores für die 10 folds erkennbar. Die gemessenen Höchstwerte pro fold sind als Punkt dargestellt. Wir stoppen das Experiment jeweils nach rund 50 Epochen, damit man die unterschiedlichen Verläufe des Trainings sehen kann. Für eine Einschätzung der Resultate haben wir die Ergebnisse der Validierung in Tabelle 4 zusammengefasst.

Geschlecht Klassifizierung auf Autoren-Ebene

			Weiblich		Männlich			
Optimizer	Total	Total corr.	pred.	corr.	pred.	corr.	accuracy	F1 score
Adadelta	433	315	172	135	261	180	72.75	72.45
Adam	433	314	143	120	290	194	72.51	71.69
Adagrad	433	306	159	128	274	186	70.67	70.01
RMSprop	433	300	129	106	304	194	69.28	67.96

Tabelle 4: Vergleich der Optimizer auf Autoren-Ebene [Eigene Darstellung]. (pred. bedeutet vorhergesagt und corr. bedeutet richtig vorhergesagt)

Wie man der Tabelle 4 entnehmen kann, haben wir mit den beiden Optimizer Adam und Adadelta die besten Resultate erzielt. Bei der genaueren Betrachtung der Vorhersagen der Modelle wurde deutlich, dass das Geschlecht häufiger männlich vorhergesagt wird. Diese Tendenz hatten alle Modelle, jedoch nicht alle gleich stark ausgeprägt: Adadelta 261 Mal (60 % der Vorhersagen), Adam sogar 290 Mal (67 % der Vorhersagen). Trotz dieser Tendenz erreicht Adam in der Vorhersage der männlichen Autoren nur einen leicht höheren F1 Score (76.53 %), als Adagrad (75.31 %). Da jedoch Adadelta im Schnitt bessere Ergebnisse erreicht hat und nicht ganz so einseitig rät, wurde in allen weiteren Experimenten nur noch Adadelta zu verwenden.

5.3 Epsilon

Beim Epsilon in Adadelta handelt es sich um einen konstanten Faktor, der einen stabilisierenden Einfluss auf die Berechnung eines Updates hat [15]. In diesem Abschnitt haben wir verschiedene Epsilon-Werte ausprobiert und analysiert.

Geschlecht Klassifizierung auf Autoren-Ebene

			Weiblich		Männlich			
Epsilon	Total	Total corr.	pred.	corr.	pred.	corr.	accuracy	F1 score
10^{-3}	433	323	182	144	251	179	74.60	74.43
10^{-4}	433	328	179	145	254	183	75.75	75.56
10^{-5}	433	331	178	146	255	185	76.44	76.25
10^{-6}	433	307	156	123	277	184	70.90	70.31
10^{-7}	433	305	152	120	281	185	70.44	69.76
10^{-8}	433	315	172	135	261	180	72.75	72.45

Tabelle 5: Auswertung der Leistung von verschiedenen Modelle bei der Geschlechtklassifizierung auf Autoren-Ebene. Die Modelle wurden mit Adadelta und den angegebenen Epsilon-Werten trainiert [Eigene Darstellung]. (pred. bedeutet vorhergesagt und corr. bedeutet richtig vorhergesagt)

In Tabelle 5 ist ersichtlich, dass die grossen Epsilon-Werte (10^{-3}, 10^{-4} und 10^{-5}) die besseren Resulate lieferten, als die kleineren Epsilon-Werte. Bei Epsilon 10^{-8} handelt es sich um den default-Wert. Es zeigte sich ausserdem, dass grosse Epsilon-Werte dazu führen, dass es beim Trainieren zu sehr starke Schwankungen kommt. Es kann sein, dass nach einer Epoche der F1 score stark einbricht. Bei sehr kleinen Epsilon-Werten gab es kaum Änderungen nach einer Epoche. Dieser Effekt hängt damit zusammen, dass Epsilon den maximalen Lernschritt limitiert. Unsere Vermutung ist, dass kleine Epsilon-Werte in einem lokalen Maximum/Minimum hängen bleiben. Hier haben wir uns deshalb für Epsilon 10^{-5} entschieden, da dieser nicht nur die besten Resultate lieferte, sondern sich auch als stabiler als die grösseren Epsilon-Werte erwiesen hat.

5.4 Class Weights

Die Herausforderung bei der Klassifizierung von Autorinnen und Autoren nach Alterskategorien, war die ungleiche Anzahl Personen pro Kategorie. In der Kategorie der 65+ jährigen befanden sich nur 6 Autorinnen und Autoren, im Vergleich zur Kategorie der 35-49 jährigen mit 181 Personen. Aus diesem Grund haben wir die class weights anhand der Anzahl von Autorinnen und Autoren pro Kategorie gesetzt. Wir haben folgende Formel benutzt: $\frac{max(allclassfrequencies)}{classfrequency}$. Die class weights werden somit für jeden fold neu berechnet. Beim Trainieren des Modells werden diese class weights als Parameter mitgegeben.

Truth \ Predicted	18-24	25-34	35-49	50-64	65-xx
18-24	816	9967	8183	219	7
25-34	1430	49106	42592	2159	54
35-49	597	31631	77785	2924	92
50-64	165	11274	29856	4060	40
65-xx	0	236	2270	8	0

Tabelle 6: Confusion-Matrix: Modell ohne class weights [Eigene Darstellung].

In Tabelle 6 sind die Auswertungen des Models ohne class weights und auf Tweet-Ebene ersichtlich. In der Diagonalen steht jeweils die Anzahl Tweets, die korrekt erraten wurde. Dieses Modell sagte die älteste Kategorie sehr selten und immer falsch vorraus.

Truth \ Predicted	18-24	25-34	35-49	50-64	65-xx
18-24	4847	7580	5698	811	256
25-34	11375	40733	34542	6933	1758
35-49	9199	30317	60643	10255	2615
50-64	3168	10986	21714	8090	1437
65-xx	52	230	2125	71	36

Tabelle 7: Confusion-Matrix: Modell mit class weights [Eigene Darstellung].

In Tabelle 7 sind die Auswertungen des Modells mit class weights und auf Tweet-Ebene ersichtlich. Mit diesem Modell wurden wie erwartet die Tweets aus Kategorien, die in den Trainingsdaten schlecht vertreten sind (18-24, 50-64, 65-xx), öfter vorausgesagt als mit dem vorherigen Modell. Es resultierte jedoch in einer Verschlechterung um 1 % auf Autoren-Ebene. Wenn das Modell Alter und Geschlecht klassifizieren musste, schnitt dieses Modell noch schlecher ab. Aus diesem Grund haben wir uns entschieden, keine class weights einzusetzen.

5.5 Embeddings

In diesem Abschnitt haben wir mit der Erzeugung der embeddings experimentiert. Wir wollten testen, wie es sich auswirkt, wenn weitere Informationen über die Verwendung des Wortes in den verschiedenen Alterskategorien einbezogen werden. Bis anhin wurde ein Wort mit einem Vektor e der Länge 52 abgebildet, welcher mit Word2Vec generiert wurde. Nun haben wir diesen Vektor e um einen Vektor a ergänzt. Um den Vektor a zu berechnen, wurden zwei verschiedene Ansätze verfolgt, *embeddings mit Trainingsdaten* und *embeddings mit predictive Lexikas*.

5.5.1 Embeddings mit Trainingsdaten

In diesem Ansatz wurden die Trainingsdaten des PAN-Wettbewerbs verwendet um den Vektor a zu berechnen. Der Vektor a enthält Informationen über die Relevanz eines Wortes in allen Kategorien.

Zur Ermittlung der Relevanz der Wörter wurde die Robertson-Sparck-Jones-Formel (RSJ-Formel) verwendet [16]. Die RSJ-Formel wird eigentlich für Information Retrieval verwendet. Wir haben die Formel für unsere Zwecke angepasst.

$$v_k = \log \frac{(R(q,k) + 0.5)/(R(q) - R(q,k) + 0.5)}{(d(k) - R(q,k) + 0.5)/(N - d(k) - R(q) + R(q,k) + 0.5)} \tag{2}$$

N bezeichnet die Anzahl Autoren. $R(q)$ bezeichnet die Anzahl aller relevanten Autoren, d.h. alle Autoren, die sich in der korrekten Kategorie befinden. $d(k)$ bezeichnet die Anzahl aller Autoren, die das Wort k verwenden und $R(q,k)$ die Anzahl aller relevanten Autoren, die das Wort k verwenden.

Die Autoren wurden in zehn Kategorien nach Geschlecht und den fünf Alterskategorien unterteilt. Aufgrund dieser 10 Kategorien hat der Vektor a eine Länge von 10 und nach dem Verketten mit Vektor e, eine Gesamtlänge von 62. Bei genauer Betrachtung der a Vektoren aller Wörter wurde festgestellt, dass bei den 65+ Kategorien Wörter häufig sehr stark gewichtet werden, die dort gar nicht vorkommen. Der Grund dafür ist auch hier, dass die Kategorien sehr unausgeglichen sind und die RSJ-Formel die seltenen Kategorien stärker gewichtet.

In der ersten Implementierung wurden die embeddings aus den gesamten Trainingsdaten generiert. Erst zu einem späteren Zeitpunkt wurden die Trainingsdaten in Trainings- und Testteile unterteilt. Dies hat bei der Evaluierung zu sehr guten Resultaten geführt. Unser Modell konnte die Testdaten um ca. 20% besser klassifizieren. Dieser Fehler machte deutlich, dass embeddings einen sehr starken Einfluss auf die Klassifizierung der Tweets haben. Das Einbinden von geschlechts- und altersrelevanten Informationen in die embeddings führt im Model zu grossen Fortschritten.

Um diesen Fehler zu beheben, wurde für jeden cross validation fold seperate embeddings erzeugt. Doch weder dieser Versuch noch ein anderes Experiment, bei dem die Relvanz auf 0 gesetzt war falls $R(q,k) = 0$, führte zu Verbesserungen.

5.5.2 Embeddings mit Predictive Lexica

In diesem Ansatz wird der Vektor a anhand der Lexika, die in Kapitel 3.1 beschrieben wurde, berechnet. Für die Berechnung des Alters und Geschlechts wird folgende Formel angewendet:

$$y(\mathbf{x}) = \left(\sum_{f \in features} w_f x_f \right) + w_0. \tag{3}$$

x_f steht für die relative Häufigkeit eines Wortes in einem Dokument und w_0 ist eine Konstante Verschiebung des Y-Achsenabschnittes (intercept). Diese Verschiebung sorgt dafür, dass die Resultate auf das Lösungsniveau gehoben werden. Nachstehend ein Beispiel wie mit dem Lexikon das Alter eines Autors berechnet werden kann. Beispielsatz: "My dog is my best friend."

Wort	weight
_intercept	23
my	13
dog	202
is	-20
best	-121
friend	56

Tabelle 8: Abbildungen einiger Wörter und deren weihgts aus dem age Lexikon. Diese Werte werden benötigt, um das Alter des Verfassers des Beispielsatzes "My dog is my best friend." zu berechnen. Zur Vereinfachung wurden die Zahlen auf ganze Zahlen gerundet [Eigene Darstellung].

Die weights sind in der Tabelle 8 ersichtlich. x_f ist für das Wort "my" $\frac{2}{6}$ ansonsten immer $\frac{1}{6}$. Die gesamte Berechnung des Alters sieht wie folgt aus:

$$13\frac{2}{6} + \frac{202}{6} - \frac{20}{6} - \frac{121}{6} + \frac{56}{6} + 23 = 46.8 \tag{4}$$

Anhand dieser Berechnung wird das Alter der Autorin oder des Autors auf 46 Jahre geschätzt. Mit diesem Lexikon erreicht die Berechnung des Alters eine Genauigkeit von ± 7 Jahren auf Facebook Status Nachrichten [12].

Der Vektor a muss aufgrund der relativen Worthäufigkeit x_f für jedes Dokument neu berechnet werden. Bis anhin wurde der Vektor a direkt in die embeddings eingefügt, das ist unter diesen Umständen nicht mehr möglich. Dieses Problem löst ein zusätzlicher Layer, der den Vektor a auf Tweet-Ebene berechnet und in die embeddings einfügt. Die Entwicklung dieses Layers ist im Rahmen dieser Arbeit nicht möglich und wird deshalb in der Bachelorarbeit implementiert.

6 Resultate

In diesem Kapitel werden die Erkenntnisse des Kapitels 5 zusemmengefasst. Anschliessend wird das resultierende Modell mit seinen Leistungen beschrieben.

Im Verlauf unserer Experimente wurde deutlich, dass Modelle mit einem hohen F1 score auf Tweet-Ebene bessere Vorhersagen machen. Das Modell, welches mit Adadelta trainiert wurde, erreichte bei der Validierung den besten F1 score. Adadelta gilt als Optimizer, der nicht konfiguriert werden muss. Durch eine Anpassung des Epsilon-Wertes konnte seine Genauigkeit jedoch verbessert werden. Mit class weights haben wir die Vorhersage von Randklassen erzwungen. Bei der Validierung dieses Modells auf Autoren-Ebene verschlechterte sich jedoch die Genauigkeit und die Randklassen wurden nicht besser klassifiziert. Das grösste Potential sehen wir bei der Erstellung eigener embeddings, wenn Wörter nach der Häufigkeit der Benutzung in den verschiedenen Klassen rangiert werden. Dafür werden jedoch mehr Daten in bestimmten Alterskategorien benötigt.

Es wurden schliesslich drei Modelle entwickelt, die Geschlecht, Alter oder beides ermitteln können. Die Modelle sind auf englische Twitterdaten ausgelegt und bestehen aus einem 2-Layer CNN, welches in Kapitel 4.1 beschrieben wird. Die Modelle wurden mit Adadelta trainiert, wobei Epsilon auf 10^{-5} gesetzt wurde. Es wurden keine class weights eingesetzt und die vorgegebenen embeddings verwendet.

Klasse	Genauigkeit
Geschlecht	76.44 %
Alter	52.42 %
Geschlecht und Alter	36.03 %

Tabelle 9: Diese Tabelle zeigt die Genauigkeit der Modelle [Eigene Darstellung].

Die Modelle wurden mit unseren Trainingsdaten trainiert und deren Leistung mit cross validation überprüft. Die Genauigkeiten der jeweiligen Klassen ist in Tabelle 9 ersichtlich.

7 Ausblick

Für die Interpretation haben wir die Resultate unserer Modelle mit den Resultaten des PAN-Wettbewerbs 2016 verglichen [2]. Dabei wurde nur die Rangliste der englischen Social Media Daten betrachtet. Für den Vergleich wurden die Teams gewählt, die in mindestens einer Kategorie am besten abgeschnitten haben.

Team	Geschlecht und Alter kombiniert	Geschlecht	Alter
Diese Arbeit	**0.3603**	**0.7644**	**0.5242**
Waser	0.2098	0.523	0.3879
Busger et al.	0.1897	0.5575	0.3046

Tabelle 10: In dieser Tabelle werden die besten zwei Teams des PAN-Wettbewerbs 2016 mit dieser Arbeit verglichen. Verglichen wird aufgrund der Genauigkeit der Modelle in englischen Social Media Daten [Eigene Darstellung].

Tabelle 10 zeigt, dass unsere Modelle in allen Klassen eine höhere Genauigkeit erreichen. Dieses gute Ergebnis verdanken wir den für Twitter optimierten Wort-embeddings, dem 2-layer CNN und den Erkenntnisse aus den Versuchen.

Das Ziel dieser Arbeit war die Auseinandersetzung mit machine learning für die Vorbereitung auf den PAN-Wettbewerb 2017. Dieses Ziel konnten wir erreichen, was sich in den Resultaten abbildet. Wir sind zuversichtlich für den Wettbewerb im nächsten Jahr.

Diese Arbeit wird in einer Bachelorarbeit ausgehend von folgenden Ansätzen fortgesetzt:

Embeddings ergänzen: Die zusätzlich gesammelten Twitter-Daten werden in die embeddings einfliessen. Wir wollen die Relevanz der einzelnen Wörter mit der RSJ-Formel ermitteln und mit anderen Formeln vergleichen. Die predictive lexica soll in unser Modell eingebaut und beobachtet werden, ob dies die Leistung des Modells verbessert.

Auf Autoren-Ebene trainieren: Bis anhin wurde immer auf Tweet-Ebene trainiert. Wir wollen ermitteln, ob unser Modell auf Autoren-Ebene trainiert werden kann. Bei der Implementierung sind zwei Ideen enstanden: 1. Die Autoren könnten mittels unsupervised learning geclustert und anschliessend mit supervised learning klassifiziert werden. 2. Der Einsatz eines zweidimensionalen convolution-Layers: Ein Autor würde mit einer zweidimensionalen Matrix, die all seine Tweets enthält, dargestellt. Der Filter des convolution-Layers hätte dabei eine Höhe von 1, damit die verschiedenen Tweets nicht vermischt würden. Für diese Ideen müsste eine neue Architektur erarbeitet werden.

8 Verzeichnisse

Abbildungsverzeichnis

Tabellenverzeichnis

Auflistungen

Literatur

[1] PAN. Pan @ clef 2017. [Online]. Available: http://pan.webis.de/tasks.html

[2] B. V. W. D. M. P. B. S. Francisco Rangel, Paolo Rosso, "Overview of the 4th author profiling task at pan 2016: Cross-genre evaluations," Autoritas Consulting, S.A., Spain, PRHLT Research Center, Universitat Politècnica de València, Spain, CLiPS - Computational Linguistics Group, University of Antwerp, Belgium, Web Technology & Information Systems, Bauhaus-Universität Weimar, Germany, Tech. Rep., 2016.

[3] J. Brownlee. A tour of machine learning algorithms. [Online]. Available: http://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

[4] S. I. Inc. Machine learning: What it is & why it matters. [Online]. Available: http://www.sas.com/en_id/insights/analytics/machine-learning.html

[5] M. A. Nielsen. (2015) Neural networks and deep learning. [Online]. Available: http://neuralnetworksanddeeplearning.com/chap1.html

[6] M. Copeland. What's the difference between artificial intelligence, machine learning, and deep learning? [Online]. Available: https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

[7] J. Brownlee. What is deep learning? [Online]. Available: http://machinelearningmastery.com/what-is-deep-learning/

[8] MathWorks. feedforwardnet. [Online]. Available: https://ch.mathworks.com/help/nnet/ref/feedforwardnet.html

[9] PAN. Author profiling. [Online]. Available: http://pan.webis.de/clef16/pan16-web/author-profiling.html

[10] A. Colyer. The amazing power of word vectors. [Online]. Available: https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/

[11] F. U. A. L. V. D. L. M. J. Jan Deriu, Maurice Gonzenbach, "Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision," Tech. Rep., 2016.

[12] J. C. E. M. L. K. D. S. M. K. L. H. U. H. A. S. Maarten Sap, Gregory Park, "Developing age and gender predictive lexica over social media," Psychometrics Centre, University of Cambridge, Tech. Rep., 2014.

[13] M. C. Jan Deriu, "Sentiment analysis using convolutional neural networks with multi-task training and distant supervision on italian tweets," Zurich University of Applied Sciences Switzerland, Tech. Rep., 2016.

[14] S. Ruder. An overview of gradient descent optimization algorithms. [Online]. Available: http://sebastianruder.com/optimizing-gradient-descent

[15] M. D. Zeiler, "Adadelta: An adaptive learning rate method," New York University, USA, Tech. Rep., 2012.

[16] R. Ferber. Die robertson-sparck-jones-formel. [Online]. Available: http://information-retrieval.de/irb/ir.part_3.chapter_2.section_3.html

9 Anhang

Datenträger: Dies ist eine Beschreibung der Struktur sowie des Inhalts des dieser Arbeit zugehörigen Datenträgers.

- Bericht
  - Finale PDF-Version
  - LaTex-Projekt
- Dokumente
  - Paper (alle in dieser Arbeit zitierten Paper)
- System
  - age_gender (Das in dieser Arbeit entwickelte System)

Zürcher Hochschule für Angewandte Wissenschaften School of Engineering

[ONLINE ADMINISTRATION PRAKTISCHE ARBEITEN]

[DEPT. T ADMIN TOOLS]

zurück Logout

Projektarbeit 2016 - HS: PA16_ciel_2

Allgemeines:

Titel: Weiss mein Computer wie alt ich bin? [Machine Learning]
Anzahl Studierende: 2

Betreuer:

HauptbetreuerIn: Mark Cieliebak, ciel
NebenbetreuerIn: Stephan Neuhaus, neut

Zugeteilte Studenten:

Diese Arbeit ist zugeteilt an:
- Florin Hardegger, hardeflo (IT)
- Don Anson Kodiyan, kodiydon (IT)

Fachgebiet:

DA	Datenanalyse
SOW	Software

Studiengänge:

ET	Elektrotechnik
IT	Informatik
WI	Wirtschaftsingenieurwesen

Zuordnung der Arbeit :

InIT	Institut für angewandte Informationstechnologie

Infrastruktur:

Arbeitsplatz kann frei zugeteilt werden

Interne Partner :

Es wurde kein interner Partner definiert!

Industriepartner:

SpinningBytes AG 8700 Küsnacht

Beschreibung:

Es klingt fast nach Zauberei: man gibt dem Computer ein paar Zeilen Text einer Person, und schon erkennt er das Alter und Geschlecht des Autors (bzw. der Autorin)! Wenn dies zuverlässig funktioniert, kann man z.B. Fake-Accounts auf Facebook erkennen und ausschliessen.

Aber wie macht man das? Wir verwenden einen Ansatz mittels Deep Learning, den wir gemeinsam mit der ETH entwickelt haben: Zunächst erzeugt der Computer aus Milliarden Beispiel-Texten ein generisches Sprachmodell. Dieses wird anschliessend für Alters- bzw. Geschlechtserkennung trainiert und optimiert wird.

Einen analogen Ansatz haben wir bereits für die Sentiment-Analyse (ist ein Text positiv oder negativ) eingesetzt und damit SemEval 2016 gewonnen. Auch für die Alterbestimmung gibt es einen Wettbewerb, PAN, an dem wir dieses Jahr zum ersten Mal teilgenommen haben (Resultate ausstehend). Unser Ziel ist es den nächsten Wettbewerb, PAN 2017, zu gewinnen!

In dieser Arbeit exadaptieren Sie die existierenden Technologien für die Alters/Geschlechtsbestimmung. Dazu verwenden Sie Standard-Libraries für Deep Learning (zB Theano), unsere IT Infrastruktur mit GPU-Maschinen, einen Datenpool von über einer Milliarde Tweets sowie verschiedenen Methoden aus dem Machine Learning.

Falls Sie Interesse an diesem spannenden Thema haben, können wir gern einen Termin abmachen und die konkrete Aufgabenstellung besprechen. Email: ciel@zhaw.ch oder Telefon: 058 934 72 39.

Informations-Link:

Unter folgendem Link finden sie weitere Informationen zum Thema:
http://PAN 2016: http://pan.webis.de/clef16/pan16-web/author-profiling.html

Voraussetzungen:

- Java und/oder Python Kenntnisse
- Bereitschaft sich in Machine Learning, GPU Programmierung und insbesondere Neuronale Netze einzuarbeiten
- Freude an Experimentieren und Ausprobieren

zurück Logout