Universität Trier
WiSe 2021/2022
Fachbereich II – Computerlinguistik und Digital Humanities
Seminar: Programmieren I: Textprozessieren
Dozentin: Ariadne Baresch

Hausarbeit zum Abschluss des Moduls:

PROGRAMMIEREN 1: TEXTPROZESSIEREN

Modulkürzel nach PORTA: MA2DHU1009

Sentiment Analysis in Zeiten von Covid19

Verfasserin: Miriam Coccia
Matrikelnummer: 1565760
Anschrift: Januarius-Zick-Straße, 48
54296 Trier
E-Mail-Adr.: s0micocc@uni-trier.de

Im Rahmen des Moduls habe ich außerdem die Veranstaltung

1) Programmieren I: Textprozessieren (Prof. Dr. Christof Schöch)

besucht.

Ich versichere, dass ich meine Hausarbeit selbstständig angefertigt, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und alle wörtlich oder sinngemäß übernommenen Textstellen als solche kenntlich gemacht habe.

Mir ist bekannt, dass die ungekennzeichnete Übernahme fremder Texte – auch aus dem Internet – als Täuschung gewertet wird und die entsprechende Prüfungsleistung als nicht erbracht gilt.

Trier, den 14. April 2022 Unterschrift:

Inhaltsverzeichnis

1. Einleitung[1]

Am 31. Dezember 2019 wurde die Weltgesundheitsorganisation (WHO) über die ersten Fälle des Coronavirus in der chinesischen Stadt Wuhan informiert (vgl. WHO Health Topics, 2020). In den darauffolgenden Monaten breitete sich das Virus auf allen Kontinenten aus und verursacht bis heute mit über rund 340 Millionen Infizierten und mehr als 5,5 Millionen Toten in Zusammenhang mit Covid-19 eine enorme Anzahl an Ansteckungen und Todesfällen (vgl. Daube 2022, 1). Um eine unkontrollierte und zu schnelle Ausbreitung der Pandemie zu verhindern, ergriffen viele Regierungen restriktive Maßnahmen, die von der verordneten Schließung bestimmter Geschäfte und Einrichtungen bis hin zu Ausgangssperren reichten (vgl. Fuchs 2022, 10, 89–90). Diese aufgrund der Pandemie getroffenen Maßnahmen und die daraus entstandenen Einschränkungen in vielen Lebensbereichen blieben und bleiben auch aktuell nicht ohne ökonomische und gesellschaftliche Folgen (vgl. Daube 2022, 1). So verloren u.a. viele Menschen ihren Arbeitsplatz und die soziale Distanzierung verstärkte allgemein das Gefühl von Unsicherheit und Einsamkeit sowie psychische Erkrankungen wie Depressionen (vgl. Fuchs 2022, 11, 89). Die Debatte um den richtigen Umgang mit der Pandemie hat in der Gesellschaft zudem zu sozialen Konflikten und einer Art Spaltung der Gesellschaft (vgl. Frei, Schäfer, und Nachtwey 2021, 249). Während ein großer Teil der Menschen die Maßnahmen zur Bekämpfung des Corona-Virus befürwortet und sich gegen das Virus impfen lässt (vgl. Beckmann und Schönauer 2021, 6; Daube 2022, 1), gibt es auch eine Reihe an Menschen, die die Corona-Maßnahmen und eine Impfung kritisieren und sogar entschieden ablehnen (vgl. Frei, Schäfer, und Nachtwey 2021, 249). Die sehr heterogenen Einstellungen und Meinungen zur Corona-Pandemie werden in den westlich-demokratisch ausgerichteten Ländern von den unterschiedlichen Gruppierungen u.a. während öffentlichen Corona-Demonstrationen, die sich für oder gegen die Maßnahmen und eine Impfung richten, kundgegeben (vgl. ebd., 250; Tagesschau 2022). Darüber hinaus hat die Covid19-Pandemie vor allem durch die soziale Distanzierung zu einer zunehmenden Veränderung der alltäglichen Kommunikationsweise der Menschen beigetragen (vgl. Fuchs 2022, 19). Dadurch hat die Debatte über die Corona-Maßnahmen und die Covid19-Impfungen einen großen Raum im Internet und vor allem in den

[1] In der vorliegenden Arbeit wird im Fließtext zitiert. Die Zitation erfolgt mit Hilfe von Zotero und der verwendete Zitierstil ist der von der Modern Humanities Research Association (MHRA). Außerdem wird das generische Maskulinum verwendet. Dieses bezieht sich auf die männliche, weibliche und alle anderen Geschlechteridentitäten.

sozialen Medien eingenommen (vgl. ebd., 89). Obwohl es sich bei der Abneigung gegen Impfungen keineswegs um ein während der Covid-19-Pandemie neu entstandenes Phänomen handelt, hat die Online-Präsenz sogenannter „Anti-Vaxxer“-Gruppen seit 2019 deutlich zugenommen (vgl. Meyer und Reiter 2004, 1183). So haben die von Impfskeptikern geführten Social-Media-Accounts bereits zwischen 2019 und 2020 mindestens 7,8 Millionen neue Mitglieder gewonnen (vgl. Burki 2020, 504). Analysen, die sich auf Daten aus sozialen Medien stützen, sind deshalb sehr gut geeignet, um die Einstellungen der Nutzer zum Thema Impfen zu ermitteln. Eine der am weitesten verbreiteten Techniken zur Meinungsermittlung ist dabei die Sentiment Analysis. Diese kann allgemein in einer Vielzahl von Bereichen eingesetzt werden wie beispielsweise im Gesundheitswesen, bei gesellschaftlichen Ereignissen und sogar bei politischen Wahlen (vgl. Liu 2012, 9). Insbesondere wird sie jedoch heutzutage im Kontext der sozialen Medien angewandt – hier vor allem von Unternehmen und Organisationen, um die Meinungen der Öffentlichkeit oder von Kunden über ihre Produkte und Dienstleistungen zu erforschen (vgl. ebd., 8). Aber kann die Sentiment Analysis auch eingesetzt werden, um die Einstellung zur Covid-19-Impfung in den sozialen Medien zu ermitteln? Dies soll in der vorliegenden Arbeit exemplarisch auf der Grundlage von YouTube-Kommentaren des YouTube-Kanals Deutsche Welle (DW) beantwortet werden. Die YouTube-Kommentare werden hierbei als Datengrundlage aus YouTube extrahiert und für das Training eines auf maschinellem Lernen basierenden Sentiment Analysers verwendet. In der vorliegenden Arbeit wird im Folgenden zunächst ein kurzer Forschungsüberblick gegeben, in dem vergleichende Studien vorgestellt werden. Dann werden die Verfahren der Datenextraktion und der Datenauszeichnung sowie des Aufbaus eines neuronalen Netzes und dessen Trainings beschrieben. Hierbei wird auch auf die jeweiligen Problematiken eingegangen werden. In einem weiteren Schritt werden dann die Ergebnisse der Analyse präsentiert. Die Arbeit wird mit einem kurzen Ausblick und einem Fazit abgeschlossen.

2. Forschungsstand

Mit Millionen von aktiven Nutzern pro Monat bieten beliebte Social-Media-Plattformen wie Facebook, YouTube und Twitter nützliches Material, um die Meinung der Menschen zu aktuellen Ereignissen und Themen – wie beispielsweise auch der Covid19-Pandemie – zu beobachten und zu analysieren (vgl. Kemp 2022, 99). Die Nutzung sozialer Medien als

Ausgangspunkt für die Sentiment Analysis ist deshalb ein weit verbreiteter Ansatz in der Forschung.

Während bisher jedoch keine einschlägigen Studien YouTube-Kommentare als Datengrundlage nutzten, wurden in den vergangenen Jahren vor allem Twitter-Daten als Grundlage mehrerer Analysen verwendet.[2] So wurden in einer im Jahre 2022 von Bjarke Mønsted und Sune Lehmann durchgeführten Studie beispielsweise Twitter-Accounts analysiert, deren Nutzer durchweg eine positive oder negative Einstellung zur Covid19-Impfung zum Ausdruck brachten. Ziel der Studie war es, die Polarität der Aussagen zu klassifizieren und letztlich das Zusammenspiel zwischen Meinungen über die Impfung, der Struktur des sozialen Netzwerks und Online-Informationen aufzuzeigen (vgl. Mønsted und Lehmann 2022, 1–19).

Des Weiteren führten Philipp Wicke und Marianna M. Bolognesi in ihrer im Jahre 2021 veröffentlichten Studie eine diachrone Analyse von Twitter-Daten vom 20. März bis zum 1. Juli 2020 über die Coronavirus-Pandemie durch und waren in der Lage, die Veränderung des sprachlichen Diskurses mit Hilfe des Topic modelings zu messen. Die Polarität der Tweets wurde in diesem Fall mit Hilfe des in der „TextBlob“-Bibliothek enthaltenen und bereits vortrainierten Sentiment Analysers gemessen (vgl. Wicke und Bolognesi 2021, 1–2).

Obwohl das Thema der skizzierten Studien und die Methode zur Erstellung des Datensatzes mit der in der vorliegenden Arbeit verwendeten Methode vergleichbar sind, gibt es doch wesentliche Unterschiede in Bezug auf die verwendete Datengrundlage, die Methode der Sentiment Analysis und die Ausrichtung der Arbeit. So weisen das in den angegebenen Studien für die Datenextraktion verwendete Medium Twitter und dessen Tweets besondere Merkmale auf, die sich grundlegend von den Charakteristiken der in der vorliegenden Studie als Datengrundlage verwendeten YouTube-Kommentare unterscheiden. Hier ist beispielsweise die maximale Länge der Tweets, die bei Twitter auf 280 Zeichen begrenzt ist, anzuführen, wodurch eine Verdichtung der Nachricht erzwungen wird (vgl. Twitter Counting characters). Außerdem bietet Twitter die Möglichkeit der Verwendung von Hashtags, um einen Tweet in einen bestimmten Kontext zu stellen und die Nachverfolgung eines Themas zu erleichtern (vgl. Twitter Hashtags). Darüber hinaus sind in Tweets nicht selten URLs zu finden,

[2] Im Folgenden werden nur die aktuellsten Studien angeführt.

die auf Nachrichtenquellen oder andere Tweets verweisen. Diese Besonderheiten des Mediums wurden in der Studie von Mønsted und Lehmann genutzt, um die Interaktionen zwischen Twitter-Nutzern zu erfassen und in einem Netzwerk darzustellen (vgl. Mønsted und Lehmann 2022, 6). Die von YouTube-Nutzern als Reaktion auf die auf YouTube hochgeladenen Videos verfassten Kommentare haben hingegen keine Längenbegrenzung. Auch Kommentare, die Links zu anderen Seiten enthalten, sind nicht üblich, da sie als Spam gewertet werden können (vgl. YouTube Richtlinien). Was die Methode betrifft, so wird in der vorliegenden Studie ähnlich wie bei Mønsted und Lehmann für die Klassifizierung der Sentiment Analysis ein selbst entwickeltes neuronales Netz implementiert. Wicke und Bolognesi haben hingegen – wie bereits erwähnt – in ihrer Studie auf den vortrainierten Sentiment Analyser aus der „TextBlob"-Bibliothek zurückgegriffen.

3. Datenerfassung und -verarbeitung[3]

3.1. Datenextraktion

Die Auswahl der Datengrundlage erfolgte wie folgt: Zunächst wurde als Grundlage für die Materialfindung der YouTube-Kanal DW News bestimmt. Als Deutschlands Auslandssender mit Inhalten in 30 Sprachen zählt er zu den erfolgreichsten YouTube-Kanälen der öffentlich-rechtlichen Sender in Deutschland und hat mit seinen 3,78 Millionen Abonnenten eine große Reichweite (vgl. DW Home) [4]. Aus der Kanalinfo geht darüber hinaus hervor, dass der Kanal unabhängig über gesellschaftliche, politische und wirtschaftliche Entwicklungen in Deutschland und Europa berichtet und dabei deutsche und andere Perspektiven einbringt (vgl. DW About). Als Datengrundlage dienen die Nutzerkommentare, welche als Reaktion auf zwei englischsprachige Playlists des YouTube-Kanal DW News verfasst wurden. Die Videos beider Wiedergabelisten mit den Titeln „COVID 19 Special - Coronavirus scientific background and latest developments" und „Coronavirus" (vgl. DW) konzentrieren sich dabei inhaltlich auf die Covid19-Pandemie. Die bereits vorgegebene Aufbereitung der Videos in zwei thematische Wiedergabelisten erleichterte darüber hinaus die Extrahierung der über 7.000 Nutzerkommentare.

[3] Die in diesem und in den Folgekapiteln im Text angegebenen Zeilenangaben beziehen sich auf die entsprechenden Zeilen des sich im Anhang der Arbeit befindenden Codes.

[4] Im Vergleich dazu hat der YouTube Kanal der Tagesschau beispielsweise nur 1,19 Millionen Abonnenten (vgl. https://www.youtube.com/user/tagesschau) und der Kanal des SWR 466.000 Abonnenten (vgl. https://www.youtube.com/c/SWR).

Um die Daten automatisch aus dem Web zu extrahieren, stehen zwei häufig verwendete Verfahren zur Verfügung: Web Scraping und API-Aufrufe (Application Programming Interface). Ersteres wird „most commonly accomplished by writing an automated program that queries a web server, requests data (usually in the form of HTML and other files that compose web pages), and then parses that data to extract needed information“ (Mitchell 2018, X). Web Scraping erfordert somit den Einsatz von Bots, sogenannten „Crawlern“, um nützliche Daten direkt von einer oder mehreren gewünschten Webseiten abzurufen. Im Gegensatz zum Web Scraping sind APIs eine Schnittstelle oder „a point where two systems, subjects, organisations, and so forth meet and interact“ (Lauret 2019, Kapitel 1.1.1). Hierbei wird somit anhand eines Programms die API aufgerufen, um auf die auf einem bestimmten Webserver gespeicherten Daten zuzugreifen. Bei korrekter Ausführung des Aufrufs, gibt die API die angeforderten Daten an das Programm zurück. Im Gegensatz zum Web Scraping, bei dem die extrahierten Daten zuerst geparst werden müssen, ermöglicht ein API-Aufruf dem Programm, nur die benötigten Daten abzurufen, indem vom Nutzer im Programm die Parameter des Aufrufs angepasst und die benötigten Elemente aus der Ausgabe des Aufrufs ausgewählt werden. Die Daten werden normalerweise im JSON-Format zurückgegeben. Sowohl Web Scraping als auch API-Aufrufe haben ihre Vor- und Nachteile, die projektabhängig sind. So kann Web Scraping beispielsweise nützlich und recht praktisch sein, um auf Daten aus Quellen zuzugreifen, die keine API haben, oder um Informationen über geschützte Daten zu sammeln. Da YouTube jedoch über eine API verfügt, konnte in der vorliegenden Arbeit von dieser Gebrauch gemacht werden, um die Nutzerkommentare zu extrahieren (vgl. YouTube Data API Overview).

Um die von Google zur Verfügung gestellte YouTube Data API für eine Anwendung nutzen zu können, ist ein API-Schlüssel erforderlich (vgl. YouTube Obtaining authorization credentials). Hierbei handelt es sich um ein Token oder Passwort, das eine erfolgreiche Authentifizierung und Identifizierung des eigenen Projekts ermöglicht, um auf die API zugreifen zu können. Dieser API-Schlüssel ist erhältlich, nachdem ein neues Projekt auf der Google Cloud Platform (GCP) erstellt wurde, und kann im Anschluss als Konstante in dem Python-Projekt gespeichert werden. In dem bearbeiteten Projekt wurde der API-Schlüssel in einer Umgebungsvariable abgespeichert (Zeile 18), damit sie vor Dritten sicher in dem von der Anwendung verwendeten System gespeichert werden konnte. Der API-Schlüssel reicht jedoch nicht aus, um die YouTube-Daten-API aufrufen zu können. Hierzu ist zusätzlich die Installation der Bibliothek

„google-api-python-client“ (vgl. Google API) sowie der Import der build()-Funktion aus dem Modul discovery innerhalb der Bibliothek „googleapiclient“ erforderlich.

Die Interaktion mit der YouTube-API erfordert darüber hinaus ein Service-Objekt, welches durch die build()-Funktion instanziiert wird (vgl. ebd.). Das Service-Objekt (gespeichert in der Variablen yt_service, Zeile 102) bietet Methoden, die für den Zugriff auf die API-Daten verwendet werden können. Im vorliegenden Projekt wurden für die Extraktion der linguistischen Daten aus den Videos mehrere aufeinander basierende Funktionen definiert, um zunächst die Kanal-ID, daraufhin die Playlists, dann die Videos in den Playlists und schließlich die Kommentare abzurufen. Diese Herangehensweise war aufgrund des Formats der API-Antwort in JSON erforderlich. Dieses ähnelt der Struktur eines Python-Dictionarys und enthält somit projektrelevante Informationen als Werte von Schlüsseln, die stufenweise extrahiert und als Parameter der folgenden Funktion verwendet werden müssen, um die nächste verschachtelte Information abzurufen.

Die im Projekt angewandten Funktionen sind für folgende Abläufe zuständig: In der ersten Funktion (Zeilen 23–32) wird das Service-Objekt verwendet, um die Kanal-ID, die durch Auswahl der entsprechenden Argumente der Parameter „part“ und „forUsername“ und durch Ausführen der Anfrage ermittelt werden kann, zu extrahieren. Die Funktion gibt dann die Antwort in der Python-Konsole aus, so dass die Channel-ID manuell kopiert werden kann. Da ihr Ergebnis in eine spätere Funktion eingefügt wird, wird die Funktion „get_channel_id()“ nur einmal aufgerufen, nämlich bei der Definition der Funktion „get_playlist()“ (Zeilen 35–51). Wie anhand des Namens der Funktion bereits ersichtlich wird, extrahiert sie relevante Playlists des YouTube-Kanals, welcher mit der zuvor extrahierten Kanal-ID korrespondiert. Trotz der Möglichkeit, eine große Anzahl von Wiedergabelisten abzurufen (deren Anzahl durch das Argument des optionalen Parameters „maxResults“ definiert wird), wurden zielgerichtet nur die für das vorliegende Projekt relevanten Wiedergabelisten ausgewählt, die die Zeichenkette „Coronavirus“ enthalten. Diese Playlists wurden dann in einem Dictionary mit dem Titel der Wiedergabelisten („COVID 19 Special - Coronavirus scientific background and latest developments” und „Coronavirus”) (vgl. DW) als Schlüssel und der Wiedergabelisten-ID als Wert gespeichert. Das Dictionary wird im Anschluss daran von der Funktion zurückgegeben. In diesem Projekt wurde ein Dictionary als Datenstruktur für den Output der Wiedergabelisten-ID verwendet, um die Bedeutung des Titels der Wiedergabeliste, der die

Zeichenfolge „Coronavirus“ enthält, hervorzuheben; die Verwendung einer Liste, die nur die Wiedergabelisten-IDs enthält, wäre jedoch ebenfalls eine Option gewesen, da der Name der Wiedergabeliste im Projekt anschließend nicht weiter verwendet wird.

Die Funktion „get_playlist_video()“ (Zeilen 54–69) benötigt sowohl das Service-Objekt als auch das Dictionary mit den Wiedergabelisten, um die Video-IDs zu extrahieren. Dazu werden die Werte des Playlist-Dictionarys, d. h. die Playlist-IDs, in einer Schleife durchlaufen und ein API-Aufruf unter Verwendung der IDs durchgeführt, um die in der jeweiligen Wiedergabeliste enthaltenen Videos abzurufen. In der innersten for-Schleife wird die Video-ID gefunden und zu einer Liste hinzugefügt. Daraus ergibt sich eine Folge aller Video-IDs, die zur weiteren Verwendung zurückgegeben werden.

Nach dieser Vorarbeit können die Kommentare anhand der Video-IDs abgerufen und dann zu einer Liste hinzugefügt werden, die von der Funktion „get_comments()“ (Zeilen 72–88) zurückgegeben wird. Um die extrahierten Daten verwenden zu können, wird die Funktion „save_comments()“ definiert (Zeilen 91–97). Diese dient dazu, die Kommentare in einem „pandas“-DataFrame (vgl. McKinney u. a. 2010, 51–56) zu speichern und im Anschluss daran eine Methode des DataFrame-Objekts zu verwenden, die die Daten in einer CSV-Datei abspeichert.

Die Funktion, die den Teil der Datenextraktion in diesem Projekt abschließt, trägt den Namen „get_raw_data()“ (Zeilen 100–108). Sie erstellt das Service-Objekt, das die Interaktion mit der YouTube Data API ermöglicht, und verwendet es als Parameter beim Aufruf der zuvor definierten Funktionen, die damit die Extraktion ausführen. Abschließend erhält der Benutzer eine Rückmeldung über die erfolgreiche Datenextraktion.

3.2. Datenverarbeitung

Die in den sozialen Medien von den Benutzern verwendete Sprache weicht in den meisten Fällen von der in Wörter- und Grammatikbüchern festgelegten Standardsprache ab. So auch im Falle der aus der API-Antwort resultierenden Daten: Wie in den folgenden Beispielen deutlich wird, weisen diese typische Merkmale der Internetsprache auf wie beispielsweise die Wiederholung der Zeichensetzung, die Verwendung von Abkürzungen oder die Verwendung von Emoticons (vgl. Haas u. a. 2011, 385):

(1) It will be an endless loop! Forget it ,NOT TAKING IT!!!!!
(2) Not this bs again, we have other problem's now

(3) 💉 =☠

Zusätzlich dazu charakterisieren Rechtschreibfehler, das Vorhandensein von Kommentaren in anderen Sprachen als Englisch, Escapezeichen und eine inkonsistente Großschreibung die Daten. All diese Merkmale stellen ein Problem beim Versuch der Nutzung dieser Daten zum Trainieren eines Algorithmus für maschinelles Lernen dar, denn sie können die Leistung des Algorithmus negativ beeinflussen, insbesondere wie in diesem Fall, in dem der Umfang der Trainingsdaten begrenzt ist (vgl. Camacho-Collados und Pilehvar 2017, 4).

Um die Daten zum Trainieren des Algorithmus nutzbar zu machen, wurden sie zunächst bereinigt und bearbeitet. Die Verfahren zur Datenbereinigung und -verarbeitung sind in der Funktion „clean_data()“ definiert (Zeilen 113–142), welche auf die im DataFrame gespeicherten Kommentare zugreift und eine neue Spalte mit den bereinigten Daten erstellt. In den Kommentaren wurden mit der „unescape()“-Funktion der „html“-Bibliothek Escapezeichen getilgt (vgl. Python HyperText Markup Language Support). Des Weiteren wurden Satzzeichen, Ziffern sowie Stoppwörter entfernt. Für die Entfernung von Stoppwörtern wurde eine vorgefertigte Liste aus dem „corpus“-Modul der „nltk“-Bibliothek verwendet (vgl. NLTK Documentation). Um die Bedeutung der Sätze nicht zu verändern, wurde aus dieser Liste das Stoppwort „not“ entfernt.

Die Lemmatisierung wurde mit derselben Bibliothek durchgeführt. Um das Lemma eines Wortes zu identifizieren, wurde ein Lemmatizer-Objekt aus der „WordNetLemmatizer“-Klasse instanziiert und in Kombination mit einem PoS-Tagger verwendet (vgl. ebd.). Wenn für das Wort keine Wortart gefunden wird, kann keine Lemmatisierung durchgeführt werden.[5] In diesem Fall wird das Wort selbst als Lemma betrachtet. Die Ergebnisse der Lemmatisierung sind in diesem Projekt nicht zufriedenstellend: So wurde nur die Singularform einiger regelmäßiger Plurale gefunden, ein Vorgang, der auch von einem Stemmer hätte durchgeführt werden können (vgl. Willett 2006, 3–4). Außerdem erschwerten die oben erwähnten linguistischen Merkmale der Daten eine erfolgreiche Lemmatisierung. Nichtsdestotrotz war die Anwendung dieser Technik nützlich, um die Dimensionalität der Daten zu reduzieren.

[5] Dieser Teil der Funktion enthält eine Adaptation des Codes im folgenden URL: https://stackoverflow.com/questions/15586721/wordnet-lemmatization-and-pos-tagging-in-python

4. Data Labeling für die Sentiment Analysis

Bei der Charakterisierung der in den YouTube-Kommentaren zum Ausdruck gebrachten Haltung gegenüber Corona-Impfungen wurden in der vorliegenden Studie nur zwei Stimmungen zum Thema Impfung berücksichtigt, nämlich eine positive und eine negative. Durch das Entfernen neutraler oder potenziell nicht relevanter Kommentare kann eine solche Analyse als binäres Klassifikationsproblem bezeichnet werden (vgl. Liu 2012, 31). Da es sich hierbei um ein Verfahren des überwachten maschinellen Lernens handelt, muss die Zugehörigkeit eines Kommentars zu einer der beiden Klassen durch Labels gekennzeichnet werden. Auf diese Weise kann der Algorithmus die Beziehung zwischen den Kommentaren und den Labels erlernen und Vorhersagen für neue Daten treffen.

Um den Kommentaren ein Label zuzuweisen, wurde im Projekt die Funktion „label_data()“ (Zeilen 145–171) definiert. In ihr wurden zwei verschiedene reguläre Ausdrucksmuster erstellt, die für das Projekt bedeutende Schlagwörter aus den YouTube-Kommentaren enthalten („no_vax_pattern“, Zeilen 147–149, und „pro_vax_pattern“, Zeile 150). Diese Muster beziehen sich zum einen auf eine positive Stimmung gegenüber Impfungen sowie ein allgemeines Vertrauen in das Gesundheitssystem, zum anderen auf eine negative Stimmung sowohl gegenüber Impfungen als auch gegenüber der Impfpolitik. Die Wahl der Schlagwörter für die Muster ergab sich aus der manuellen Betrachtung der Daten. So wurden beispielsweise Kommentare, die die Lexeme „scam“ oder „freedom“ enthielten, als negativ gewertet, wohingegen Kommentare, in denen Ausdrücke wie „stay safe“ oder „stay healthy“ erwähnt wurden, als eine eher positive Einstellung gegenüber Impfungen und Impfpräventionsmaßnahmen bewertet wurde.

Das Labeln der Kommentare erfolgte mit Hilfe einer for-Schleife über die Zeilen im DataFrame (Zeilen 153–156 und Zeilen 158–162). In der for-Schleife wird ein Muster innerhalb des Kommentars identifiziert. Zudem wird parallel zum Kommentar in einer neu definierten Spalte des DataFrames eine Nummer eingefügt, um den entsprechenden Kommentar als positiv oder negativ zu kennzeichnen. Kommentare, die kein Label erhielten, weil sie keines der vordefinierten Schlagwörter enthielten, mussten verworfen werden. Zu diesem Zweck wurde den nicht gelabelten Kommentaren zunächst ein NaN-Wert zugewiesen, um diese im Anschluss anhand der dropna-Methode des „pandas“-DataFrame entfernt werden zu können.

5. Sentiment Analysis und maschinelles Lernen

Die gesammelten und gelabelten Daten können verwendet werden, um einen maschinellen Lernalgorithmus zu trainieren.[6] Eine der Techniken des maschinellen Lernens, die aktuell häufig für Sentiment Analysis-Klassifikation verwendet wird, ist das Deep Learning (vgl. Iglesias und Moreno 2019, 1). Obwohl verschiedene Strukturen neuronaler Netze für diese Aufgabe eingesetzt werden können, haben sich LSTM (Long Short Term Memory), eine besondere Form des rekurrenten neuronalen Netzwerks (RNN), besonders bewährt (vgl. Hochreiter und Schmidhuber 1997, 1–30). Da sie nicht so stark vom Problem des verschwindenden Gradienten betroffen sind wie reguläre RNN, sind LSTM in der Lage, längere Sequenzen und nicht unmittelbar zusammenhängende Abhängigkeiten zu lernen (vgl. Sundermeyer, Schülter, und Ney 2012, 195). Aufgrund dieser Eigenschaft sind sie für sprachliche Sequenzen wie beispielsweise YouTube-Kommentare geeignet.

Obwohl die Kommentare von Interpunktion befreit, lemmatisiert und in Kleinbuchstaben umgewandelt wurden, sind weitere Vorverarbeitungsschritte erforderlich, um die Daten erfolgreich als Eingabe für das LSTM-Netz verwenden zu können. Aus diesem Grund wurde die Funktion „data_preprocessing()“ definiert (Zeilen 14–28). In ihr werden zunächst die Kommentare und die Labels abgerufen und dann mit einer Instanz der „Tokenizer“-Klasse aus der „keras“-Bibliothek tokenisiert (vgl. TensorFlow Core Tokenizer; Keras API) .

Durch die Anwendung der Methode „fit_on_texts()“ des Tokenizers auf die Kommentare wird das interne Vokabular des Tokenizers mit den Wörtern aus den Kommentaren aktualisiert. Letzteren wird dadurch ein eindeutiger ganzzahliger Wert zugewiesen. Anschließend werden die Kommentare durch den Aufruf der Methode „text_to_sequences()“ des Tokenizer-Objekts in numerische Sequenzen umgewandelt. Dieser Schritt ist von entscheidender Bedeutung, da Modelle für maschinelles Lernen Daten in Form von Strings nicht verarbeiten können. Im letzten Schritt der Datenvorverarbeitung wird die Funktion „pad_sequences()“ aus der „keras“-Bibliothek verwendet. Der Zweck dieser Funktion besteht darin, die vektorisierten Kommentare in ein zweidimensionales „NumPy“-Array umzuwandeln (vgl. Harris u. a. 2020, 357–362), indem kürzeren Sequenzen ein Wert hinzugefügt und längere gekürzt werden, so dass jede Sequenz die gleiche Länge hat. Nachdem alle diese Schritte ausgeführt wurden, gibt

[6] Dieser Teil des Projekts enthält eine Adaptation des Codes im folgendem URL: https://github.com/nagypeterjob/Sentiment-Analysis-NLTK-ML-LSTM/blob/master/lstm.ipynb (vgl. Nagy 2017)

die „data_preprocessing()“-Funktion die maximale Vokabulargröße, die vektorisierten und aufgefüllten Daten sowie die Originaldaten zurück.

Nach der Vorverarbeitung der Daten wird das LSTM-Netzwerk mit der Funktion „lstm_network()“ erstellt (Zeilen 33–47). Diese erhält als Input die maximale Vokabulargröße und die vektorisierten Daten. Da mit Text gearbeitet wurde, der eine Art von sequentiellen Daten ist, d. h. Daten mit zeitlicher Reihenfolge, wurde eine Instanz der Klasse „Sequential“ aus der „keras“-Bibliothek verwendet. Dieser wurden dann Layer hinzugefügt. Das Embedding-Layer hat drei Input-Parameter: die Dimension des Inputs, d. h. die maximale Vokabulargröße, die Dimension des Outputs, d. h. die Anzahl der Dimensionen, die jeden wortrepräsentierenden Vektor charakterisieren, und die Eingabelänge, d. h. die maximale Dokumentenlänge, die in diesem Fall die Länge der Kommentare ist. Nach dem Embedding-Layer wird eine Regularisierungstechnik namens Dropout angewendet. Hierbei werden Neuronen nach dem Zufallsprinzip abgeschaltet, um ein Overfitting zu vermeiden (vgl. Baldi und Sadowski 2013, 1). Dies bedeutet, dass ein Zustand vermieden werden soll, in dem das neuronale Netz die Trainingsdaten so genau lernt, dass es nicht in der Lage ist, gute Verallgemeinerungen auf andere Daten zu treffen. Danach werden zwei weitere Layer hinzugefügt, eine LSTM-Einheit und ein dense-Layer. Als Parameter nimmt das erste Layer eine ganze Zahl, die die Dimensionalität der Ausgabe beschreibt, und zwei Fließkommazahlen, die den Prozentsatz der Neuronen beschreiben, die aus der Eingabe beziehungsweise aus dem rekurrenten Zustand herausfallen. Das dense-Layer ist die letzte Schicht des Netzes und nimmt als Input zwei Parameter, welche die Dimensionalität des Ausgangsvektors und die zu verwendende Aktivierungsfunktion (im Fall dieses Projektes die Softmax-Funktion) darstellen. Nachdem alle Layer hinzugefügt wurden, kann das Netz kompiliert werden. Als loss-function wird die categorical crossentropy verwendet (vgl. Keras Probabilistic losses) da es sich bei dem Problem, auf das sie angewendet wird, um ein Mehrklassen-Klassifikationsproblem handelt. Als Optimizer wurde für das Modell der Adam-Algorithmus verwendet (vgl. Kingma und Ba 2014, 1–15) und als Metrik zur Berechnung der Leistung wurde die in „keras“ integrierte accuracy-Metrik verwendet. Das kompilierte Modell wird abschließend von der Funktion als Output zurückgegeben, damit es mit den Daten trainiert werden kann.

Um das neuronale Netz zu trainieren, wurde die Funktion „train_lstm()“ definiert (Zeilen 51–75). In ihr wurden die Daten mit Hilfe der Funktion „train_test_split()“ aus der Bibliothek

„scikit-learn“ (vgl. Pedregosa u. a. 2011, 2825–2830) in Trainingsdaten (80 % der Gesamtdaten) und in Testdaten (20 % der Gesamtdaten) aufgeteilt. Dann wurde das Modell anhand der Daten während 7 Epochen trainiert, d.h. das Modell wurde 7 Mal durch den gesamten Trainingsdatensatz geführt, wobei jedes Mal 36 Proben („batch_size“) durch das Netzwerk propagiert wurden. Um zu vermeiden, dass das Modell die Daten nur in die Mehrheitsklasse einordnet und damit einen falschen Eindruck von Genauigkeit vermittelt, wurde ein Dictionary mit Gewichten definiert, das der Minderheitsklasse mehr Bedeutung verleiht und die Verzerrung des Modells eindämmt. Nachdem ein validation-set erstellt wurde, indem ein Teil der Daten aus dem Testsatz entfernt wurde, wurde das Modell durch die fit-Methode trainiert. Darüber hinaus wurde das Modell auf Grundlage des Test-Sets evaluiert und die Ergebnisse wurden in der Konsole angezeigt.

In der folgenden Funktionsdefinition (Zeilen 80–98) wird das Modell implementiert, um Vorhersagen über die Daten im validation-set zu treffen. Zunächst werden vier Zähler erstellt. Diese erfassen nicht nur die Anzahl der Daten, die vom Modell als positiv oder negativ vorhergesagt wurden, sondern zählen auch, wie oft das Modell die Daten korrekt als positiv oder negativ gekennzeichnet hat. Das Modell führt dann Vorhersagen für jedes Element im validation-set durch. Die Korrektheit der Vorhersage wird gemessen, indem die Dimension mit dem Maximalwert zwischen dem Vorhersagevektor und den Testdaten im validation-set verglichen wird. Wenn diese Werte übereinstimmen, wurde eine korrekte Vorhersage getroffen. Um festzustellen, ob das Item korrekt als positiv oder negativ gekennzeichnet wurde, muss die Position des höchsten Wertes im Testvektor beobachtet werden. Sobald das Modell Vorhersagen für alle Elemente des validation-sets gemacht hat und alle Zähler aktualisiert wurden, werden der positive und negative accuracy-score in der Konsole angezeigt.

Schließlich werden alle zuvor definierten Funktionen durch die „sentiment_analysis“-Funktion ausgelöst (Zeilen 103–108).

6. Ergebnisse

Die Ergebnisse zeigen, dass die Leistung des neuronalen Netzes insbesondere im Hinblick auf die korrekte Kennzeichnung von Kommentaren, die eine positive Einstellung zur Covid-19-Impfung einnehmen, sehr schwach ist. Dies ist auf verschiedene Aspekte des Codes und der vorliegenden Daten zurückzuführen.

An dieser Stelle muss zunächst der labeling-Prozess genannt und näher betrachtet werden: Die Ergebnisse des labeling-Prozesses waren in einigen Fällen korrekt. So wurden beispielsweise die folgenden Kommentare, die eine negative Einstellung zur Impfung aufweisen, richtig gekennzeichnet:

(4) I'm unvaccinated person too..to me there's nothing radical about people who refused a politically pedalled impure vaccination incursion
(5) What a scam
(6) The health care works do not feel valued? How about those that are unvaccinated? The health care workers are treating patients bad because they are unvaxxed. It goes both ways.

Beispiele für Kommentare mit einer positiveren Einstellung zu diesem Thema, die korrekt gekennzeichnet wurden, sind hingegen die Folgenden:

(7) Good luck to ALL of the world in having ZERO Covid & Varients cases!!!! Everybody, Pray, Pray, and Pray again. Also, Please get vaccines (if you're able to) and MASK UP!!! 🙏❤🚀 STAY HEALTHY!!!
(8) What, not saying its mild anymore! Can't use the wall street spin anymore because to many dying! Leaving bad taste in your mouth! Or are you feeling a little guilty finally! Hospitals still full here, 48 passed away! Stay safe, and please take it serious

Allerdings darf die Ungenauigkeit dieser Methode nicht übersehen werden. In der Tat wurden viele Kommentare der falschen Klasse zugeordnet, nur weil sie Wörter enthielten, die zum entgegengesetzten Muster gehörten. Beispiele hierfür sind:

(9) It is very confusing that Germany has a 67% fully vaccinated population and still the hospitalization rate is high..... What is the relevance of the vaccine if it doesn't work for countries with more than 50% fully vaccinated population?
(10) I hope all long COVID patients get better. This new drug sounds logic. I am working my MBA thesis on Long COVID. Any help would be really appreciated. Thanks!
(11) I am watching thousands of people dying daily, my country's health system on the brink of collapsing, doctors and nurses exausted, the country's economy going dow, business and people livelihood in danger. But nothing is affecting me directly, then, why should I care? What is important in live is my freedom of being a SELFISH individual.

Beispiel (9) wurde fälschlicherweise als positiv eingestuft, während die rhetorische Frage auf Skepsis schließen lässt. Dem Beispiel (10) hingegen wurde fälschlicherweise ein negatives Etikett zugewiesen, obwohl es sich um einen Kommentar handelt, der den Einsatz des Impfstoffs lobt. Was den Kommentar (11) betrifft, so wurde er allein aufgrund des Lexikons als negativ eingestuft, doch aufgrund der Typografie und der rhetorischen Frage „why should I care“ kann ein menschlicher Leser leicht Elemente von Sarkasmus in diesem Kommentar erkennen.

Das falsche Labeln steht in diesem Fall im Einklang mit der übergreifenden Problematik der Sentiment Analysis in Bezug auf implizite Bedeutung und Sarkasmus. Die Schwierigkeit, ein korrektes Label zu vergeben, wird darüber hinaus durch den politischen Charakter des

Diskurses über dieses Thema erschwert. In diesem Zusammenhang stellt Liu fest: „Opinions about products and services are usually easier to analyze. Social and political discussions are much harder due to complex topic and sentiment expressions, sarcasms and ironies“ (Liu 2012, 17).

Ein weiterer wichtiger Punkt ist, dass der Datensatz nach dem Labeln stark verkleinert wurde. So wurden von den ursprünglich 7.116 Kommentaren nur 1.302 mit einem Label versehen, so dass der Großteil der Daten verworfen wurde. Dies war nicht immer auf die Irrelevanz des Kommentars zurückzuführen, sondern vielmehr auf die Tatsache, dass der Kommentar keines der vordefinierten Schlagwörter enthielt, wie die folgenden Beispiele zeigen:

(12) My sister is a nurse in the NHS and she is an antivaxer, I'm not sure doctors and nurses are educated to the same standards anymore.
(13) Skepticism and asking for more proof and information is good. However when answers and data is provided and overwhelming it is called denial of reality. Which is completely different.

Die beiden Beispiele (12) und (13) vermitteln eine positive Einstellung, wurden aber vom Algorithmus als negativ markiert. Bei der Betrachtung der den jeweiligen Kategorien zugeordneten Elementen wird die Unausgewogenheit der gelabelten Daten deutlich. So machen die als negativ gelabelten Kommentare mit insgesamt 1215 Kommentaren die überwiegende Mehrheit der Daten aus. Die positiven Kommentare machen hingegen nur 6,68% der gesamten Daten aus.

Obwohl Maßnahmen zur Begrenzung der Auswirkungen der Unausgewogenheit des Datensatzes ergriffen wurden, wie beispielsweise die Zuweisung eines höheren Gewichts für die Minderheitenklasse, hatte das Bias des Datensatzes dennoch einen Einfluss auf die Ergebnisse der Vorhersagen des maschinellen Lernmodells. Zusätzlich wurde die Leistung des Modells anhand der accuracy-Metrik gemessen (vgl. Jurafsky u. a. 2009, 67). Dieser Bewertungsmaßstab besteht darin, die Gesamtzahl der korrekten Vorhersagen zu addieren und durch die Gesamtzahl der Vorhersagen zu teilen, um zu berechnen, welcher Prozentsatz der Beobachtungen richtig klassifiziert wurde. Die Auswertung ergab einen accuracy-Score von 95,775% für die Kommentare der Gegner der Corona-Impfung und 55,556% für die Befürworter. Da der Datensatz jedoch unausgewogen ist, muss festgehalten werden, dass die accuracy nicht die beste Metrik ist. Eine gute Alternative wäre die Verwendung des F-scores, das die Minderheitenklasse stärker gewichtet (vgl. ebd.).

Außerdem wurde aus pragmatischen Gründen die Anzahl der Epochen, in denen das Modell die Daten durchläuft, auf 7 begrenzt, wodurch die Möglichkeit des Modells, die Daten richtig anzupassen, eingeschränkt wird. Durch eine Erhöhung dieses Hyperparameters hätten die Ergebnisse wahrscheinlich verbessert werden können.

7. Ausblick und Fazit

Der in der vorangegangenen Analyse skizzierte Prozess umfasst mögliche Ansätze zur Datenerhebung, zum Datenlabeling, zur Datenvorverarbeitung und zum Training eines einfachen neuronalen Netzes.

Alle diese Schritte haben zu unterschiedlich erfolgreichen Ergebnissen geführt, die einerseits den Weg für eine weitere Erforschung des Themas ebnen können, andererseits aber auch die Notwendigkeit von Optimierungen in verschiedenen Aspekten des Projekts aufzeigen.

So könnte beispielsweise die Anzahl der extrahierten Kommentare erhöht werden, um einen größeren Datensatz zu erhalten. Noch wichtiger wäre, die Methode zum Labeling der Daten zu verbessern, um zu vermeiden, dass potenziell nützliche Daten verworfen werden.

Generell zeigen die Ergebnisse des Labelings, wie schwierig und ungenau es ist, die Polarität eines ganzen Satzes nur auf der Grundlage seines Lexikons zu bestimmen. In diesem Fall hätte die unbequemere und zeitaufwändigere manuelle Annotation der Daten bessere Ergebnisse gebracht.

Ein weiterer Aspekt, der verbessert werden könnte, ist die quantitative Beziehung zwischen den beiden Klassen im Datensatz. Da die Daten einen Bias enthielten, war die Ausgabe des neuronalen Netzes vorhersehbar verzerrt. In diesem Zusammenhang muss eingeräumt werden, dass die Erstellung eines ausgewogenen Datensatzes dadurch erschwert wurde, Kommentare zu finden, die sich offen positiv zum Thema der Covid-19-Impfung äußerten. Dies kann mit der Quelle zusammenhängen, die zur Extraktion der Daten verwendet wurde, denn es hat sich gezeigt, dass die Inhalte der sozialen Medien zum Thema Impfung überwiegend ungenau oder negativ sind. Auch wenn positive Einstellungen vorhanden sind, führen sie nicht so häufig zu einem Online-Engagement wie Anti-Impf-Narrative und verbreiten sich daher nicht so effektiv. Ein solches Muster war in dem Datensatz offensichtlich.

Schließlich könnten die Hyperparameter des neuronalen Netzes weiter optimiert werden, um bessere Ergebnisse zu erzielen. Diesbezüglich könnten das Hinzufügen von Layern zum Modell,

das Erhöhen der Anzahl der Epochen, das Experimentieren mit BatchSizes und das Ändern der Bewertungskriterien (F-score anstelle von accuracy) mögliche Ansätze sein, um das bestehende Netz zu verbessern.

Zusammenfassend hat die vorangegangene Arbeit ergeben, dass es technisch möglich ist, eine Sentiment Analysis anzuwenden, um die Einstellung zur Covid-19-Impfung auf Grundlage von YouTube-Kommentaren zu erforschen. Allerdings zeigt das vorliegende Projekt auch, dass es eine Herausforderung darstellt, mit realen Daten umzugehen und diese korrekt darzustellen. Abgesehen von der technischen Schwierigkeit, die mit der Verwendung von implizitem Wissen und Sarkasmus in den Kommentaren verbunden ist, führt der Versuch, ein komplexes Thema durch die Brille eines binären Klassifizierungsproblems zu sehen, zwangsläufig zu einer Vereinfachung und damit zu einer ungenauen Darstellung der Meinungen über Covid-19-Impfungen, die sich größtenteils auf einer Skala bewegen und damit nicht immer durchweg positiv oder negativ sind.

Anhang: Programmiercode und Datensätze

Siehe zum Programmiercode die Dateien: data_collection.py und model_training.py

Siehe zu den Datensätzen die Dateien: data.csv und new_data.csv

Literaturverzeichnis

Baldi, Pierre, und Peter J Sadowski. 2013. „Understanding Dropout“. In *Advances in Neural Information Processing Systems*, herausgegeben von C. J. Burges, L. Bottou, M. Welling, Z. Ghahramani, und K. Q. Weinberger, 26:1–9. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2013/file/71f6278d140af599e06ad9bf1ba03cb0-Paper.pdf.

Beckmann, Fabian, und Anna-Lena Schönauer. 2021. „Spaltet Corona die Gesellschaft? Eine empirische Milieuanalyse pandemiebezogener Einstellungen“. In *Gesellschaft unter Spannung*. Digital.

Burki, Talha. 2020. „The online anti-vaccine movement in the age of COVID-19“. *www.thelancet.com/digital-health* 2 (10): 1–2. https://doi.org/10.1016/S2589-7500(20)30227-2.

Camacho-Collados, Jose, und Mohammad Taher Pilehvar. 2017. „On the Role of Text Preprocessing in Neural Network Architectures: An Evaluation Study on Text

Categorization and Sentiment Analysis“, 1–7. https://doi.org/10.48550/ARXIV.1707.01780.

Daube, Carl Heinz. 2022. „Covid-19 – Auswirkungen auf Gesellschaft und Wirtschaft“. IUCF Working Paper 2/2022. Kiel, Hamburg: ZBW - Leibniz Information Centre for Economics. http://hdl.handle.net/10419/249209.

DW News. o. J. „Coronavirus“. https://www.youtube.com/playlist?list=PLT6yxVwBEbi1HanfWlccJ5ag1qN-EwQmH.

———. o. J. „DW News About“. https://www.youtube.com/c/dwnews/about.

———. o. J. „DW News Home“. https://www.youtube.com/c/dwnews.

Frei, Nadine, Robert Schäfer, und Oliver Nachtwey. 2021. „Die Proteste Gegen Die Corona-Maßnahmen: Eine Soziologische Annäherung“. *Forschungsjournal Soziale Bewegungen* 34 (2): 249–58. https://doi.org/10.1515/fjsb-2021-0021.

Fuchs, Christian. 2022. *Verschwörungstheorien in der Pandemie: wie über COVID-19 im Internet kommuniziert wird*. UTB Kommunikationswissenschaften, Sozialwissenschaften 5796. München: UVK Verlag.

Google. o. J. „Google API“. https://github.com/googleapis/google-api-python-client/blob/main/docs/README.md.

Haas, Christina, Pamela Takayoshi, Brandon Carr, Kimberley Hudson, und Ross Pollock. 2011. „Young People’s Everyday Literacies: The Language Features of Instant Messaging“. *Research in the Teaching of English* 45 (4): 378–404.

Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, u. a. 2020. „Array Programming with NumPy“. *Nature* 585 (7825): 357–62. https://doi.org/10.1038/s41586-020-2649-2.

Hochreiter, Sepp, und Jürgen Schmidhuber. 1997. „Long Short-Term Memory“. *Neural Computation* 9 (8): 1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.

Iglesias, Carlos A., und Antonio Moreno. 2019. „Sentiment Analysis for Social Media“. *Applied Sciences* 9 (23): 5037. https://doi.org/10.3390/app9235037.

Jurafsky, Dan, James H. Martin, Peter Norvig, und Stuart J. Russell. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Second Edition, Pearson International Edition. Prentice Hall Series in Artificial Intelligence. Upper Saddle River, NJ: Prentice Hall, Pearson Education International.

Kemp, Simon. 2022. „DIGITAL 2022: GLOBAL OVERVIEW REPORT“. https://datareportal.com/reports/digital-2022-global-overview-report.

Keras. o. J. „Keras API“. https://keras.io/.

———. o. J. „Probabilistic losses“. Documentation. *Keras API reference* (blog). https://keras.io/api/losses/probabilistic_losses/#categoricalcrossentropy-class.

Kingma, Diederik P., und Jimmy Ba. 2014. „Adam: A Method for Stochastic Optimization“, 1–15. https://doi.org/10.48550/ARXIV.1412.6980.

Lauret, Arnaud. 2019. *The design of web APIs*. Shelter Island, NY: Manning Publications Co.

Liu, Bing. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

McKinney, Wes und others. 2010. „Data structures for statistical computing in python“. In *Proceedings of the 9th Python in Science Conference*, 445:51–56. Austin, TX.

Meyer, C., und S. Reiter. 2004. „Impfgegner und Impfskeptiker.“ *Bundesgesundheitsbl - Gesundheitsforsch - Gesundheitsschutz* 47 (Dezember): 1182–88. https://doi.org/10.1007/s00103-004-0953-x.

Mitchell, Ryan. 2018. *Web Scraping with Python: Collecting More Data from the Modern Web*. 2. Aufl. O’Reilly Media, Inc.

Mønsted, Bjarke, und Sune Lehmann. 2022. „Characterizing polarization in online vaccine discourse—A large-scale study“. *PLoS ONE*, Februar, 1–19. https://doi.org/10.1371/journal.pone.0263746.

Nagy, Peter. 2017. *Sentiment-Analysis-NLTK-ML-LSTM*. Python. https://github.com/nagypeterjob/Sentiment-Analysis-NLTK-ML-LSTM/blob/master/lstm.ipynb.

NLTK. o. J. „Documentation“. https://www.nltk.org/api/nltk.corpus.html.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, u. a. 2011. „Scikit-learn: Machine Learning in Python“. *Journal of Machine Learning Research* 12: 2825–30.

Python. o. J. „HyperText Markup Language Support“. Documentation. https://docs.python.org/3/library/html.html.

Sundermeyer, Martin, Ralf Schülter, und Hermann Ney. 2012. „LSTM Neural Networks for Language Modeling“. In . https://www.isca-speech.org/archive_v0/archive_papers/interspeech_2012/i12_0194.pdf.

Tagesschau. 2022. „Tausende auf der Straße - dafür und dagegen“, 18. Januar 2022. https://www.tagesschau.de/inland/corona-bundesweite-proteste-101.html.

Tensorflow. o. J. *tf.keras.preprocessing.text.Tokenizer*. Python. TensorFlow. https://github.com/keras-team/keras-preprocessing/blob/1.1.2/keras_preprocessing/text.py#L141-L487.

Twitter. o. J. „Twitter Counting characters when composing Tweets“. Documentation. https://developer.twitter.com/en/docs/counting-characters.

———. o. J. „Twitter How to create and use hashtags“. https://business.twitter.com/en/blog/how-to-create-and-use-hashtags.html#:~:text=On%20Twitter%2C%20adding%20a%20%E2%80%9C%23,that%20they%27re%20interested%20in.

„WHO announces COVID-19 outbreak a pandemic“. 2020. World Health Organization. https://www.euro.who.int/en/health-topics/health-emergencies/coronavirus-covid-19/news/news/2020/3/who-announces-covid-19-outbreak-a-pandemic.

Wicke, Philipp, und Marianna M. Bolognesi. 2021. „Covid-19 Discourse on Twitter: How the Topics, Sentiments, Subjectivity, and Figurative Frames Changed Over Time“. *Frontiers in Communication* 6: 1–20. https://doi.org/10.3389/fcomm.2021.651997.

Willett, Peter. 2006. „The Porter Stemming Algorithm: Then and Now“. *Program* 40 (3): 219–23. https://doi.org/10.1108/00330330610681295.

YouTube. o. J. „Obtaining authorization credentials“. https://developers.google.com/youtube/registering_an_application.

———. o. J. „Richtlinien zu Spam, irreführenden Praktiken und Betrug“. Documentation. https://support.google.com/youtube/answer/2801973.

———. o. J. „YouTube Data API Overview“. Documentations. https://developers.google.com/youtube/v3/getting-started#intro.