Text als Daten

Extraktion von Variablen mittels LSTM-Netzwerken

Hendrik Erz und Anastasia Menshikova

Beitrag zur Veranstaltung »Aktuelle Themen der empirischen Sozialforschung« der Sektion Methoden der empirischen Sozialforschung

Einleitung

Textdaten sind für Sozialwissenschaftler*innen wertvolles Datenmaterial. Sie enthalten Informationen über gesamtgesellschaftliche Bedeutungszusammenhänge (Mohr 1998), Intentionen und Handlungen (Franzosi 1989), Stereotype (Nelson 2021), geschlechterbasierte Diskriminierung (Garg et al. 2018) und mehr. Die Extraktion dieser Daten stellt den*die Wissenschaftler*in allerdings vor analytische und methodische Probleme.

Zum einen hat die Verbreitung des Internets und sozialer Medien seit den 1990er-Jahren zu einem starken Anwachsen der für sozialwissenschaftliche Forschung zur Verfügung stehenden Korpora geführt (Lazer und Radford 2017; Trübner und Mühlichen 2019). Dabei sind die Datenmengen so rasant gewachsen, dass diese Korpora nicht mehr durch Menschen gelesen, kategorisiert und analysiert werden können.

Zum anderen konnte die Computerlinguistik wiederum seit etwa den 1990er-Jahren maßgebliche Durchbrüche in der automatisierten Analyse von Text vorweisen (für einen Überblick vgl. Jurafsky und Martin 2020; Eisenstein 2018). Teils wurden Methoden aus den Computerwissenschaften importiert – wie „word embeddings" (Mikolov et al. 2013) – und teils eigens für die Extraktion von Informationen entwickelt – wie „topic models" (Grimmer et al. 2022; Mohr et al. 2020).

Lange Zeit war es damit möglich, Bedeutungen automatisiert aus Text zu extrahieren (vgl. Mohr 1998; Kozlowski et al. 2019; Gerow et al. 2018). Doch war es lange Zeit schwer, *Handlungen* zu extrahieren. Es gab einige Versuche mit den etablierten Methoden (vgl. bspw. Mohr et al. 2013); andere setzten auf qualitative Mittel (vgl. bspw. Franzosi 1989). Derzeit experimentieren erste Autor*innen mit *dependency parsing*, d. h. der automatisierten Extraktion von Grammatik, um Handlungen in Text zu analysieren (Stuhler 2022; Knight 2022).

Liegt das Erkenntnisinteresse allerdings ausschließlich bei den handelnden Subjekten selbst, so reicht *dependency parsing* alleine nicht aus, da Attribute von Akteur*innen nicht explizit im Text enthalten sind und daher auch nicht durch *dependency parser* erfasst werden können. Darunter fällt beispiels-

weise das Geschlecht, welches teils durch das grammatikalische Geschlecht oder sogenannte „Ko-Referenzierung“[1] definiert wird.

In diesem Artikel stellen wir eine neue Methode vor, welche in der Lage ist, Attribute von Subjekten und Objekten zu extrahieren. Wir nutzen neuronale Netzwerke des Typs „Long Short-Term Memory“ (LSTM, Hochreiter und Schmidhuber 1997). Ziel ist es, latente Informationen aus Text in nominal-skalierte Variablen zu überführen, welche im Anschluss statistisch ausgewertet werden können.

Wir testen unsere Methode mit einer Fallstudie und untersuchen, ob weiblich gelesene Personen seltener *handeln* als männlich gelesene Personen (vgl. auch Garg et al. 2018). Dabei definieren wir handelnde Akteure nach Franzosi (1989) als Wörter, welche in der Subjektposition von Sätzen stehen. Wir trainieren in einer Fallstudie zwei LSTM-Netzwerke darauf, das Geschlecht von Subjekten und Objekten zu erkennen und nutzen sie, um mehrere Textkorpora entsprechend zu labeln.

Als Datenmaterial nutzen wir zum einen Wikipedia-Biografien (Lebret et al. 2016), die wir sowohl zu Trainings- als auch Validierungszwecken nutzen. Zum anderen nutzen wir Artikel der New York Times (Sandhaus und Evan 2008) sowie U.S. Parlamentsreden (Gentzkow et al. 2019), um einen Vergleichswert für die Wikipedia-Biografien zu haben.

Im Folgenden geben wir zunächst einen Überblick über computerlinguistische Methoden zur Textanalyse und ihre Anwendung in den Sozialwissenschaften und führen in die Funktionsweise von LSTM-Netzwerken ein. Im Anschluss führen wir eine Machbarkeitsstudie durch und implementieren ein LSTM-Netzwerk, welches in der Lage ist, das Geschlecht von Subjekten und Objekten in Sätzen zu erkennen. Wir schließen mit einer Diskussion der Ergebnisse und geben Empfehlungen zur Nutzung von LSTM-Netzwerken zur Textklassifikation.

Forschungsstand

Text ist nichts Neues für die Soziologie. Allerdings war das Aufbereiten von Text lange Zeit Territorium der qualitativen Sozialforschung, die mittels Coding-Guides Textbausteine kategorisiert und im Anschluss analysiert hat (bspw. im Grounded Theory-Ansatz oder der qualitativen Inhaltsanalyse, vgl. Baur und Blasius 2019, S. 525–543 sowie S. 633–648). Die quantitative Sozialforschung interessiert sich schon lange für Text, konnte aber erst in den vergangenen Dekaden robuste Methoden hierfür vorweisen.

Der erste Versuch, Text quantitativ zu fassen, geht auf Mosteller und Wallace (1963) zurück, welche mithilfe der Häufigkeit von Funktionswörtern die anonymen „Federalist Papers“ der Gründungsväter der Vereinigten Staaten de-anonymisiert haben. In den folgenden vier Dekaden setzte die quantitative Sozialforschung weiterhin auf Ansätze, die mit Worthäufigkeiten oder ähnlichen Metriken arbeiten.

Erst 2003 entwickelte David Blei mit den *topic models* eine neue Methode zur Textklassifikation (Blei et al. 2003). *Topic models* wurden in den folgenden Jahren stetig weiterentwickelt (Roberts et al. 2014). *Topic models* sind Bayes’sche Modelle und klassifizieren Dokumente in Themen (*topics*). Sie arbeiten *unsupervised*, also nicht durch den*die Forschende*n geleitet. Außerdem arbeiten sie unter der „bag of words”-Annahme, was bedeutet, dass die Reihenfolge der Wörter und dementsprechend die grammatikalische Struktur nicht bedeutungsvoll ist.

Zehn Jahre später entstand ein anderer Ansatz, Text computergestützt zu analysieren: *Word embeddings*. Basierend auf Googles **`word2vec`**-Algorithmus (Mikolov et al. 2013) sind *word embeddings* in der Lage, die Semantik von Wörtern anhand derer Position im Satz zu erlernen. Damit geben sie die ggf. zu

[1] Ko-Referenzierung bezeichnet den linguistischen Fall, in welchem das Subjekt eines Satzes durch das Subjekt eines anderen Satzes referenziert wird. Zum Beispiel: den zwei Sätzen „Robert ist Schüler. Er geht in die 12. Klasse“ referenziert das Personalpronomen „er“ das Subjekt des vorhergehenden Satzes, „Robert“.

triviale Annahme eines „bag of words“ auf. Zudem handelt es sich um ein neuronales Netzwerk. Sie werden vornehmlich genutzt, um Nähe- und Distanzbeziehungen zwischen Wortbedeutungen zu analysieren (Kozlowski et al. 2019; Garg et al. 2018).

Vaswani et al. (2017) stellten später das „Transformer“-Modell vor, welches seither genutzt wird, um große Textkorpora abhängig vom Erkenntnisinteresse in Kategorien einzuordnen (vgl. Bonikowski et al. 2022; Do et al. 2022). Zahlreiche weitere statistische Modelle wurden außerdem erprobt (vgl. z.B. Perry und Benoit 2017; Däubler und Benoit 2022; Nelson et al. 2018; Chang und DeDeo 2020).

Das sozialwissenschaftliche Erkenntnisinteresse reicht jedoch über Bedeutungszusammenhänge hinaus. Text enthält dezidiert Intentionen (wie jemand handeln möchte) und Beschreibungen von Handlungen (wie jemand gehandelt hat).

Erst kürzlich erschienen erste Arbeiten, die sich diesem Problem mittels *dependency parsing* nähern (Stuhler 2022; Knight 2022). *Dependency parsing* beschreibt den Prozess der Extraktion von grammatikalischen Beziehungen aus Text, um beispielsweise die gegenseitigen Abhängigkeiten (*dependencies*) von Subjekt, Verb und Objekt zu identifizieren.

Soziolog*innen können bei der Analyse von Text auf drei Sprachebenen zurückgreifen: die *Semantik*, d. h. die Bedeutungen von Wörtern selbst; die *Syntax*, also die Grammatik; und die *Pragmatik*, d. h. die Bedeutung eines ganzen Satzes im Kontext.

Syntax und Pragmatik waren lange Zeit nicht für Computer analysierbar. Dies hat sich erst in den 1990er-Jahren mit der Entwicklung von neuronalen Netzwerken geändert. Der erste Durchbruch fand 1997 mit der Entwicklung von „Long Short-Term Memory“-Netzwerken statt (Hochreiter und Schmidhuber 1997). Mit solchen LSTM-Netzwerken war es möglich, ganze Sätze zu klassifizieren. Die Methode des *dependency parsing* beispielsweise nutzt LSTM-Netzwerke, um die syntaktischen Abhängigkeiten von Sätzen zu extrahieren (vgl. Qi et al. 2020).

LSTM-Netzwerke sind ebenso in der Lage, die Pragmatik von Sätzen zu erfassen. So können LSTM-Netzwerke Sätze in abstrakte Kategorien klassifizieren oder gar in andere Sprachen übersetzen. Google hat für seinen Übersetzungs-Service *Google Translate* auch LSTM-Netzwerke verwendet (Metz 2016).

Allerdings sind LSTM-Netzwerke nie in der Soziologie angekommen. Dies könnte mit dem Aufwand zu tun haben, der lange Zeit nötig war, um ein LSTM-Netzwerk einzurichten. 2017 wurden LSTM-Netzwerke dann durch sogenannte Transformer-Modelle überholt (Vaswani et al. 2017). Transformer-Modelle erfüllen effektiv denselben Zweck wie LSTM-Netzwerke, unterscheiden sich von diesen aber in einigen Aspekten.

Erstens sind Transformer besser auf großen Computersystemen zu skalieren. Während Sätze in LSTM-Netzwerken Wort für Wort eingeführt werden müssen, können Transformer ganze Absätze in einer Operation analysieren. Dieses – rein technische – Problem war die zentrale Motivation für Google, an einem Nachfolger für LSTM-Netzwerke zu arbeiten (Whittaker 2021). Zweitens können Transformer in zwei separaten Schritten trainiert werden: In einem ersten lernen sie generelle Aspekte von Sprache (sogenanntes „pre-training“), bevor sie in einem zweiten Schritt auf ein spezifisches Problem angepasst werden (das sogenannte „finetuning“, vgl. Brown et al. 2020). Diese Methode wird als „transfer learning“ bezeichnet.

Besonders der zweite Aspekt hat Transformer auch für die Soziologie interessant gemacht, da „transfer learning“ mit Transformern vergleichsweise unkompliziert umsetzbar ist. So nutzen einige Artikel die Methode des „Aktiven Lernens“, um große Mengen Text schnell in abstrakte Kategorien zu klassifizieren (Do et al. 2022). Bonikowski et al. (2022) beispielsweise klassifizieren Reden aus dem U.S.-Wahlkampf als „populistisch“ oder „nicht populistisch“.

Diese Möglichkeiten werden aus sozialwissenschaftlicher Perspektive allerdings teuer erkauft: Erstens können Transformer-Modelle nur schwer für ressourcenarme Sprachen angewendet werden, da sie erhebliche Datenmengen zum Vortraining benötigen (Bender et al. 2021). Sprachen mit wenigen

digitalisierten Texten können daher nicht von den Vorteilen des Transformers profitieren (siehe aber Lankford et al. 2021). Zweitens sind Transformer-Modelle wesentlich größer als LSTM-Netzwerke und benötigen entsprechend leistungsstarke Hardware. Auf handelsüblichen Computern benötigt ihr Training unzumutbar lange und vielfach ist der Zugang zu Rechenzentren vor allem für Sozialwissenschaftler*innen erschwert (Whittaker 2021). Drittens verfügen Transformer nicht notwendigerweise über ein besseres Verständnis von Sprache als andere, kleinere Sprachmodelle (Bender et al. 2021).

LSTM-Netzwerke haben aus sozialwissenschaftlicher Perspektive jedoch einen Vorteil, welche Transformer nicht (mehr) besitzen. Ein zentraler Aspekt von LSTM-Netzwerken ist das „feature engineering". Damit wird der Prozess der Variablenselektion beschrieben. Denn anders als Transformer können LSTM-Netzwerke auch nicht-sprachliche Daten für die Klassifikation nutzen. *Feature engineering* kann theoriegeleitet stattfinden und ermöglicht es, beispielsweise Grammatik zu nutzen (Levy und Goldberg 2014; Komninos und Manandhar 2016). Diese Fähigkeit ist von zentraler Bedeutung für unsere Fallstudie (vgl. unten).

Methodik

Ziel der Studie ist, ein LSTM-Netzwerk darauf zu trainieren, das Geschlecht von Subjekt und Objekt eines Satzes zu bestimmen. Wir trainieren dazu zwei LSTM-Netzwerke separat – eines für die Subjekte, eines für die Objekte. Im Folgenden erläutern wir zunächst allgemein die Funktionsweise von LSTM-Netzwerken, bevor wir im Anschluss die konkrete Implementation vorstellen.

Zur Funktion von LSTM-Netzwerken

LSTM-Netzwerke sind neuronale Netzwerke. Sie haben zum Ziel, einen Input (entspricht den unabhängigen Variablen in einer Regression) zu klassifizieren (entspricht der abhängigen Variable), ohne aber vorzugeben, mit welcher Link-Funktion (beispielsweise linear oder logistisch) das geschehen soll (Breiman 2001).

Da neuronale Netzwerke keine funktionale Form vorgeben, müssen sie trainiert werden. Dafür wird ein Datenset benötigt, welches Eingabedaten der gewünschten Klasse (dem „label") zuweist. Diese Daten werden dann durch das Netzwerk klassifiziert. Je größer die Abweichung zwischen der Vorhersage des Netzwerkes und dem korrekten Label, desto stärker werden im Anschluss die Parameter des Netzwerkes angepasst. Dieser Vorgang wird so oft wiederholt, bis die Genauigkeit des Netzwerkes ausreichend hoch ist. Im Anschluss kann das Netzwerk dann neue Daten klassifizieren.

LSTM-Netzwerke gehören zur Familie der Recurrent Neural Networks (RNN). Diese Netzwerke verarbeiten den Input Wort für Wort und updaten für jedes Wort eine interne Datenmatrix – den sogenannten „hidden state". Diese Datenmatrix kann zum Schluss zur Klassifikation des Textes genutzt werden.

Die Besonderheit von LSTM-Netzwerken gegenüber generischen RNNs ist, dass die Datenmatrix in einer „memory cell" verarbeitet wird, die dabei hilft, dass spätere Wörter solche am Anfang des Satzes nicht „überschreiben". Anders gesagt hilft diese „Gedächtniszelle", dass die Netzwerke den Anfang eines Satzes nicht „vergessen", während sie die letzten Worte verarbeiten (Hochreiter und Schmidhuber 1997).

Nachdem ein Satz in den „hidden state" eingelesen wurde, wird dieser von einem letzten Layer, welcher der LSTM-Zelle nachgelagert ist, in die gewünschten Klassen eingeordnet.

Als Eingabe in das Netzwerk können allerdings nicht die Wörter selbst verwendet werden, da Computer mit Zahlen arbeiten. Daher müssen die Wörter in Zahlen umgewandelt werden. Dies geschieht mittels „Einbettungen" (*embeddings*). Dabei wird jedem Wort ein Zahlenvektor zugewiesen, der zunächst

mit Zufallszahlen besetzt wird. Während des Trainingsvorgangs werden diese Zufallszahlen dann angepasst.

Hier ist der **`word2vec`**-Algorithmus, welcher auch die oben bereits eingeführten *word embeddings* produzieren kann, eine große Hilfe. Solche vortrainierten *word embeddings* helfen dem LSTM-Netzwerk, schneller korrekt zu klassifizieren, da nur noch die Parameter der LSTM-Zelle innerhalb des Netzwerkes angepasst werden müssen. Die Adaption des LSTM-Netzwerkes, welche wir im vorliegenden Artikel verwenden, nutzt beispielsweise **`word2vec`**, um grammatikalische Abhängigkeiten in *embeddings* umzuwandeln (Levy und Goldberg 2014). Dieser Vorgang wird wie auch das Vortraining bei Transformer-Modellen als „transfer learning" bezeichnet.

Implementation

Für die Fallstudie verwenden wir ein LSTM-Netzwerk nach dem Vorbild von Komninos und Manandhar (2016), welche zur Verbesserung der Klassifikationsleistung nicht nur vortrainierte *word embeddings* nutzt, sondern auch die grammatikalischen Abhängigkeiten der Wörter (Levy und Goldberg 2014). Die vollständige Architektur des Netzwerkes ist in Abbildung 1 wiedergegeben.

Weiterhin experimentieren wir auch damit, die Koreferenz-Auflösung des Netzwerkes zu verbessern. Dies erreichen wir dadurch, dass wir den „hidden state" des LSTM-Netzwerkes vor der Vorhersage manipulieren und in die Richtung des zuletzt erkannten Geschlechtes „nudgen". Dafür wandeln wir die gewünschte geschlechtliche Verteilung der drei Kategorien durch einen umgekehrten Layer in einen „hidden state" um, der dann als Start-Status genutzt werden kann.

Als Datenset haben wir uns für ein ausgeglichenes Sample aus englischen Wikipedia-Biografien (Lebret et al. 2016) entschieden. Damit stellen wir sicher, dass in unserem Trainings-Datenset möglichst viele Sätze mit Personen als Subjekte und Objekte vorkommen, um die Menge geschlechtsneutraler Subjekte (wie Unternehmen oder Stoffnamen) zu minimieren.

Insgesamt haben wir aus dem Datensatz 1.851 Sätze extrahiert und händisch mit dem Geschlecht von Subjekt und Objekt versehen. Wir haben drei Codes verwendet – „weiblich", „männlich" und „unbekannt". Uns ist bewusst, dass Geschlecht keine binäre Kategorie ist, jedoch wollten wir für diese explorative Studie den Aufwand gering halten. Prinzipiell spricht nichts dagegen, die Kategorien beliebig zu erweitern. Die dritte Kategorie – „unbekannt" – dient für geschlechtsneutrale Nomen wie Stoffnamen, Unternehmen und Institutionen.

Diese rund 2.000 Sätze haben wir in einen Trainings- und einen Validierungs-Datensatz aufgeteilt. Mit dem Trainings-Datensatz haben wir das Modell verbessert, während wir mit dem Validierungs-Datensatz Metriken zur Leistung des Netzwerkes berechnet haben. Zum einen haben wir die Präzision gemessen, das heißt, wie oft das Netzwerk Beispiele aus dem Validierungs-Datenset richtig klassifiziert hat. Zum anderen haben wir den F1-Score berechnet. Der F1-Score bildet ein gewichtetes Mittel aus der Präzision und der Sensitivität. Beide Metriken werden in Prozent gemessen, wobei ein höherer Wert besser ist.

Im Anschluss an das Training haben wir die Netzwerke zwei weitere Datensets klassifizieren lassen: ein Sample von 300 Sätzen aus der New York Times (Sandhaus und Evan 2008) und ein Sample von 300 Sätzen aus U.S.-Kongressreden (Gentzkow et al. 2019). Aus den entsprechenden Vorhersagen haben wir schlussendlich einfache Häufigkeitsstatistiken berechnet, die wir im folgenden Abschnitt vorstellen.

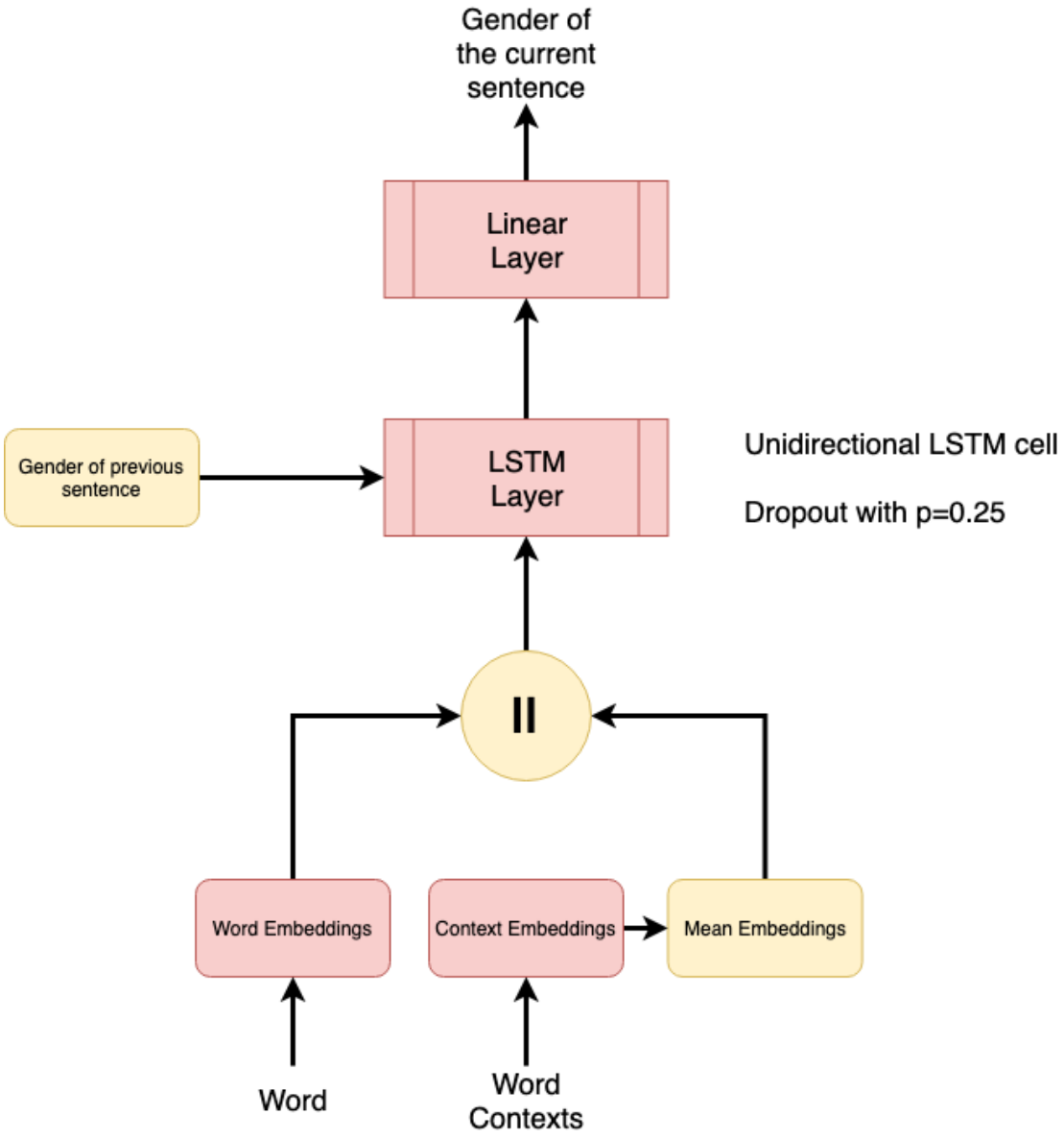

Abbildung 1: Die Architektur des hier verwendeten LSTM-Netzwerkes
Zu Beginn wird ein *hidden state* generiert, der mit dem vorhergesagten Geschlecht des vorhergehenden Satzes in eine Richtung gelenkt wird. Während der Klassifikation eines Satzes wird der Eingabe-Vektor für jedes Wort in das Netzwerk wie folgt generiert: Zunächst werden die Grammatik-Embeddings in einen Durchschnitts-Vektor überführt und an den entsprechenden Vektor des Wortes angehangen. Bei der Verwendung von 1x300-dimensionalen Embedding-Vektoren hat der resultierende Vektor also die Dimensionalität 2x300. Der finale *hidden state* nach Verarbeitung des letzten Wortes wird dann durch einen einfachen linearen Layer geschleift, welcher die Wahrscheinlichkeiten für die drei möglichen Kategorien ausgibt – männlich, weiblich und unbekannt.

Ein großer Teil der Methodik bei der Anwendung von neuronalen Netzwerken ist die Vorbereitung („preprocessing“) der Daten. In unserem Fall bestand das *preprocessing* aus mehreren Schritten.

Nachdem die Sätze des Wikipedia-Datensets gesampelt wurden, haben wir für jeden Satz die Syntax extrahiert. Hierfür haben wir das Tool „Stanza“ aus dem NLP-Lab der Stanford University genutzt (Qi et al. 2020). Stanza nutzt zur Bestimmung von syntaktischen Abhängigkeiten in Sprache die Konventionen des „Universal Dependencies“ Projektes (vgl. Abb. 2).[2] Das bedeutet, dass die grammatikalischen Bezie-

[2] Vgl. https://universaldependencies.org/.

hungen von Sätzen als eine Baumstruktur dargestellt werden, deren Wurzelelement immer das Verb des Satzes ist. Subjekt und Objekt bilden jeweils Blattelemente dieser Baumstruktur, welche von der Wurzel abhängen. Dieser Schritt verlief vollautomatisch.

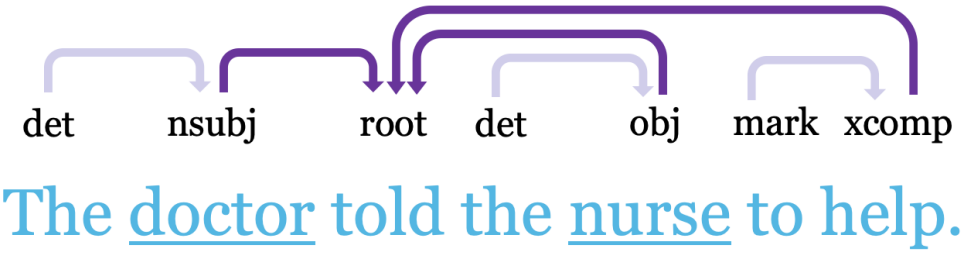

Abbildung 2: Die Grammatik eines Beispielsatzes in der Syntax des Universal Dependencies-Projektes

Das Verb bildet die Wurzel der Baumstruktur, während Subjekt – nsubj – und Objekt – obj – als abhängige Konstruktionen davon abhängen.

Im nächsten Schritt musste der Datensatz mit den entsprechenden Kategorien versehen werden. Hier haben wir uns für einen semi-automatischen Ansatz entschieden. Da wir bereits wussten, welche Worte Subjekt und Objekt der Sätze im Datensatz waren, konnten wir diese Informationen nutzen, um gebräuchlichen Vornamen und Pronomen bereits automatisch das korrekte Geschlecht zuzuweisen.[3] Alle Subjekte und Objekte, die mittels dieser Listen nicht zugeordnet werden konnten, haben wir händisch bestimmt.

Mit den so vorbereiteten Daten konnten wir simultan zwei LSTM-Netzwerke trainieren.

Ergebnisse

Unsere Netzwerke haben wir für 10 Epochen mit dem Datensatz trainiert. Insgesamt erreichen wir mit beiden Netzwerken über 75 % Genauigkeit. Wir haben die Netzwerke in mehreren Konfigurationen trainiert (vgl. Tab. 1), um zu testen, welche Features dem Netzwerk bei der Klassifikation behilflich sind.

Tabelle 1: Ergebnisse des Trainings von vier verschiedenen Netzwerkkonfigurationen nach 10 Epochen

Features	Präzision	F1-Score
Unmodifiziertes LSTM*	71,29 %	38,09 %
LSTM + GloVe	73,65 %	42,84 %
LSTM, GloVe + Grammatik*	75,53 %	46,38 %
LSTM, GloVe, Grammatik + Koreferenz-Auflösung	69,51 %	41,26 %

Note: Der Asterisk markiert Embeddings, die nicht vortrainiert wurden.

[3] Die Listen für Vornamen haben wir aus dem folgenden Artikel entnommen: https://www.geeksforgeeks.org/python-gender-identification-by-name-using-nltk/. Die Pronomen stammen aus dem folgenden GitHub-Repository: https://github.com/nikhgarg/EmbeddingDynamicStereotypes/tree/master/data.

In einer ersten Konfiguration haben wir die Netzwerke im Ausgangszustand belassen, das heißt: Die benutzten *word embeddings* waren mit Zufallszahlen vorbesetzt und nicht vortrainiert. Wir haben die Fähigkeit des Netzwerkes zur Koreferenz-Auflösung ausgeschaltet und das Training mit einer Lernrate von 0,01 gestartet. In dieser Einstellung erreichten die Netzwerke durchschnittlich 71 % Genauigkeit und einen F1-Score von rund 38 %.

Für die zweite Konfiguration haben wir die lokal trainierten *word embeddings* mit Vektoren des „Global Vectors for Word Representations" (GloVe)-Projektes genutzt (Pennington et al. 2014). Das Verwenden dieser – mit einem weit größeren Datensatz trainierten Vektoren – hat die Präzision um rund zweieinhalb Prozentpunkte erhöht und den F1-Score um beinahe 5 % verbessert.

Bis zu diesem Zeitpunkt handelte es sich um ein reguläres LSTM-Netzwerk. Für eine dritte Konfiguration haben wir nun die Grammatik-Embeddings beim Training hinzugezogen (vgl. Abb. 1). Dies hat die Genauigkeit des Netzwerkes nochmals um rund zwei Prozentpunkte gesteigert und den F1-Score auf über 46 % erhöht.

Zuletzt haben wir in einer vierten Konfiguration die Koreferenz-Auflösung des Netzwerkes aktiviert. Dies hat allerdings zu einer Reduktion von Präzision und F1-Score geführt. Dies kann mehrere Gründe haben. Zum einen bestanden die Biografien, mit welchen wir das Netzwerk trainiert haben, aus wenigen Sätzen. Dadurch konnte die Koreferenz-Resolution wenig bei der Klassifikation helfen. Zum anderen wurde im Plenum angemerkt, dass Koreferenz-Resolution mit Bezug auf das Geschlecht von Subjekten auch rückwärts stattfinden könnte (d. h. dass ein Pronomen im ersten Satz nicht bestimmbar ist, aber das Geschlecht in einem darauffolgenden Satz spezifiziert wird). Unser Netzwerk implementiert nur die Vorwärts-Auflösung.

Zuletzt haben wir das beste Netzwerk genutzt, um die Geschlechtsinformationen aus zwei anderen Datensätzen zu extrahieren: 300 zufällige Sätze aus einem New York Times-Datenset sowie 300 zufällige Sätze aus U.S.-Kongressreden. Wir haben die Vorhersagen des Netzwerkes manuell überprüft und daraus die Präzision mit diesen unbekannten, neuen Daten berechnet. Die Präzision bei diesen Datensätzen lag bei 54 %, was beweist, wie wichtig das Training eines neuronalen Netzwerkes mit korrekten Daten ist. In unserem Fall war der „out-of-domain error"[4] erheblich.

Tabelle 2: Relative Häufigkeiten der drei Geschlechts-Labels in den drei untersuchten Korpora

Datensatz	Wikipedia-Biografien		Congressional Records		New York Times	
	Subjekte	Objekte	Subjekte	Objekte	Subjekte	Objekte
Männlich	45,7 %	8,9 %	38,4 %	20,4 %	34,7 %	16,1 %
Weiblich	37,8 %	7,3 %	26,9 %	10,7 %	31,0 %	12,2 %
Unbekannt	16,5 %	83,8 %	34,7 %	68,9 %	34,3 %	71,7 %

Dennoch lässt sich – mit der entsprechenden Vorsicht – feststellen, dass weiblich gelesene Personen in Artikeln der New York Times häufiger in der Subjektposition stehen, damit also häufiger handeln als in U.S.-Kongressreden, was so ein erwartbares Ergebnis ist (vgl. Tab. 2). Ebenso ist erkenntlich, dass während die Wikipedia-Biografien kaum „unbekannte" Subjekte enthalten, dieser Anteil in sowohl den Congressional Records als auch in den New-York-Times-Artikeln wesentlich höher ist. Auch das ist ein

[4] Der Trainings-Datensatz eines neuronalen Netzwerkes bestimmt die „Domain" des Netzwerkes. Wird ein Netzwerk beispielsweise nur mit englischen Biografien trainiert, ist das die Domain. Das heißt, das Netzwerk kann englische Biografien klassifizieren. Wird nun ein Datensatz aus Zeitungsartikeln mit dem Netzwerk klassifiziert, kann es passieren, dass das Netzwerk Schwierigkeiten bei der Klassifikation hat, da es nicht derselbe Sprachstil ist, den es während des Trainings erlernt hat. Dies bezeichnet man als „out-of-domain error".

erwartbares Ergebnis, da die Biografien explizit ausgesucht wurden als ein Datensatz, in welchem es hauptsächlich um Personen geht. In Kongressreden wie aber auch in journalistischen Artikeln werden oftmals auch Gesetze oder Unternehmen besprochen, welche kein Geschlecht haben.

Diskussion

Dieser Artikel stellt eine neue Methode vor, um mithilfe von LSTM-Netzwerken Sätze zu klassifizieren. Sie ordnet sich zwischen die Verwendung von Transformern und *dependency parsing*-basierten Methoden ein. Wir konnten zeigen, dass das Training eines solchen Netzwerkes auch mit wenig Daten möglich ist. Eine wichtige Frage ist allerdings, wann die Methode nützlich ist. Transformer sind weitaus präziser und – je nach Situation und verfügbaren Ressourcen – einfacher zu trainieren.

Derzeit jedoch ist das Training von Transformer-Modellen auf persönlichen Laptops noch nicht handhabbar.[5] Daher kann die Verwendung von LSTM-Netzwerken vor allem in Situationen mit begrenzter Rechenkapazität Sinn ergeben. Es ist allerdings wichtig, die Vorbereitung für das Training korrekt durchzuführen. Hier lassen sich aus den Ergebnissen dieser Fallstudie Empfehlungen ableiten.

Erstens sollten *word embeddings* auf einem möglichst großen Datensatz vortrainiert werden. Das GloVe-Projekt bietet *word embeddings* für eine Reihe von Sprachen an. Sind keine vortrainierten *word embeddings* verfügbar, sollten *word embeddings* mit einem möglichst großen, eigenen Datensatz vortrainiert werden. Diese Vorarbeit muss allerdings nur einmal geschehen und ist mit gängigen Methoden schnell und effizient umzusetzen. Gleiches gilt für das Vortraining von Grammatik-Embeddings.

Zweitens ist es wichtig, eine ausreichende Menge an Trainingsdaten zu haben. Während Transformer teilweise mit wenigen hundert Beispielen bereits akkurate Vorhersagen treffen können, benötigen LSTM-Netzwerke substanziell mehr Datenmaterial. Hierbei darf die geringe Menge unseres Datensatzes in diesem Fall nicht darüber hinwegtäuschen, dass unsere Kategorie von Interesse bereits in Teilen explizit im Text vorhanden war. Für abstraktere Klassifikationsziele wie beispielsweise in „populistisch" oder „nicht populistisch" liegt die notwendige minimale Datenmenge wahrscheinlich wesentlich höher.

Basierend auf den Ergebnissen dieser Studie empfehlen wir die Nutzung von LSTM-Modellen nur in bestimmten Fällen. Es bieten sich zwei Forschungsdesiderata an.

Zum einen sollte ein genauerer Blick auf die Leistung von LSTM-Netzwerken mit Blick auf ressourcenarme Sprachen gelegt werden. Denn hier könnten LSTM-Netzwerke in der Tat einen großen Vorteil gegenüber Transformer-Modellen haben. Wie zahlreiche Studien bereits gezeigt haben, sind Transformer den kleineren LSTM-Netzwerken in ressourcenreichen Sprachen wie Englisch, Deutsch, Französisch, Spanisch oder Chinesisch weit überlegen. Doch bei vielen Sprachen mit nur wenig digitalisiertem Text könnten Transformer aufgrund ihrer hohen Parameterdichte stark an Leistung verlieren.

Zum anderen bietet die höhere Modularität mehr Möglichkeiten, LSTM-Netzwerke besser theoretisch zu leiten. Je nach theoretischen Vorüberlegungen können Variablen selektiv hinzugezogen bzw. ausgeschlossen werden. Transformer-Modelle müssen extern in Rechenzentren vortrainiert werden. Während des anschließenden Finetunings haben Wissenschaftler*innen dann nur sehr begrenzte Möglichkeiten, weitere Daten in die Vorhersage einfließen zu lassen.

[5] Zum Vergleich: Das Training des hier verwendeten LSTM-Netzwerkes benötigt auf einem handelsüblichen Computer (MacBook Pro M1 2020, 8 GB Speicher) durchschnittlich rund 5 Minuten, während ein Transformer, der mit dem gleichen Aufwand und auf dem gleichen Datensatz trainiert wird, mehrere Stunden benötigt. Diese Zahlen sind lediglich anekdotische Evidenz, zeigen jedoch den erheblichen Unterschied in der Komplexität beider Netzwerktypen.

Literatur

Baur, Nina, und Jörg Blasius, Hrsg. 2019. *Handbuch Methoden der empirischen Sozialforschung*. Wiesbaden: Springer Fachmedien Wiesbaden.

Bender, Emily M., Timnit Gebru, Angelina McMillan-Major und Margaret Mitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? 🦜. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 610–623. Virtual Event Canada: ACM https://dl.acm.org/doi/10.1145/3442188.3445922 (Zugegriffen: 20. Apr. 2021).

Blei, David M, Andrew Y. Ng und Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3:993–1022.

Bonikowski, Bart, Yuchen Luo und Oscar Stuhler. 2022. Politics as Usual? Measuring Populism, Nationalism, and Authoritarianism in U.S. Presidential Campaigns (1952–2020) with Neural Language Models. *Sociological Methods & Research* 51:1721–1787.

Breiman, Leo. 2001. Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). *Statistical Science* 16:199–231.

Brown, Tom B. et al. 2020. Language Models are Few-Shot Learners. *arXiv:2005.14165 [cs]*.

Chang, Kent K., und Simon DeDeo. 2020. Divergence and the Complexity of Difference in Text and Culture. *Journal of Cultural Analytics* 4(11):1–36.

Däubler, Thomas, und Kenneth Benoit. 2022. Scaling hand-coded political texts to learn more about left-right policy content. *Party Politics* 28:834–844.

Do, Salomé, Étienne Ollion und Rubing Shen. 2022. The Augmented Social Scientist: Using Sequential Transfer Learning to Annotate Millions of Texts with Human-Level Accuracy. *Sociological Methods & Research* doi: 00491241221134526.

Eisenstein, Jacob. 2018. *Natural Language Processing*.

Franzosi, Roberto. 1989. From Words to Numbers: A Generalized and Linguistics-Based Coding Procedure for Collecting Textual Data. *Sociological Methodology* 19:263–298.

Garg, Nikhil, Londa Schiebinger, Dan Jurafsky und James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences* 115:E3635–E3644.

Gentzkow, Matthew, Jesse M. Shapiro und Matt Taddy. 2019. Measuring Group Differences in High-Dimensional Choices: Method and Application to Congressional Speech. *Econometrica* 87:1307–1340.

Gerow, Aaron, Yuening Hu, Jordan Boyd-Graber, David M. Blei und James A. Evans. 2018. Measuring discursive influence across scholarship. *Proceedings of the National Academy of Sciences* 115:3308–3313.

Grimmer, Justin, Margaret E. Roberts und Brandon M. Stewart. 2022. *Text as data: a new framework for machine learning and the social sciences*. Princeton, New Jersey Oxford: Princeton University Press.

Hochreiter, Sepp, und Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9:1735–1780.

Jurafsky, Daniel, und James H. Martin. 2020. *Speech and Language Processing (DRAFT)*. https://web.stanford.edu/~jurafsky/slp3/ (Zugegriffen: 1. Aug. 2023).

Knight, Carly. 2022. When Corporations Are People: Agent Talk and the Development of Organizational Actorhood, 1890–1934. *Sociological Methods & Research* doi: 00491241221122528.

Komninos, Alexandros, und Suresh Manandhar. 2016. Dependency Based Embeddings for Sentence Classification Tasks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1490–1500. San Diego, California: Association for Computational Linguistics https://www.aclweb.org/anthology/N16-1175 (Zugegriffen: 13. Feb. 2021).

Kozlowski, Austin C., Matt Taddy und James A. Evans. 2019. The Geometry of Culture: Analyzing the Meanings of Class through Word Embeddings. *American Sociological Review* 84:905–949.

Lankford, Seamus, Haithem Alfi und Andy Way. 2021. Transformers for Low-Resource Languages: Is Féidir Linn! In *Proceedings of Machine Translation Summit XVIII: Research Track*, 48–60. Virtual: Association for Machine Translation in the Americas https://aclanthology.org/2021.mtsummit-research.5 (Zugegriffen: 1. Feb. 2023).

Lazer, David, und Jason Radford. 2017. Data ex Machina: Introduction to Big Data. *Annual Review of Sociology* 43:19–39.

Lebret, Rémi, David Grangier und Michael Auli. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1203–1213. Austin, Texas: Association for Computational Linguistics https://aclanthology.org/D16-1128 (Zugegriffen: 31. Jan. 2022).

Levy, Omer, und Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 302–308. Baltimore, Maryland: Association for Computational Linguistics https://www.aclweb.org/anthology/P14-2050 (Zugegriffen: 13. Feb. 2021).

Metz, Cade. 2016. An Infusion of AI Makes Google Translate More Powerful Than Ever. *Wired*, September https://www.wired.com/2016/09/google-claims-ai-breakthrough-machine-translation/ (Zugegriffen: 1. Feb. 2023).

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado und Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *Advances in Neural Information Processing Systems* 26:3111–3119.

Mohr, John W. et al. 2020. *Measuring Culture*. New York: Columbia University Press.

Mohr, John W. 1998. Measuring Meaning Structures. *Annual Review of Sociology* 24:345–370.

Mohr, John W., Robin Wagner-Pacifici, Ronald L. Breiger und Petko Bogdanov. 2013. Graphing the grammar of motives in National Security Strategies: Cultural interpretation, automated text analysis and the drama of global politics. *Poetics* 41:670–700.

Mosteller, Frederick, und David L. Wallace. 1963. Inference in an Authorship Problem: A Comparative Study of Discrimination Methods Applied to the Authorship of the Disputed *Federalist* Papers. *Journal of the American Statistical Association* 58:275–309.

Nelson, Laura K. 2021. Leveraging the alignment between machine learning and intersectionality: Using word embeddings to measure intersectional experiences of the nineteenth century U.S. South. *Poetics* 88:1–14.

Nelson, Laura K., Derek Burk, Marcel Knudsen und Leslie McCall. 2018. The Future of Coding: A Comparison of Hand-Coding and Three Types of Computer-Assisted Text Analysis Methods. *Sociological Methods & Research* doi: 0049124118769114.

Pennington, Jeffrey, Richard Socher und Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. Doha, Qatar: Association for Computational Linguistics http://aclweb.org/anthology/D14-1162 (Zugegriffen: 18. Mai 2022).

Perry, Patrick O., und Kenneth Benoit. 2017. Scaling Text with the Class Affinity Model. http://arxiv.org/abs/1710.08963 (Zugegriffen: 28. Nov. 2022).

Qi, Peng, Yuhao Zhang, Yuhui Zhang, Jason Bolton und Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. *arXiv:2003.07082 [cs]*.

Roberts, Margaret E. et al. 2014. Structural Topic Models for Open-Ended Survey Responses. *American Journal of Political Science* 58:1064–1082.

Sandhaus, Evan. 2008. The New York Times Annotated Corpus. 3250585 KB. https://catalog.ldc.upenn.edu/LDC2008T19 (Zugegriffen: 31. Jan. 2022).

Stuhler, Oscar. 2022. Who Does What to Whom? Making Text Parsers Work for Sociological Inquiry. *Sociological Methods & Research* doi: 00491241221099551.

Trübner, Miriam, und Andreas Mühlichen. 2019. Big Data. In *Handbuch Methoden der empirischen Sozialforschung*, vol. 1, Hrsg. Nina Baur und Jörg Blasius, 143–158. Wiesbaden: Springer.

Vaswani, Ashish et al. 2017. Attention Is All You Need. *arXiv:1706.03762 [cs]*.

Whittaker, Meredith. 2021. The steep cost of capture. *Interactions* 28:50–55.