Research and Professional Development in Creative Media

„Entwicklung einer deutschsprachigen Künstlichen Intelligenz zur automatischen News-Analyse und politischen Einordnung für eine Unbiased News Webapplikation“

Modulnummer:	*6FSC1WD102*
Modulname:	*Web Development Major Project (BSc)*
Abgabedatum:	*23.08.2024*
Abschluss:	*Bachelor of science (Hons.) Webdesign und Development*
Semester:	*August 2024*
Name:	*Daniel Hilmer*
Campus:	*München*
Land:	*Deutschland*
Wortanzahl:	*9.907*

Selbstständigkeitserklärung:
Hiermit bestätige ich, dass ich die vorliegende Arbeit eigenständig erarbeitet habe und alle Hilfsmittel sowie Inhalte von Dritten vollständig angegeben wurden. Hierunter fällt unter anderem die korrekte Belegarbeit für die direkte oder sinngemäße Übernahme von Texten, Bild-, Audio- und Videomaterial sowie Code etc. Weiterhin die klare Kennzeichnung von Inhalten, die von anderen Personen oder technischen Hilfsmitteln wie einer künstlichen Intelligenz erstellt wurden.

München, 21.08.2024	Daniel Hilmer
Ort, Datum	Unterschrift Student/in

Rechtevereinbarung:
Hiermit räume ich, dem SAE Institut das nicht exklusive, jedoch zeitlich und örtlich unbeschränkte Recht ein, die vorliegende Arbeit zum Zweck der Ausbildung, sowie der Darstellung von Ausbildungsinhalten, zu speichern und für Personen des SAE Instituts zugänglich zu machen.

München, 21.08.2024	Daniel Hilmer
Ort, Datum	Unterschrift Student/in

Abstract

Ziel dieser Arbeit ist die Entwicklung einer automatisierten Einordnung von Nachrichtenartikeln in politische Gruppen und die Darstellung der Ergebnisse mittels einer Webanwendung. Für die Analyse und politische Einordnung der Nachrichten wurde eine künstliche Intelligenz trainiert.

Zu diesem Zweck wurde eine Reihe von Arbeitsschritten durchgeführt. Darunter fallen unter anderem das Sammeln der Daten, deren Aufbereitung und Labeling, das Training eines KI-Modells und die Veröffentlichung einer Inference API (Schnittstelle zur Abfrage des Modells). All dies wurde mit minimalem finanziellem Aufwand durchgeführt.

Das Ergebnis dieser Experimente zeigte, dass eine entscheidende Stellschraube beim Training von künstlichen Intelligenzen die Datenbasis ist. So konnte das Training mit manuell gekennzeichneten Daten (gruppierte Labels, z. B. alle Nachrichten der Tagesschau sind POLITISCH MITTEL) doppelt so gute Ergebnisse liefern wie das Training mit KI-gekennzeichneten Daten, bei denen jeder Artikel einzeln von einer Labeling-KI analysiert und beschriftet wurde.

Das zeigt, dass es prinzipiell möglich ist, ein KI-Modell zu trainieren und für die Einordnung von Nachrichtenartikeln in politische Gruppierungen zu verwenden. Zu beachten sind dabei mögliche Verzerrungen, mit denen das Modell trainiert wird. Hier empfiehlt sich eine möglichst vielfältige Datenbasis mit entsprechenden Kontrollinstanzen, um den Bias zu minimieren.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Disclaimer

Im Rahmen der Erstellung dieser Bachelorarbeit wurde künstliche Intelligenz (KI) zur Unterstützung bei der Formulierung und Überarbeitung von Texten eingesetzt. Dies umfasst insbesondere die Verbesserung von Satzbau, Satzstellung und Rechtschreibung. Der inhaltliche Kern sowie die wissenschaftlichen Analysen, Schlussfolgerungen und alle themenbezogenen Aussagen wurden jedoch vollständig von der Autorin/ dem Autor selbstständig erarbeitet. Die KI diente ausschließlich als Werkzeug zur sprachlichen Optimierung und nicht zur inhaltlichen Gestaltung der Arbeit.

1 Einleitung

1.1 Hinführung zum Thema

Durch Social Media, Google News und andere Webseiten erhalte ich oft nur ausgewählte Informationen und Nachrichten. Dies führt dazu, dass ich und viele andere Personen Ihre eigene Meinung nur anhand der bereitgestellten Informationen bilden können. Diese Informationen sind meistens zugeschnitten auf persönliche Werte, Interessen und Vorlieben. Allerdings führt es dazu, dass andere Meinungen oder Informationen seltener eine Rolle spielen. Dies bei trivialen Dingen wie lustigen Videos durchaus in Ordnung. Bei Nachrichten und wichtigen Informationen, finde ich, sollten alle Menschen verschiedene Blickwinkel präsentiert bekommen. Denn nur so lässt sich eine fundierte Meinung bilden.

Unter anderem deshalb halte ich es für sinnvoll, eine künstliche Intelligenz (KI, engl. AI) zu entwickeln, das die politische Ansicht in einem deutschsprachigen Artikel ermittelt.

Mit dieser Technologie kann eine Plattform entwickelt werden, die es den Nutzenden ermöglicht, Nachrichten aus verschiedenen politischen Blickwinkeln zu betrachten, um ein ausgewogeneres Bild zu erhalten.

1.2 Kreative Idee

Die Idee meines Projekts besteht darin, eine künstliche Intelligenz zu entwickeln, die deutsche Nachrichten anhand der Sprache auf die politische Herkunft zu analysieren. Die KI und die daraus resultierende Analyse bieten die Möglichkeit für deutschsprachige Webseiten, Web-Apps oder Apps, mit dem Fokus auf die Bereitstellung von Nachrichten, eine Analyse dieser durchzuführen und die politische Ausrichtung auf der Plattform darzustellen.

Ein großer Fokus soll darin bestehen, eine künstliche Intelligenz bereitzustellen, die Nachrichten analysiert und sie in politische Gruppierungen einordnet. Eine zugehörige Web-App zu entwickeln, ist sekundär.

1.3 Zielsetzung

Tabelle 1: Zielsetzung und Priorisierung

Ziele	**Muss**	**Soll**	**Kann**
Inhaltliche Ziele			
News-Datenmanagement Automatischer Erhalt von neuen News, Speicherung der News.	X		

Labeling Labeling der Nachrichtenartikel, damit die KI mit Nachrichten und Labeln (POLITISCH LINKS – POLITISCH LEICHT LINKS – POLITISCH MITTEL – POLITISCH LEICHT RECHTS – POLITISCH RECHTS) trainiert werden kann.	X		
Fake News Detektion Scannen der Nachrichten auf Fake News, bevor Sie für das Training verwendet werden, Scannen der Nachricht auf Fake News und Tagging/ Kennzeichnung auf der Web-App.		X	
Politische Einordnung und Analyse (Inference) Analyse der News und politische Einordnung anhand der verwendeten Sprache.	X		
Entwicklung der Web-Anwendung Webanwendung zur Bereitstellung der gelabelten Nachrichten.	X		
Zusammenfassungen der News KI generierte Zusammenfassung aus News Artikeln.			X
Qualitative Ziele			
Design und Layout nach Vorbild von Google News		X	
Optimierung primär auf mobile Geräte		X	
Politische Einordnung nach Vorbild von Ground News			X
Zusammenfassung der Artikel nach Vorbild von Ground News			X
Mindestens 75 % accuracy = < 25 % loss	X		
Quantitative Ziele			
Deutsche News	X		
Training/ Finetuning der KI mit ~ 5.000 – 50.000 Datensätzen	X		
Nicht Ziele			
Die Web-App ist ein Prototyp. Deshalb wird diese nicht öffentlich zugänglich gemacht. Personen für Evaluationen, Bewertung, etc. bekommen einen Zugang.			
Die App wird nicht die Komplexität von (Google News, 2023; Ground News, 2023; The Factual, 2023) annehmen. Sie wird lediglich eine einfache Benutzeroberfläche bieten.			

1.4 Verortung in der Industrie

Spezialisierte KIs, die den Service anbieten, deutsche Nachrichtenartikel auf die politische Herkunft zu analysieren, konnte ich in meiner Recherche nicht finden. Allerdings gibt es einige Anbieter (mit Inhalten auf Englisch), die ihre Web-Apps mit ähnlichen Fähigkeiten auf dem

Markt anbieten. Deshalb habe ich mich dazu entschlossen, die Positionierung in der Industrie als Web-App mit KI-Anbindung anzustreben.

Ground News (Ground News, 2023), eine bekannte Unbiased News Seiten aus den USA, hatte im Dezember 2023 laut (Similarweb, 2024a) 6,6 Millionen Aufrufe. Die App von Ground News hat im Google Play Store über 500.000 Downloads. Ich konnte nicht herausfinden, wie viele monatliche Benutzer das Unternehmen hat. Die Unbiased News Seiten AllSides (AllSides, 2019) hatte im Dezember 1,3 Millionen Aufrufe (Similarweb, 2024b). Auch diese ist v.a. in den USA vertreten. Im Vergleich dazu: Google News (news.google.com) hatte 381 Mio. Aufrufe und Yahoo News (news.yahoo.com) 240 Mio. Aufrufe.

Darüber hinaus gibt es weitere Faktoren, die für eine Positionierung dieser Webanwendung in der Branche sprechen. So ist zu beobachten, dass immer mehr Menschen vom offline- zum online-Nachrichtenkonsum übergehen (Abb. 1: Sources of news in Germany (Hölig, 2023)). Zudem nimmt das Vertrauen der Bevölkerung in die Nachrichtendienste immer weiter ab (Hölig, 2023; Newman u. a., 2023) (Abb. 2: Overall trust score (Hölig, 2023)). Gerade hier könnte eine Seite, die die politische Herkunft von Nachrichten analysiert, helfen, Vertrauen zu schaffen.

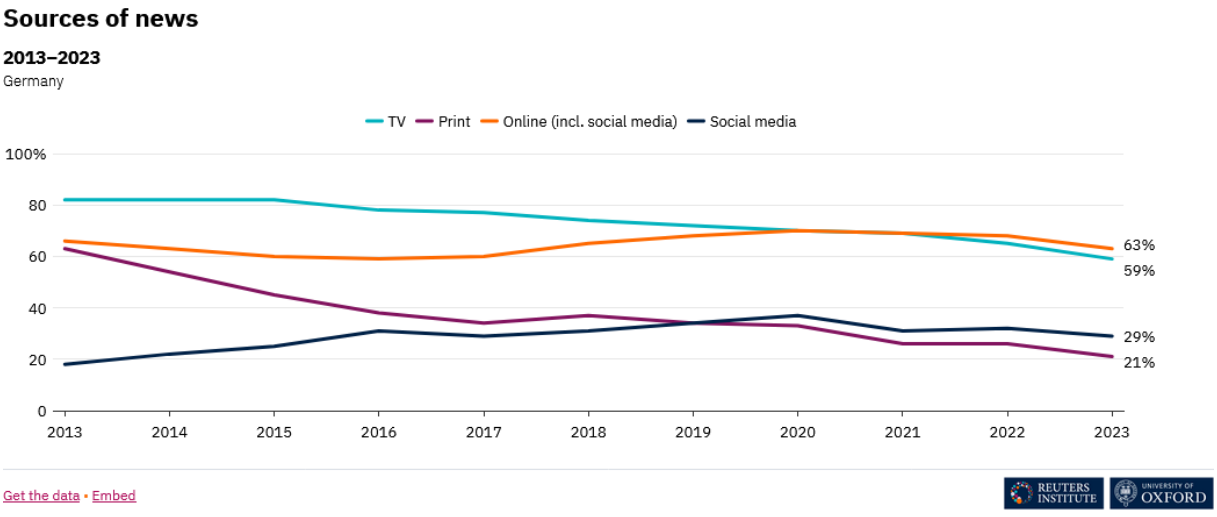


Abb. 1: Sources of news in Germany (Hölig, 2023)

Ein Problem, das sich abzeichnet, ist, dass Menschen nur selten (gerade einmal 11 %) für Online-Nachrichten bezahlen (Hölig, 2023; Newman u. a., 2023). Dies spielt zwar für den Prototyp keine Rolle, wäre aber für die Vermarktung eines solchen Projekts relevant.

Aus diesen Gründen stufe ich mein Projekt als kommerzielles Nischenprodukt ein.

1.5 Zielgruppe

Die Zielgruppen für dieses Projekt sind sehr breit aufgestellt. Die Zielgruppen sind folgende:

1. **Personen,** die sich in einer Medienblase gefangen fühlen und Nachrichten aus verschiedenen politischen Spektren lesen möchten.
2. **Redaktionsmitarbeitende** könnten die Anwendung verwenden, um schneller unterschiedliche Perspektiven im politischen Spektrum einsehen zu können.
3. **Bildungseinrichtungen** könnten die Anwendung für Lehr- und Forschungszwecken nutzen, um den Lernenden und Forschenden eine effiziente Möglichkeit zur Analyse und Einordnung von Nachrichten zu bieten.
4. **Entwickelnde,** die eine Anwendung erstellen, in der die Informationen aus der KI ein wichtiger Bestandteil sind (z. B. eine Unbiased News Webseite). *Für diese Zielgruppe ist die API für die KI relevant.*
5. Die schriftliche Arbeit ist außerdem für **Medienforschende** mit dem Fokus auf Nachrichten und künstliche Intelligenzen und **Entwickelnde**, die mit der API arbeiten möchten, relevant.

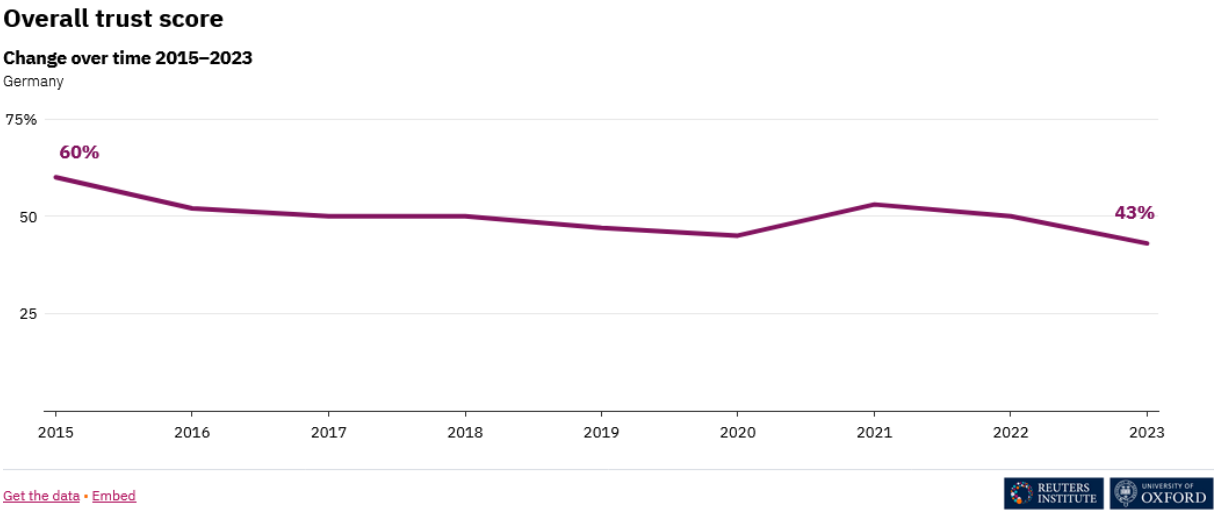


Abb. 2: Overall trust score (Hölig, 2023)

2 Kontext

2.1 Bias in künstlichen Intelligenzen

2.1.1 Was ist ein Bias?

Das Wort „Bias“, zu Deutsch „Vorurteil“/ „Voreingenommenheit“ beschreibt in Bezug auf den Menschen eine verzerrte Wahrnehmung (OECD, 2020, S.17). „That we as humans do make errors in thinking, judgment, and memory is undisputed.“ (Pohl, 2022, S.16).

2.1.2 Bias in KIs

Durch den natürlichen Bias des Menschen sind automatisch auch KIs betroffen, da sie von Menschen entwickelt werden (TED und Sharma, 2019; Prete u. a., 2022, S.17; Varsha, 2023, S.1). Die FRA (European Union Agency for Fundamental Rights) beschreibt das Problem wie folgt: „the bias itself and the resulting discrimination is pervasive in society, rooted in psychological, social and cultural dynamics, and hence reflected in the data and texts that are used for developing AI models.“ (Prete u. a., 2022, S.17). In einem Beitrag der Tagesschau wird über einen Fall berichtet, in dem eine KI-Gesichtserkennungssoftware das Geschlecht von weiblichen und farbigen Personen deutlich schlechter erkennt als das von weißen und männlichen Personen (Beck, 2023, Abs.2–3). Dies liegt daran, dass „überdurchschnittlich viele weiße Männer, die künstliche Intelligenzen entwickeln. Der Frauenanteil in der deutschen IT-Branche lag 2021 bei gerade mal 18 Prozent.“ (Beck, 2023, Abs.4).

Es gibt unzählige Beispiele wie dieses. Vom Gesundheitswesen über Online-Werbung bis hin zur Bilderzeugung (Varsha, 2023, S.3ff.). Denn je nach Datenbasis, Algorithmus und Training kann die KI historische oder heutige Vorurteile übernehmen. Das hat zur Folge, dass die KI z. B. unabsichtlich rassistisch oder sexistisch trainiert werden kann (CrashCourse, 2019; Park, 2021; Prete u. a., 2022; IBM Data und AI Team, 2023; TED und Luccioni, 2023).

2.1.3 Wie Bias in KI reduziert werden können

Es gibt einige Möglichkeiten und Wege, den Bias von KI zu reduzieren. Die wichtigsten sind unter anderem:

1. **Vielfältige und repräsentative Datensätze**
   Die Trainingsdaten sind ein entscheidender Faktor für den Bias von KI. Um künstliche Intelligenzen fair trainieren zu können, werden möglichst viele verschiedene Daten benötigt. Ein Beispiel dafür ist der zuvor beschriebene Tagesschau-Beitrag. Um die KI fair zu trainieren, muss sie mit etlichen unterschiedlichen Nationalitäten, Geschlechtern

und weiteren Merkmalen gleichermaßen trainiert werden. Nur so kann sie im Anschluss Personen mit unterschiedlichen Gesichtsmerkmalen erkennen (Beck, 2023).

2. **Regelmäßige Prüfung und Validierung der KI**
   Feedbackschleifen sind ein wichtiges Instrument, um die Ausgaben einer KI zu bewerten. Beispielsweise können die Antworten der KI auf ChatGPT (OpenAI, 2023) positiv oder negativ bewertet werden. Diese Bewertungen werden von Personen überprüft und neuen ggf. angepasste Versionen können zum KI-Training verwendet werden.
3. **Ausbildung und Bewusstsein der Entwickelnden**
   Viele Entwickler sind weiß und männlich. Dadurch werden beispielsweise andere Personengruppen übersehen oder Tests fallen fälschlicherweise positiv aus. Um das Bewusstsein dieser Problematik zu verstärken, müssen Entwicklerteams diverser eingestellt und geschult werden (Beck, 2023, Abs.5ff.).
4. **Ethikrichtlinien**
   In dem Artikel „How can we manage biases in artificial intelligence systems – A systematic literature review“ (Varsha, 2023) werden folgende wichtige Ethikrichtlinien genannt, die für verantwortungsvolle KI-Systeme wichtig sind (Tabelle 2):

Tabelle 2: Responsible AI factors (Varsha, 2023, S.6)

Factors	**Explanation**	**Measures are taken to address AI Bias**
Fairness	AI devices validate the diversity inclusion and reduce the biases	IBM has taken the initiative to minimize bias by providing the open-source toolkit to evaluate, create a report and alleviate discrimination and bias through machine learning models (IBM AI Fairness, 360)
Transparency	AI systems must be more transparent with respect to processes and results in the firms	Accenture practices the responsible AI has a transparency in four pillars organizational, operational, technical and reputational to create company values and ethics. By understanding the sources of bias and developing the Accenture Algorithmic toolkit used to investigate errors and bring the fairness and transparency decisions by creating a new model (Accenture, 2021)

Accountability	AI systems must bring the accountability of their results with ethics	In Microsoft, people are accountable for the AI system's impact on the world due to a variety of models, data sets and new technology disruption. The company follows principles and guidelines to understand the customers through facial recognition and understanding and monitoring the errors in each stage to minimize the bias in the AI life cycle (Microsoft, 2022)
Robustness & Safety	AI systems should be created with precautionary measures taken to reduce the errors	Google is in adversarial learning while using the neural network by creating adversarial illustrations to fool a system in the network to detect the frauds (Google)
Privacy & Governance	Personal data is used to make decisions and privacy controls must be created to support technology ensure that personal data will be used for specific and fair purposes. AI Systems must follow the regulations related to international data privacy laws and standards	Cisco developed a privacy engineering practices into the Cisco Secure Development Lifecycle (CSDL). These practices help to assures that data privacy in the service offerings. Further, the company sets and follows the principles of the Global Personal Data Protection and Privacy Policy (Cisco, 2022).
Societal & Environmental Wellbeing	AI Systems bring the ethical and equitable AI as a comprehensive approach to society and environment well being	Intel is designed the AI lifecycle to reduce the risks by bringing the ethical principles and to maximize the benefits the society by using the right tools and enabling an inclusive and sustainable environment (Intel)

Festzustellen ist, dass es nicht einen richtigen Weg gibt, um den Bias in KI zu reduzieren. Nur eine Reihe von richtigen Entscheidungen von der Datenbeschaffung, Programmierung bis hin zu den Kontrollinstanzen kann dazu führen, dass KI in Zukunft ethisch korrektere Entscheidungen trifft (OECD, 2020, S.147 ff.).

2.1.4 KI als Hilfe entgegen dem menschlichen Bias?

Allerdings gibt es auch Behauptungen und Programme, die dafürsprechen, dass KI den Bias des Menschen positiv beeinflussen kann. Mit der richtigen Entwicklung von Algorithmen und ethischen Richtlinien kann Künstliche Intelligenz den Menschen bei den eigenen Vorurteilen helfen und sie darauf aufmerksam machen (Fawn Fitter und Steven T. Hunt, PhD, 2023, Abschn.3). So gibt es Tools, die versuchen, Arbeitsbeschreibungen zu analysieren, um eine voreingenommene Sprache zu erkennen oder KI, die Erfahrungen und Fähigkeiten von Bewerbern hervorhebt und Herkunft, Geschlecht und andere Bias ausschließt (Gow, 2024, Abschn.4). Auch im Marketing und in der Medizin finden sich bereits Programme, die versuchen, die menschliche Voreingenommenheit zu reduzieren.

2.1.5 Fazit und eigene Meinung

Der Bias von KI kann nicht verhindert werden. Der Versuch, diesen zu reduzieren, ist jedoch essenziell, um diese Technologie in Zukunft sicher verwenden zu können. Dafür müssen v.a. Entwickelnde geschult und Datensätze möglichst divers sein. Darüber hinaus müssen KI-Modelle ethisch entwickelt und transparent gegenüber den Benutzern dargestellt werden. Denn dann können wir es schaffen, dass Gefahren durch Bias in der Gesellschaft nicht verstärkt, sondern durch KI reduziert werden.

2.2 KI-Training und Fine-Tuning

Moderne künstliche Intelligenzen wie Large Language Models (LLMs) basieren auf der Transformer-Technologie, wie sie im Google Paper „Attention Is All You Need“ (Vaswani u. a., 2017) beschrieben wird. Die Aufgabe der Transformer ist es, in unstrukturierten Daten, wie Sprache, ein Muster zu finden und anhand dessen Wahrscheinlichkeiten für eine Ausgabe zu berechnen (siehe Anhang 1.1). Dafür führt das Modell einige mathematische Berechnungen aus, um die Muster in den sogenannten Weights und Biases zu speichern. Die Weights und Biases sind künstliche Neuronen, die an Neuronen im menschlichen Gehirn erinnern (Abb. 3 Neuronen und Schichten im Transformer Modell (Dontsov, 2021)). So gibt es in einem Transformer-Modell eine Eingabeschicht (grün), die für die Aufnahme der Daten verantwortlich ist und eine Ausgabeschicht (rot), die die berechneten Wahrscheinlichkeiten ausgibt. Zwischen den Eingabe- und Ausgabe-Schichten befindet sich eine Anzahl *n* von verborgenen Schichten, die die Wahrscheinlichkeitsberechnungen durchführen. Die in Abb. 3 dargestellten Punkte sind die Neuronen, die die Weights und Biases speichern.

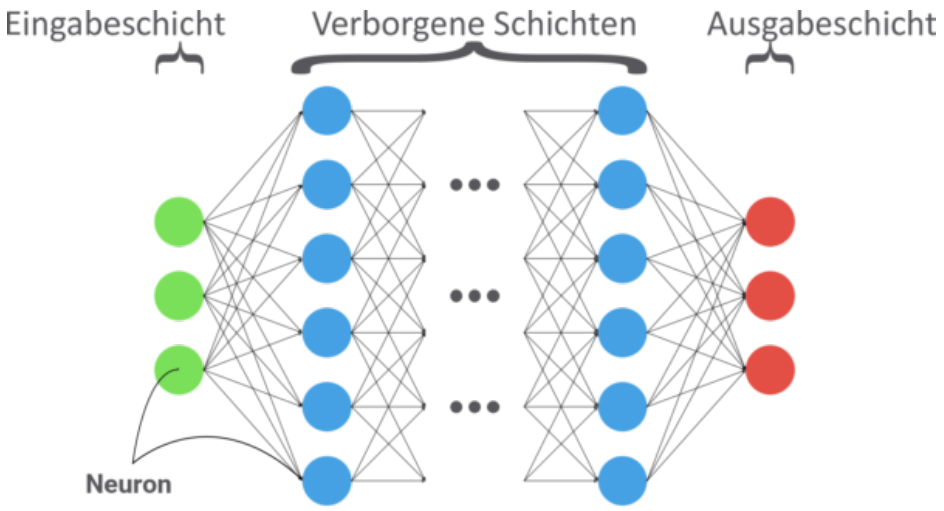


Abb. 3 Neuronen und Schichten im Transformer Modell (Dontsov, 2021)

Das Transformer-Modell (Abb. 4) ist eine Encoder-Decoder-Architektur. Damit das Modell natürliche Sprache verstehen kann, wird Text in numerische Werte umgewandelt (Encoder). Anschließend durchläuft das Modell einige Schichten und baut das Wissen in den Weights und Biases auf. Diese Vorgehensweise hat sich seit der Veröffentlichung des Papers kaum geändert. Auch heute funktionieren Transformer-Modelle ähnlich wie auf Abb. 4 zu sehen (Karpathy, 2024). Zusätzlich dazu gibt es Weiterentwicklungen wie den reinen Encoder (Abb. 4 links) oder Decoder (Abb. 4 rechts). Diese Abwandlungen wie die GPT (Generative Pre-Training) Modelle (Decoder) oder BERT (Bidirectional Encoder Representations from Transformers) Modelle (Encoder) können bestimmte Vor- und Nachteile haben. Encoder-only-Modelle sind z. B. für Klassifizierungsaufgaben hilfreich, wohingegen Decoder-only-Modelle vor allem für generative Ausgaben verwendet werden. Encoder-Decoder-Modelle werden beispielsweise für Übersetzungen und Textzusammenfassungen eingesetzt (Raschka, 2023).

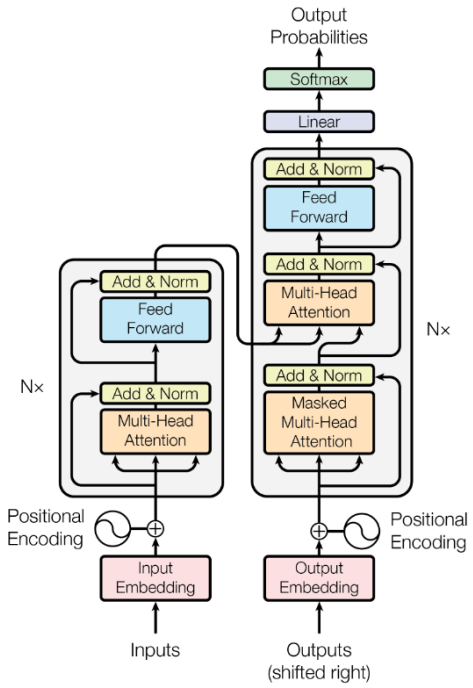


Abb. 4: Die Transformer Technologie (Vaswani u. a., 2017)

2.2.1 KI-Modelle – Vortrainiert oder nicht

Es existieren verschiedene Möglichkeiten, Natural Language Processing (NLP) Aufgaben umzusetzen. Die Erste ist, das zugrundeliegende LLM selbst zu „erstellen“. Sobald der Trainingsalgorithmus programmiert wurde, kann das KI-Modell auf großen Datenmengen trainiert werden, um allgemeines „Wissen“ (Weights & Biasas) aufzubauen. Hierfür werden oftmals große Teile des Internets, z. B. Wikipedia und andere Webseiteninhalte, verwendet (Vaswani u. a., 2017; Bourke, 2022; freeCodeCamp, 2023; Anil u. a., 2024). Aufgrund der extremen Datenmengen ist dieser Trainingsschritt sehr aufwendig, kosten- und zeitintensiv. Vor allem Großunternehmen wie OpenAI, Facebook und Google können sich dieses initiale Training leisten. Daher wird dieser Schritt häufig übersprungen, um schneller an ein Ergebnis zu kommen (Nachum, 2024; Poole, 2024). Das funktioniert, da Open-Source-Modelle auf Plattformen wie Hugging Face (Hugging Face, 2023) bereitgestellt werden, die bereits mit allgemeinen Informationen trainiert wurden (Pre-Training). Diese Modelle kommen oft von großen Unternehmen. Teilweise gibt es aber auch Community-Projekte und Start-ups wie Mistral AI (Mistral AI, 2024), die öffentliche Modelle anbieten.

2.2.2 Wahl des richtigen Basis Modells

Es gibt eine sehr große Auswahl von Transformer-Modellen. Dabei unterscheidet sich nicht nur die Technologie (Encoder/ Decoder), sondern auch die Art des Trainings, die Trainingsdaten und die Größe des Modells. Des Weiteren kann zwischen Closed und Open-

Source Varianten unterscheiden, welche Vor- und Nachteilen mit sich bringen (botpress, 2024).

Kostenpflichtige Modelle wie GPT-4 (OpenAI, 2023) oder Gemini (Google Ireland Limited, 2024) gehören zu den besten und in der Regel vielseitigsten KI-Modellen. Die Verwendung dieser Modelle ist ausschließlich über die Schnittstellen der jeweiligen Unternehmen möglich und ist teilweise kostenpflichtig. Sie können exklusiv auf den Servern der Großkonzerne verwendet werden und sind deshalb nur schwer mit hohen Datenschutzansprüchen vereinbar (botpress, 2024). Individuelle Ergebnisse für spezifische Inhalte (own Domain) lassen sich bei diesen Modellen nicht durch sogenanntes Finetuning (weiteres Training der Modelle mit individuellen Daten) realisieren, sondern über Technologien wie RAG (Retrieval Augmented Generation). Bei RAG werden dem LLM zusätzliche Informationen in Form von Datenbanken und Dokumenten bereitgestellt. Das Closed-Source-Modell kann über eine Schnittstelle auf diese Daten zugreifen und Informationen extrahieren. Das kann zum Beispiel für interne Firmendaten relevant sein, auf denen das Modell nicht trainiert wurde (AWS, 2024b).

Im Gegensatz zu kostenpflichtigen Varianten der LLMs gibt es zahlreiche Open-Source-Lösungen (OS) auf dem Markt. Eine Plattform, die Modelle, aber auch Datensätze und Lernmöglichkeiten für KI anbietet, ist Hugging Face. OS-Lösungen bieten den Vorteil, dass Sie auf eigener Hardware verwendet und fine-tuned werden können. Das ermöglicht, dass LLMs auf die eigenen Wünsche und Vorstellungen individualisiert werden können. Diese Open-Source-Modelle können in unterschiedlichen Arten vorkommen. Einige sind bereits auf eine bestimmte Art und Weise fine-tuned, andere nur mit allgemeinen Informationen trainiert (Tabelle 3). Darüber hinaus können Modelle auch für unterschiedlichen spezifischen Aufgaben (z. B. Text-zu-Text-Generierung, Textklassifikation, Audio-zu-Text und viele weitere) und Sprachen optimiert sein (Hugging Face, 2023).

Tabelle 3: Vergleich von kommerziellen und Open Source LLMs

	Kommerzielle LLMs (GPT-4, Gemini, etc.)	**Open Source LLMs (Llama, Mistral, GPT2, etc.)**
Die besten LLMs auf dem Markt	✓	×/✓ (Zu Teilen, generell eher nicht)
Schnell und einfach zu verwenden	✓	×
Individualisierung (RAG, Finetuning)	✓/× (Nur mit RAG, wenn der Hersteller dies zulässt)	✓
Kostenlos	×	✓
Eigene Hardware	×	✓

Um das richtige Modell für die Anforderungen auswählen zu können, müssen die Anforderungen an das Modell berücksichtigt werden. Für das Modell Fine-Tuning muss ein Open-Source-Modell ausgewählt werden. Da diese vielfach zur Auswahl stehen, kann auf Plattformen wie Hugging Face vorgefiltert werden, für welche Sprache, Aufgabe oder Datensätze das Modell optimiert sein soll. Darüber hinaus finden sich viele Inhalte zu Benchmarks, die unterschiedliche Modelle miteinander vergleichen (Hugging Face, 2024).

Bei der Auswahl und Training von Modellen werden diese üblicherweise auf folgende Metriken geprüft:

- **Genauigkeit (Accuracy)** misst den Anteil der insgesamt korrekt vorhergesagten Ergebnisse (sowohl wahre positive als auch wahre negative) im Verhältnis zur Gesamtzahl der Fälle.
- **Präzision (Precision)** misst das Verhältnis von korrekt positiven Vorhersagen zur Gesamtzahl der als positiv klassifizierten Fälle. Diese Metrik ist wichtig, wenn die Kosten einer falsch positiven Vorhersage hoch sind.
- **Recall (Sensitivität)** misst das Verhältnis von korrekt positiven Vorhersagen zur Gesamtzahl der tatsächlich positiven Fälle. Diese Metrik ist kritisch, wenn es wichtig ist, alle positiven Fälle zu identifizieren.
- **Vorhersagezeit (Prediction Time)**: Diese Metrik bewertet, wie schnell ein Modell eine Vorhersage trifft. Dies ist besonders wichtig für Anwendungen, die Echtzeit- oder nahezu Echtzeitreaktionen erfordern.
- **F1-Score**: Der F1-Score ist der Mittelwert aus Präzision und Recall. Er wird häufig verwendet, um die Genauigkeit eines Modells in unbalancierten Datensätzen zu bewerten. Der F1-Score ist besonders nützlich, um zu messen, wie gut ein Modell sowohl relevante Elemente findet (Recall) als auch relevante Elemente korrekt identifiziert (Präzision).

2.2.3 Fine-tuning von KI-Modellen

Um ein KI-Modell zu fine-tunen, wird ein Basismodell verwendet und auf einem individuellen Datensatz trainiert. Dadurch kann das Modell spezifischere Probleme oder Aufgaben besser lösen. Das Fine-Tuning kann auf unterschiedliche Arten durchgeführt werden.

2.2.3.1 Konventionelles Fine-Tuning

Bei dem konventionellen Fine-Tuning (Abb. 5) werden alle Parameter eines Basismodells angepasst. Dadurch lernt das Modell die neuen Inhalte. Dabei können jedoch Probleme auftreten. So kann das Training bei großen Modellen sehr rechenintensiv sein, da in jedem Durchlauf viele Parameter im Modell angepasst werden müssen. Außerdem kann bei kleinen Modellen, die mit komplexen Daten trainiert werden, das „katastrophale Vergessen“

(catastrophic forgetting) eintreten. In der Arbeit „An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks“ beschreiben die Autoren das Problem wie folgt: „When trained on one task, then trained on a second task, many machine learning models “forget” how to perform the first task.“ (Goodfellow u. a., 2015, S.1). Dies liegt daran, dass die Parameter so stark angepasst werden, dass das alte „Wissen“ überschrieben wird.

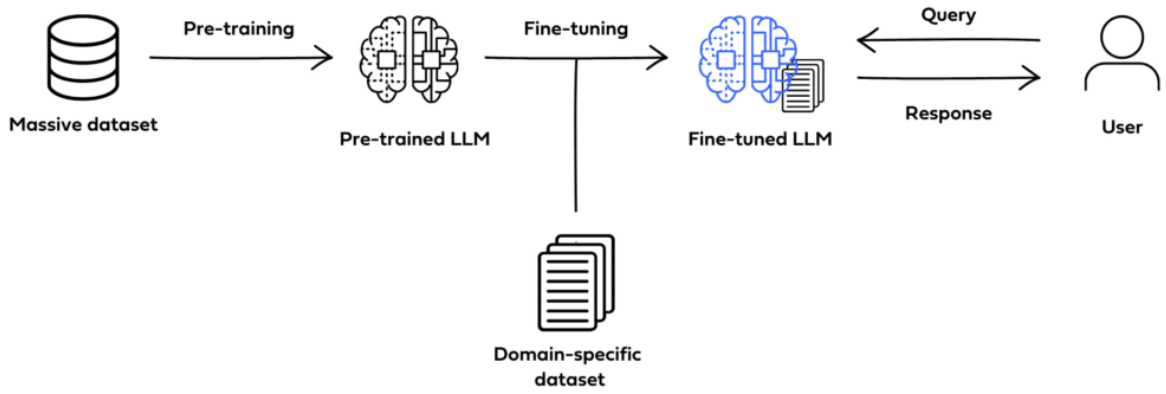


Abb. 5: Full Finetuning (Nabwani, 2023)

2.2.3.2 Probleme beim Fine-Tuning

Neben Problemen wie dem „katastrophalen Vergessen“, das vor allem bei kleineren Modellen auftritt, können auch weitere Probleme wie das Over- und Underfitting auftreten (Abb. 7).

Beim Underfitting (Abb. 7) erkennt das Modell die Merkmale der Daten nicht und ordnet sie den falschen Labels zu. Um dieses Problem zu lösen, können die Modelle z. B. mit mehr Daten über einen längeren Zeitraum werden (IBM, 2021).

Overfitting (Abb. 7) äußert sich durch eine zu granulare Einordnung der Daten. Dadurch lernt das Modell sehr gut die Daten, die für das Training verwendet werden (auswendig lernen), kann das Wissen aber nicht auf neue Daten anwenden. Deshalb ist der Loss während des Trainings (train/loss) sehr gut, wohingegen der Loss der Evaluierung (eval/loss) immer schlechter wird (Abb. 6) (Ying, 2019; Charles, 2021; IBM, 2021). Der Loss beschreibt die Fehlerrate eines Modells.

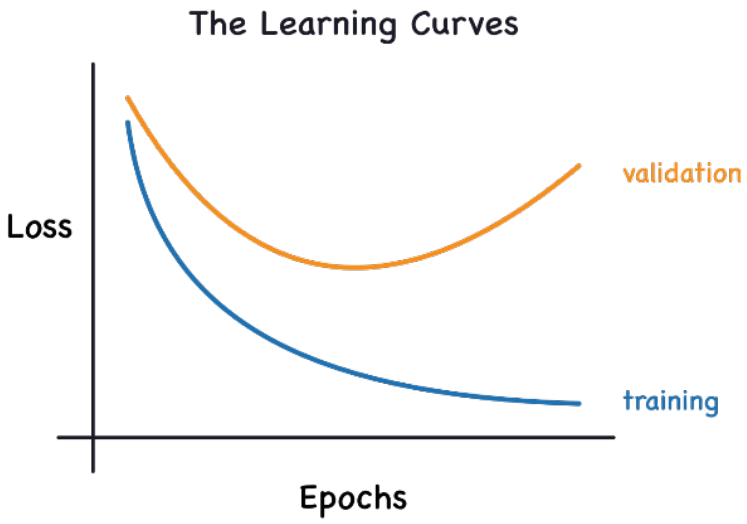


Abb. 6: Overfitting loss curve (Charles, 2021)

Um das Problem des Overfitting eines Modells zu lösen, können laut Studien (Ying, 2019) folgende Maßnahmen getroffen werden:

1. **Early-stopping** wird eingesetzt, um den genauen Zeitpunkt zu finden, an dem das Training gestoppt werden soll. Auf diese Weise wird eine perfekte Balance zwischen Underfitting und Overfitting erreicht.
2. **Network-Reduction** wird verwendet, um „noise“ zu vermeiden. Rauschen entsteht durch irrelevante Informationen in den Daten. Diese machen das Training schwieriger. Um das Rauschen zu verhindern, wird das sogenannte „Pruning“ eingesetzt. Davon gibt es zwei verschiedene Arten.

   „**Pre-pruning** algorithms function during the learning process. Commonly, they use stopping criteria to determine when to stop adding conditions to a rule, or adding rule to a model description, such as encoding length restriction based on the evaluation of encoding cost; significance testing is based on significant differences between the distribution of positive and negative examples; cutoff stopping criterion based on a predefined threshold.“ (Ying, 2019, S.4)

   „**Post-pruning** splits the training set into two subsets: growing set and pruning set. Compared to pre-learning, post-pruning algorithms ignore overfitting problems during the learning process on growing set. Instead, they prevent overfitting through deleting conditions and rules from the model generated during learning. This approach is much more accurate, and however, less efficient.“ (Ying, 2019, S.4)
3. **Erweiterung der Trainingsdaten**, kann dabei helfen, dass das Modell komplexe Zusammenhänge verstehen kann (Ying, 2019, S.3 f.).
4. **Regularisierung** kann dabei helfen, dass das Modell nur wichtige „Features“ in die Berechnungen aufnimmt.

Die L1-Regularisierung löscht Features, um das Modell einfacher zu machen. Dabei wird versucht, wichtige Features zu behalten. Allerdings kann das Modell ungenauer werden, wenn die Merkmale auf 0 gesetzt werden.

Die L2-Regularisierung (Weight Decay) setzt unwichtigere Features auf kleinere Weights, um deren Einfluss zu minimieren. Ziel ist es, so viel Information wie möglich zu erhalten.

Dropout ist eine Technik, bei der während des Trainings zufällig Neuronen im Netzwerk deaktiviert werden. Dies kann das „Auswendiglernen“ verhindern. (Ying, 2019, S.4f.)

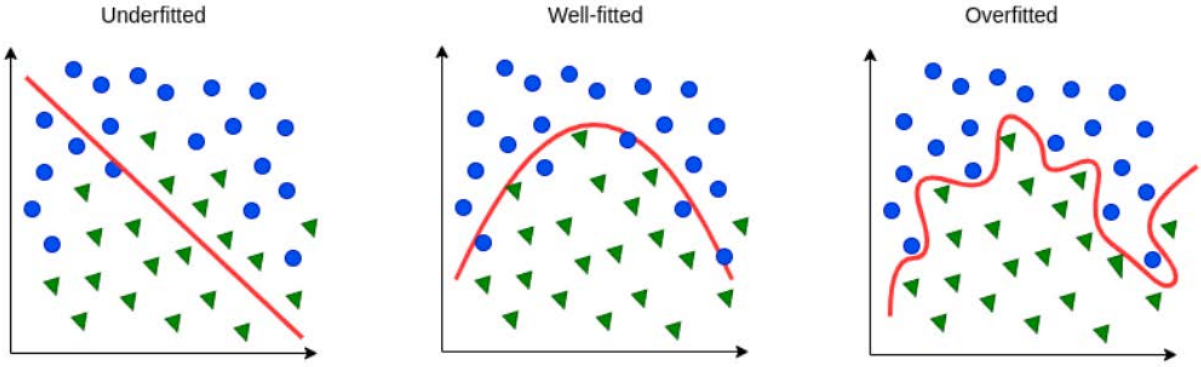


Abb. 7: Probleme beim Finetuning (PI.EXCHANGE, 2024)

2.2.3.3 Parameter-Efficient Fine-Tuning

Eine weitere Möglichkeit, Under- und Overfitting zu reduzieren und Leistungsprobleme zu vermeiden, ist Parameter-Efficient Fine-Tuning (PEFT) (Abb. 8). Dabei werden nicht alle Parameter während des Trainings verändert, sondern immer nur ein Teil. Die meisten Parameter bleiben dabei „eingefroren“ (Xu u. a., 2023). Dies hat neben den bereits genannten Vorteilen auch Vorzüge in Bezug auf die Speichergröße und das „katastrophale Vergessen“ (Mangrulkar und Paul, 2023).

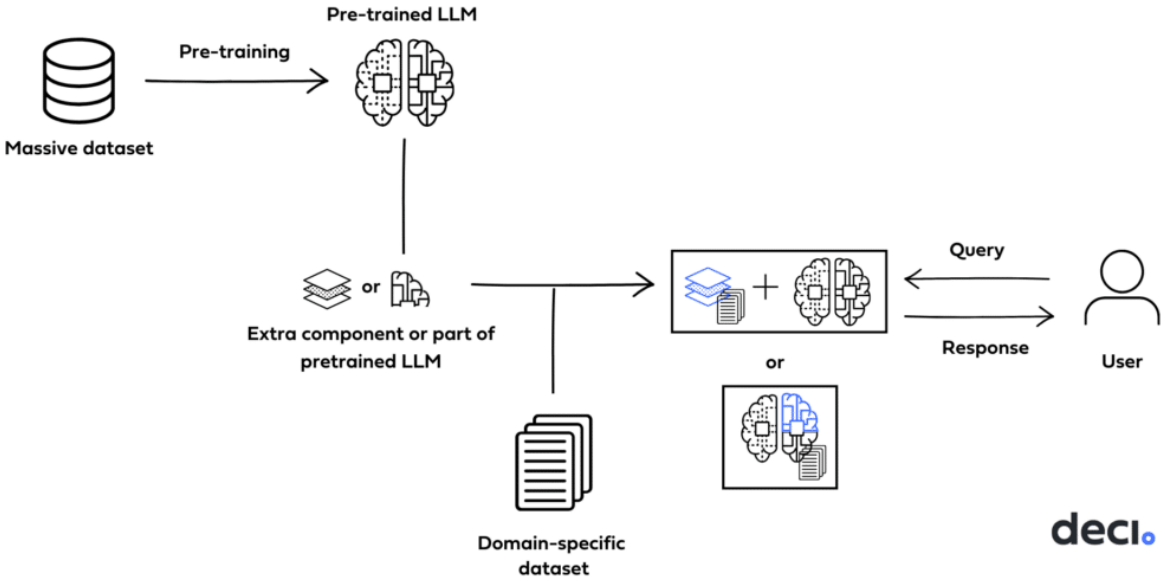


Abb. 8: Parameter-Efficient Fine-Tuning (Nabwani, 2023)

2.2.4 Prompt Engineering

Mithilfe von Prompts können Anweisungen und Fragen an ein KI-Modell gestellt werden (Query). Je nachdem, welche Informationen in den Prompts übergeben werden, kann die Antwort des LLM anders, besser oder schlechter ausfallen. Das kann bei der Modellauswahl helfen, aber auch die Flexibilität und die Benutzererfahrung verbessern (AWS, 2024a; Google Career Certificates, 2024). Mithilfe verschiedener Techniken lassen sich Prompts optimieren, um entsprechende Ergebnisse zu erhalten - Prompt Engineering (Abb. 9). Ein Artikel von AWS beschreibt es folgendermaßen: „Ein Benutzer kann eine unvollständige Problembeschreibung eingeben, z. B. *„Wo kann ich ein Hemd kaufen?“* Intern verwendet der Code der Anwendung eine technische Aufforderung, die besagt: *„Sie sind ein Verkaufsassistent für ein Bekleidungsunternehmen. Ein Benutzer mit Sitz in Alabama, USA, fragt Sie, wo er ein Shirt kaufen können. Antworten Sie mit den drei nächstgelegenen Filialen, die derzeit ein Shirt auf Lager haben.“* Der Chatbot generiert dann relevantere und genauere Informationen.“ (AWS, 2024a, Abs.3). Neben der Veränderung der Abfrage können auch Beispiele in einem Prompt zu einem besseren Ergebnis führen.

Beim Zero-Shot Prompting wird der KI eine Aufgabe gestellt, für die sie nicht direkt trainiert wurde. Es werden keine zusätzlichen Informationen oder Anweisungen mitgegeben. Dies kann schwierig sein, da die KI nicht über das notwendige Wissen verfügt, um die Aufgabe zu lösen. Zur Unterstützung bei der Lösung kann One-Shot Prompting verwendet werden, bei dem der Anfrage ein Beispiel mitgegeben wird. Auf diese Weise hat das Modell einen Orientierungspunkt, wie die Ausgabe aussehen sollte. Noch bessere Ergebnisse können ggf.

durch Multi-Shot Prompting erzielt werden, indem mehrere Beispiele in der Query mitgeliefert werden. (IBM Deutschland GmbH, 2023; Google Career Certificates, 2024; Mehta, 2024)

Prompt Engineering ist zusammengefasst, die Optimierung einer Query auf eine Art und Weise, dass das Modell das tut, was der Entwickelnde/ Benutzende von diesem erwartet. Google hat dazu Best Practices verfasst (Google for Developers, 2024):

1. Machen Sie deutlich, welche Inhalte oder Informationen am wichtigsten sind.
2. Strukturieren Sie die Aufforderung: Definieren Sie zuerst die Rolle, geben Sie Kontext-/Eingabedaten an und stellen Sie dann die Anweisung bereit.
3. Verwenden Sie spezifische, vielfältige Beispiele, um das Modell einzugrenzen und genauere Ergebnisse zu erzielen.
4. Verwenden Sie Einschränkungen, um den Umfang der Modellausgabe einzuschränken. So vermeiden Sie, dass sich die Anweisungen auf sachliche Ungenauigkeiten beziehen.
5. Zerlegen Sie komplexe Aufgaben in einfachere Aufforderungen.
6. Das Modell bitten, seine Antworten zu bewerten oder zu prüfen, bevor sie erstellt werden „Beschränken Sie Ihre Antwort auf drei Sätze.“, „Berücksichtigen Sie auf einer Skala von 1 bis 10, wie kurz und prägnant“ Ihre Aussage ist.“ „Glauben Sie, dass das richtig ist?“.

Sollten Probleme zu groß sein, um diese mit wenigen Sätzen oder mit Beispielen zu verfassen, können folgende Techniken helfen (Tabelle 4):

Tabelle 4: Komplexe Prompt-Engineering Techniken (in Anlehnung an AWS, 2024a, Abs.5)

Technik	**Erklärung**
Chain-of-Thought-Aufforderung	Chain-of-Thought-Aufforderung ist eine Technik, die eine komplexe Frage in kleinere, logische Teile zerlegt, die einen Gedankengang nachahmen. Dies hilft dem Modell, Probleme in einer Reihe von Zwischenschritten zu lösen, anstatt die Frage direkt zu beantworten. Dies verbessert seine Argumentationsfähigkeit.
Tree-of-Thought-Aufforderung	Die Tree-of-Thought-Technik verallgemeinert die Chain-of-Thought-Aufforderung. Sie fordert das Modell auf, einen oder mehrere mögliche nächste Schritte zu generieren. Anschließend wird das Modell bei jedem möglichen nächsten Schritt mithilfe einer Baumsuchmethode ausgeführt.
Maieutische Aufforderung	Maieutische Aufforderungen ähneln Chain-of-Thought-Aufforderungen. Das Modell wird aufgefordert, eine Frage mit einer Erklärung zu beantworten. Das Modell wird dann

	aufgefordert, Teile der Erklärung zu erklären. Inkonsistente Erklärungsbäume werden beschnitten oder verworfen. Dies verbessert die Leistung bei komplexen Argumenten mit gesundem Menschenverstand.
Auf Komplexität basierende Eingabeaufforderungen	Diese Prompt-Engineering-Technik beinhaltet die Durchführung mehrerer Chain-of-Thought-Rollouts. Sie wählt die Rollouts mit den längsten Gedankenketten und wählt dann die am häufigsten getroffene Schlussfolgerung aus.
Generierte Wissensvermittlung	Bei dieser Technik wird das Modell aufgefordert, zunächst relevante Fakten zu generieren, die für die Ausführung der Aufforderung erforderlich sind. Anschließend wird die Eingabeaufforderung abgeschlossen. Dies führt häufig zu einer höheren Abschlussqualität, da das Modell von relevanten Fakten abhängig ist.
Von der geringsten bis zur häufigsten Aufforderung	Bei dieser Prompt-Engineering-Technik wird das Modell aufgefordert, zunächst die Teilprobleme eines Problems aufzulisten und sie dann nacheinander zu lösen. Dieser Ansatz stellt sicher, dass spätere Teilprobleme mit Hilfe von Antworten auf frühere Teilprobleme gelöst werden können.
Automatische Eingabeaufforderung	Bei dieser Technik wird das Modell aufgefordert, das Problem zu lösen, seine Lösung zu kritisieren und dann das Problem unter Berücksichtigung des Problems, der Lösung und der Kritik zu lösen. Der Problemlösungsprozess wiederholt sich, bis ein vorher festgelegter Grund für den Stopp erreicht ist. Beispielsweise könnten dem Modell die Token oder die Zeit ausgehen, oder das Modell könnte ein Stopp-Token ausgeben.
Automatische Eingabeaufforderung	Diese Prompt-Engineering-Technik beinhaltet einen Hinweis oder einen Hinweis, z. B. gewünschte Schlüsselwörter, um das Sprachmodell zur gewünschten Ausgabe zu führen.

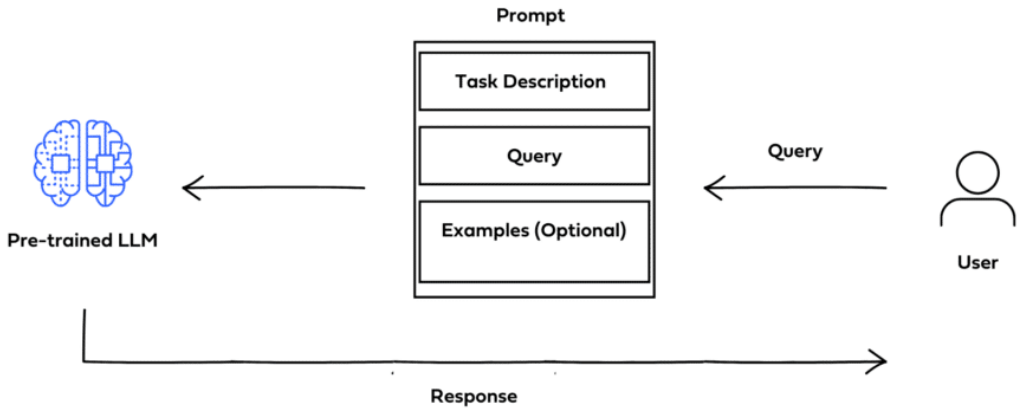


Abb. 9: Prompt Engineering (Nabwani, 2023)

2.3 Webapplikation

Neben reinen Frontend- (React, Vue, etc.) und reinen Backend-Frameworks und -Sprachen (Laravel, PHP, NodeJS, etc.) gibt es auch solche, die Client- und Serverseite kombinieren. Darunter fallen beispielsweise die JavaScript-Frameworks NextJS (Vercel, Inc., 2023) und NuxtJS (Nuxt, 2024). Diese Frameworks bieten Optimierungsmöglichkeiten wie automatisiertes Caching, Server Side Rendering (SSR), Client Site Rendering (CSR) (Vercel, Inc., 2023). Dadurch kann eine Webseite v.a. für kleine und mittelgroße Projekte schnell umgesetzt und veröffentlicht werden.

2.4 Referenzen

2.4.1 Ground News

Grund News (Ground News, 2023) ist eine bekannte unvoreingenommene Nachrichtenseite aus den USA. Sie liefert wertvolle Informationen, wie die Komplexität des Themas gut auf einer Webseite dargestellt werden kann (z. B. durch Vergleiche unterschiedlicher Politikbereiche und KI-Zusammenfassungen). Auch den politischen Bias stellt die Seite gut dar (Abb. 10). Grund News bietet außerdem einen Fakt-Checker an. Sie bietet ausschließlich englischsprachige Medien an.

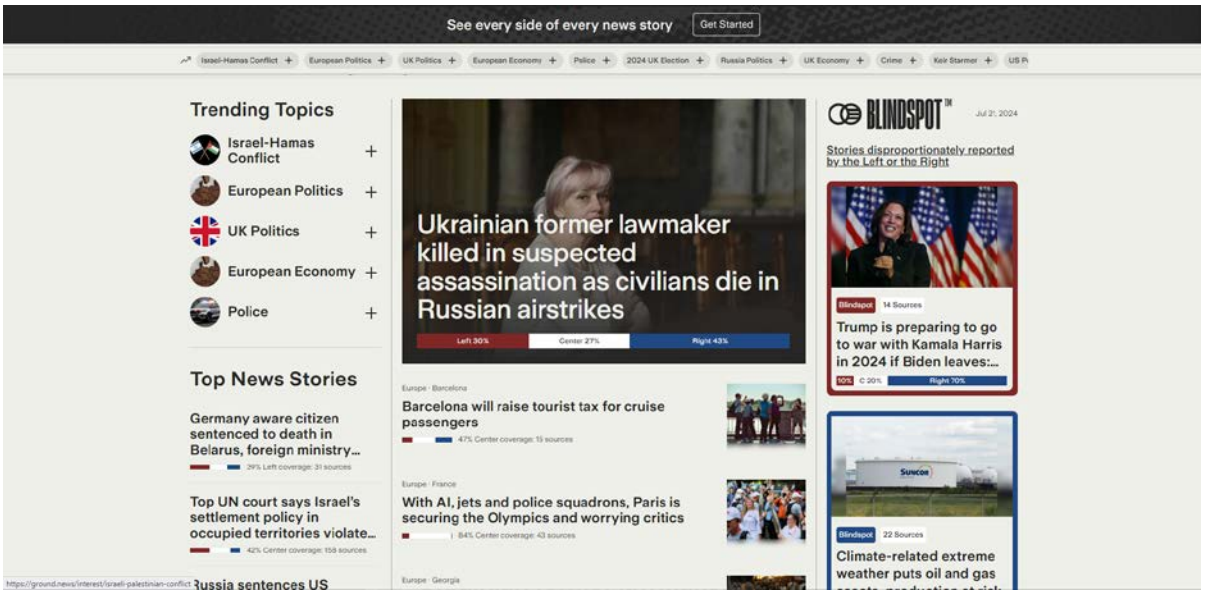


Abb. 10: Ground News Webseite

2.4.2 The Factual

The Factual (The Factual, 2023) ist eine Webseite, die Nachrichten auf Englisch anbietet. Sie hat die Besonderheit, dass sie einen großen Wert auf Faktencheck legt. So wird zu jeder Nachricht eine prozentuale Angabe gemacht, inwieweit die Aussagen des Artikels faktisch richtig sind. Die Art und Weise, wie der Fakt Checker in diesem Fall eingebaut ist, ist sehr ansprechend und vorbildlich für dieses Projekt (Abb. 11). The Factual hat ein solides Layout, ist allerdings nicht so übersichtlich wie andere Referenzen.

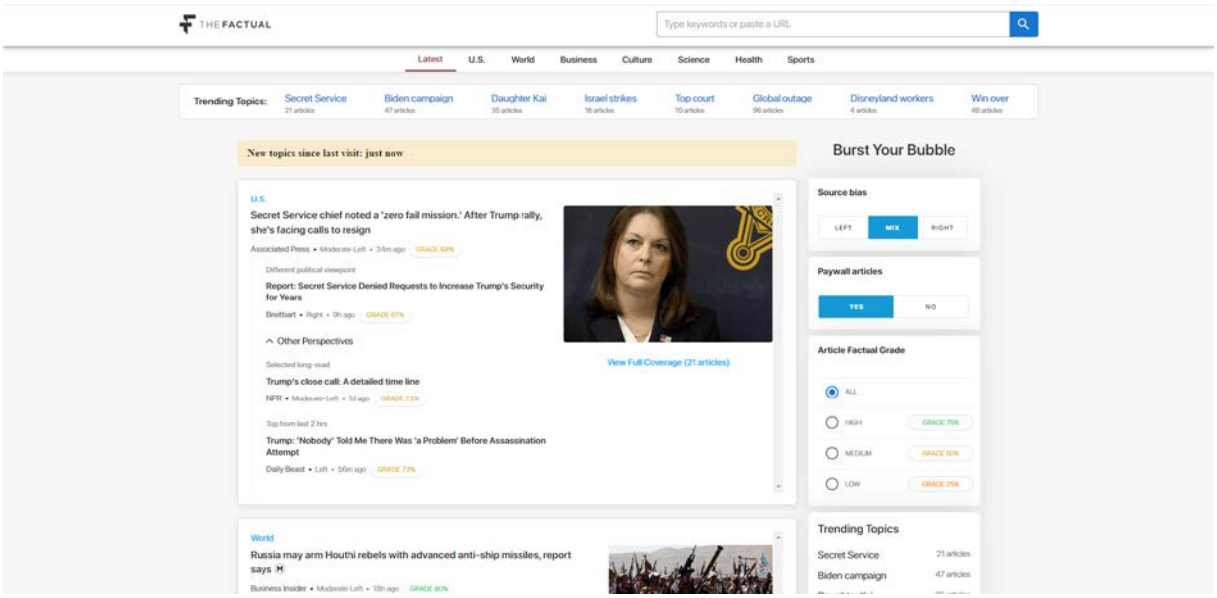


Abb. 11: The Factual Webseite

2.4.3 Google News

Google News (Google News, 2023) ist eine viel genutzte Webseite, um allerlei Nachrichten zu empfangen. Sie ermöglicht es den Nutzenden, individuelle Interessen auszuwählen, und passt so (auch im Laufe der Zeit) die angezeigten Nachrichten an die Lesenden an. Das Design von Google News ist intuitiv und übersichtlich. Zudem ist die News-Seite gut etabliert. Daher setzte ich in diesem Projekt auf ein ähnliches Design, um Benutzern einen leichten Einstieg zu bieten. Die Filterung der Nachrichten kommt in diesem Projekt nicht zum Einsatz (Abb. 12).

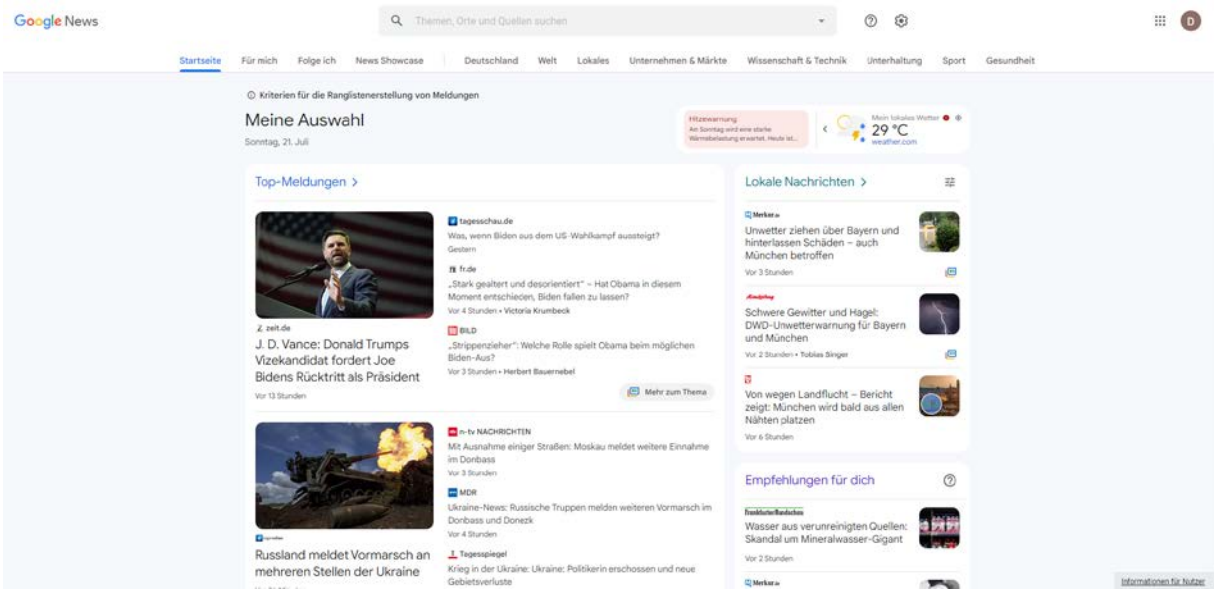


Abb. 12: Google News Webseite

3 Methodik

Tabelle 5: Definition der Arbeitsphasen

Nr.	Arbeitsphase	Ergebnis (Umfang / Qualität / Wirkung)
1	Datenbeschaffung	Speicherung von Nachrichtendaten in einer Datenbank.
2	Labeling	Labeling der Daten.
3	Research: KI-Modell	Finden eines Basis-KI-Modells, das den Anforderungen einer Nachrichtenklassifizierung standhält.
4	Training & Finetuning	Über die Nachrichtendaten trainiertes KI-Modell.
5	Entwicklung der Applikation	Frontend-Applikation zur Darstellung der Nachrichten.
6	Finalisierung	Kombination der einzelnen Arbeitsschritte zu einer Applikation.

Um das Projekt umzusetzen, wurde ein Zeitplan (Abb. 13) ausgearbeitet, welcher den zeitlichen Aufwand schätzt. Alle Meilensteine (Tabelle 5) finden in iterativen Arbeitsphasen statt, indem sich das Projekt weiterentwickelt (z. B. auch durch Fehler oder unerwartete Ereignisse). Aus diesen Gründen ist der Zeitplan lediglich eine Orientierungshilfe für den Projektfortschritt (ganzer Zeitplan: Anhang 2).

Die Evaluation findet jeweils innerhalb oder nach einer Arbeitsphase statt. Sie ist individuell auf einen Arbeitsschritt abgestimmt und kann eine Iteration des Arbeitsschritts auslösen.

Aa Name	Blockiert	Blockiert von	Datum
▶ Datenbeschaffung	Labeling Feature: Zusammenfassung der News		11. März 2024 → 24. März 2024
▶ Labeling	Training & Finetuning	Datenbeschaffung Automatisierung: Speicherung der Daten	28. März 2024 → 14. April 2024
▶ Research: KI-Model	Training & Finetuning		18. April 2024 → 28. April 2024
▶ Training & Finetuning	Finalisierung	Research: KI-Model Labeling	2. Mai 2024 → 15. Juni 2024
▶ Entwicklung der Application	Finalisierung Feature: Zusammenfassung der News		16. Juni 2024 → 14. Juli 2024
▶ Feature: Zusammenfassung der News		Datenbeschaffung Entwicklung der Application	18. Juli 2024 → 26. Juli 2024
▶ Finalisierung		Training & Finetuning Entwicklung der Application	27. Juli 2024 → 4. August 2024
▶ Sonstige Zeit (Blocker)			8. August 2024 → 23. August 2024

Abb. 13: Produktionsphasen und Meilensteine

3.1 Datenbeschaffung

Inhalt: Ziel dieses Arbeitsschrittes ist es, eine große Anzahl von Nachrichtendaten in einer eigenen Datenbank zu speichern.

3.1.1 Planung der Datenbeschaffung

Mögliche Vorgehensweisen:

1. **Web Scraping** ist eine Methode, Inhalte aus einer Webseite auszulesen bzw. zu kopieren. Dies funktioniert üblicherweise über einen Crawler, welcher bestimmte Webseiten aufsucht, um die Inhalte herauszufiltern und in einer Datenbank zu speichern (Myra Security GmbH, 2023).
2. **Nachrichten-APIs/ RSS-Feeds** werden von Unternehmen (z. B. Google) bereitgestellt. Über die Schnittstellen können Nachrichten abgerufen und verwendet werden.
3. **Datensatz**, welcher von Experten zusammengestellt und gelabelt ist. Für dieses Projekt werden zwischen 5.000 und 50.000 diverse, gelabelte Nachrichtenartikel benötigt.

Geplante Umsetzung: Für dieses Projekt habe ich mich für die Schnittstelle der News API (News API, 2023) entschieden, welche auch ein kostenloses Angebot hat. Die News API bietet eine Vielzahl von Arten von Nachrichtenartikeln an, darunter auch diverse deutsche Verlage und Nachrichten. Auf dem Markt gibt es noch weitere Anbieter, welche jedoch überwiegend kostenpflichtige Schnittstellen verkaufen. Die Nutzung einer News API bietet in erster Linie einen zeitlichen Vorteil gegenüber der Scraping-Methode. Zudem müssen die Daten nicht mehr aufbereitet werden und kommen bereits mit Metadaten wie Schlagwörtern, Namen des Autors etc. immer im selben Format.

Um die Daten über die API empfangen zu können, plane ich ein Python-Skript zu schreiben, das täglich zu einer bestimmten Uhrzeit automatisch alle Nachrichten aus verschiedenen Kategorien abfragt und diese in einer Datenbank speichert.

3.1.2 Evaluierung der Datenbeschaffung

Ziel der Evaluierung der Datenbeschaffung ist es, zu überprüfen, wie viele Nachrichtenartikel in welcher Form in der Datenbank gespeichert sind.

Die erste Evaluierung, soll nach dem ersten Tag der Beschaffung durchgeführt werden. Es muss überprüft werden, ob die Inhalte (z.B. Texte) in der Datenbank ausreichend und vielfältig sind. Außerdem soll überprüft werden, wie viele Daten pro Tag in der Datenbank gespeichert werden. Danach soll alle zwei Tage geprüft werden, wie viele Daten in der Datenbank sind und ob diese ausreichend und divers sind.

Diese Überprüfung erfolgt durch einen Funktionstest meinerseits. Sollte sich herausstellen, dass die Inhalte aus der NewsAPI nicht ausreichen, muss auf eine alternative Methode (z. B. RSS-Feeds oder Scraping) zurückgegriffen werden.

3.2 Labeling

3.2.1 Planung des Labelings

Inhalt: Das Ziel dieses Arbeitsschrittes ist es, gelabelte Daten in der Datenbank zu haben. Das Label repräsentiert die gewünschte Ausgabe der KI. In diesem Fall z. B. die politische Einordnung: POLITISCH LEICHT LINKS, POLITISCH LINKS, POLITISCH MITTEL, POLITISCH LEICHT RECHTS oder POLITISCH RECHTS.

Mögliche Vorgehensweisen:

- **Prelabeld-Datensatz,** indem sowohl die Nachrichtenartikel als auch die zugehörigen Labels beinhaltet sind.

- **Automatisiertes Labeling durch eine KI,** welche die Nachrichten automatisch einem politischen Spektrum zuordnet (API).
- **Manuelles Labeling durch eine KI:** Copy & Paste der Nachrichten und automatisches Labeling über eine KI.
- **Manuelles Labeling durch „normale“ Personen** (Freunde, Familie oder Personen online) ordnen die Nachrichten im politischen Spektrum ein.
- **Manuelles Labeling durch Experten,** die Nachrichten im politischen Spektrum einordnen.

Geplante Umsetzung: Die Umsetzung dieses Projektabschnitts ist komplex, wie in Tabelle 6 veranschaulicht:

Tabelle 6: Mögliche Umsetzungen des Labelings mit Zeitaufwand, Kosten und Professionalität

Art	Zeitaufwand	Kosten	Professionalität
Prelabeld Datensatz	Gering	Gering	Gering – Hoch
Automatisiertes Labeling über eine KI	Gering – Mittel	Gering - Hoch	Gering – Mittel
Manuelles Labeling über eine KI	Hoch	Gering	Gering – Mittel
Manuelles Labeling über „normale“ Personen	Sehr hoch	Gering	Gering – Mittel
Manuelles Labeling über Experten	Sehr hoch	Mittel	Hoch

Im Idealfall gibt es eine von Experten öffentlich verfügbare Prelabled Datenbank, die den Projektanforderungen entspricht. Das ist bisher nicht der Fall.

Die Autoren des Papers „Media Bias in German News Articles: A Combined Approach“ (Spinde, Hamborg und Gipp, 2020) möchten mir allerdings eine Software bereitstellen, die über LLMs automatisiert Labels zu Texten erstellen kann (Spinde, Mandl und Hilmer, 2024). Die Qualität ist laut Spinde vergleichbar mit menschlichen Antworten (Anhang 5).

Sollte die Verwendung der Software von Spinde nicht möglich sein, wird auf das GPT-3.5 Modell zurückgegriffen, um ein automatisiertes und schnelles Labeling durchzuführen.

3.2.2 Evaluierung des Labelings

Das Ziel der Evaluierung für das Labeling ist, dass alle Nachrichtenartikel ein entsprechendes Label haben.

Die Evaluierung soll nach den ersten 100 gelabelten Artikeln stattfinden, um festzustellen, ob das Labeling-Skript prinzipiell funktioniert. Daraufhin soll alle 1000 Artikel eine Überprüfung

stattfinden, die feststellt, ob das Labeling weiterhin funktioniert oder ob Probleme auftreten. Das kann durch einen Funktionstest durch mich erfolgen.

Sollte etwas an dem Labeling nicht funktionieren, muss das Skript angepasst oder eine andere Methode verwendet werden. Eine (externe) Überprüfung der Korrektheit der Labels wird es nicht geben.

3.3 Research: KI-Modell

3.3.1 Planung der Research des KI-Modells

Inhalt: Das Ziel dieses Arbeitsschrittes ist es, ein geeignetes KI-Basismodell zu finden.

Mögliche Vorgehensweisen:

- **Recherche & Empfehlungen**: Durch Recherche, v.a. im Internet, wird ein geeignetes KI-Modell gefunden. Wichtige Kriterien: Verständnis für Deutsch und Fähigkeiten zur Textklassifikation.

Geplante Umsetzung: Auf Hugging Face werden sehr viele kostenlose Base-Models zur Verfügung gestellt. Mithilfe des Internets, Empfehlungen und Tests (Prompt Engineering) verschiedener Modelle werde ich eines auswählen.

3.3.2 Evaluierung: Research des KI-Modells

Ziel der Evaluierung: Research des KI-Modells ist es, die Eignung eines KI-Modells zu bestimmen. Deshalb wird die Evaluierung vor der Auswahl des Base-Models stattfinden, während die Modelle miteinander verglichen werden.

Um das geeignete Modell zu finden, wird eine Messung durchgeführt, die unterschiedliche Parameter der Modelle misst und miteinander vergleicht (siehe 2.2.2). Den Basismodellen wird dafür die Aufgabe gegeben, Nachrichtenartikel aus der Datenbank in die entsprechenden Kategorien einzuordnen. Je nachdem, wie gut die Messwerte sind, wird das Modell mit den besten Werten für die eigene Forschung weiterverwendet.

Diese Methode ist sehr flexibel, da ich eine große Anzahl verschiedener Modelle gegeneinander testen und daraufhin entscheiden kann, welches Base-Model sich am besten für meine Zwecke eignet.

3.4 Training & Fine-Tuning

3.4.1 Planung des Trainings und Fine-Tunings

Inhalt: Das Ziel dieses Arbeitsschrittes ist es, ein trainiertes Modell (Fine-Tuning) zu erhalten, das Nachrichtenartikel als Input aufnehmen und eine entsprechende politische Einordnung ausgeben kann.

Mögliche Vorgehensweisen:

- **Hardware:**
  - **Google Colab**: Verwendung der von Google Colab zur Verfügung gestellten Programmierumgebung, um die GPU von Google Colab verwenden zu können. Diese wird als kostenlose (langsame) und kostenpflichtige Version zur Verfügung gestellt. *(Gut für Tests, schlechter für große Modelle, ggf. langsam)*
  - **SAE-Hardware:** Verwendung einer GPU aus der SAE, um das Training durchzuführen. Da die Rechner der SAE oft mit leistungsfähigen GPUs ausgestattet sind, können diese nach der Installation der notwendigen Software für das Training verwendet werden. *(Installation der Software ggf. aufwendig, kostengünstig, relativ schnell, schlechter für große Modelle)*
  - **Externe Hardware (mieten):** Mieten einer GPU. *(skalierbar, teurer für bessere Hardware, schnell)*
  - **Private GPU:** Kauf einer leistungsfähigen GPU. *(teuer, individuell)*
- **Training:**
  - **Auto Train:** Möglichkeit, ein Modell einfach durch das Hochladen von Daten zu trainieren. Dabei muss kein Code geschrieben werden. *(schnell, angewiesen auf Anbieter, evtl. teurer)*
  - **Manuelles Training (Code):** Schreiben von ML Trainings Code, um das Modell zu trainieren. *(anspruchsvoller, individuell, anpassbar, ggf. günstiger)*

Geplante Umsetzung: Geplant ist eine Umsetzung mit einer Kombination aus SAE-Hardware und dem Training mit eigenem Code. Dies hat den Vorteil, dass es kostengünstiger umgesetzt werden kann und gleichzeitig eine solide Performance über die integrierte GPU bietet. Des Weiteren ist das Training sehr gut individuell auf das Projekt anpassbar.

Sollte das Training auf einem Computer der SAE nicht umsetzbar sein, möchte ich auf eine günstige externe GPU z.B. von Hugging Face oder Vast.ai zurückgreifen.

Der Code wird mit Python, PyTorch und Hugging Face umgesetzt. Vorkenntnisse dafür habe ich mir bereits im ASP angeeignet (Anhang 1.1).

3.4.2 Evaluierung des Trainings & Fine-Tunings

Ziel der Evaluierung während des Fine-Tunings ist es, die Leistung des Modells kontinuierlich zu überwachen. Dazu werden laufend Messungen durchgeführt und alle 10.000 Iterationen überprüft. Der Loss und die Accuracy des Modells werden mithilfe von PyTorch berechnet. Das Ziel ist es, eine Genauigkeit von mindestens 75 % zu erreichen und den Loss unter 25 % zu halten. Um dies zu erreichen, können Anpassungen vorgenommen werden, wie z. B. die Verlängerung der Trainingszeit, das Hinzufügen weiterer Trainingsdaten oder der Einsatz alternativer Algorithmen. Diese Evaluierungsmethode ist sehr flexibel, da sie die Möglichkeit bietet, durch verschiedene Stellschrauben die Leistungsfähigkeit des Modells kontinuierlich zu verbessern.

3.5 Entwicklung der Webapplikation

3.5.1 Planung der Entwicklung der Webapplikation

Inhalt: Das Ziel dieses Arbeitsschritts ist es, eine einfache Webanwendung zu entwickeln, welche Nachrichteninhalte auf der Seite ausgeben kann.

Mögliche Vorgehensweisen:

- Umsetzung mit einem Baukastensystem (z. B WordPress)
- Programmierte Umsetzung (Full-Stack-Framework: NextJS)
- Programmierte Umsetzung (Frontend: React, Backend: NodeJS/ PHP)

Geplante Umsetzung: Um das Projekt so flexibel wie möglich zu gestalten, möchte ich die Web-App mit NextJS umsetzen. Das Framework bietet nicht nur die Vorteile der Individualisierung, sondern auch, dass es Front- und Backend kombiniert und einfach veröffentlicht werden kann (Vercel, Inc., 2023). Meine Erfahrung mit JavaScript/ NodeJS und NextJS bieten zudem entscheidende Vorteile gegenüber PHP.

3.5.2 Evaluierung der Entwicklung der Webapplikation

Ziele der Evaluierung der Entwicklung der Webapplikation sind es, ein funktionierendes UI (z. B. Buttons, Links, etc.) und Backend (Datenbank, APIs und Inferenz) sicherzustellen. Die Evaluierung soll immer nach der Fertigstellung eines neuen Moduls oder einer neuen Funktion erfolgen und über Unit-Testing durchgeführt werden.

Durch regelmäßige Tests bin ich sehr anpassungsfähig und kann schnell handeln, sobald Probleme mit Funktionen auftreten.

3.6 Finalisierung

3.6.1 Planung der Finalisierung

Inhalt: Ziel dieser Projektphase ist die Fertigstellung des Prototyps.

Geplante Umsetzung: Alle Module werden über Datenbanken und APIs zu einer Web-App verbunden. Diese wird auf einem Webserver bereitgestellt.

Ablauf:

1. Abruf der Nachrichten eines Tages
2. Speicherung der Nachrichten in einer Datenbank
3. Automatisierte Analyse (KI) der Nachrichten und politisches Ranking (Anreicherung der Daten in der Datenbank)
4. Ausgabe der Nachrichten auf der Webapplikation

3.6.2 Evaluierung der Finalisierung

Ziele der Evaluation der Finalisierung sind es, zu prüfen, ob alle Komponenten vollständig und problemlos zusammenarbeiten. Dabei ist es wichtig, dass die regelmäßige Ausgabe von neuen Nachrichtenartikeln inkl. Labeling festgestellt wird.

Für die Evaluierung werden Funktionstests, Integrationstests und End-2-End-Tests durchgeführt. Mit diesen werden die Funktionen der Webanwendung getestet.

Da die Tests am Ende des Projektes stattfinden, kann es sein, dass nicht genug Zeit ist, weitere Änderungen an der Applikation durchzuführen. Jedoch werden viele Eventualitäten bereits in den vorhergehenden Evaluierungen beseitigt, um die Evaluierung so reibungslos wie möglich zu gestalten.

4 Durchführung

4.1 Datenbeschaffung

Im Rahmen der Datenbeschaffung wurde insbesondere darauf geachtet, eine umfangreiche und vielfältige Datenbasis in einer Datenbank zu erfassen. Diese Daten werden im Projekt für die KI-Trainingseinheiten verwendet.

4.1.1 Beschaffung der Daten

Die kostenlose Version der News API (News API, 2023) ermöglicht 50 Anfragen innerhalb von 12 Stunden (max. 100 Abfragen innerhalb von 24h). Dafür ist lediglich die Einrichtung eines Kontos sowie die Generierung eines privaten Schlüssels erforderlich.

Um die Daten aus der API abfragen zu können, habe ich ein Python-Skript in einer Docker-Umgebung geschrieben, welches mittels Cronjob am Laufen gehalten wurde. Herausforderungen ergaben sich durch die starken Einschränkungen der (kostenlosen) Schnittstelle. Die Einschränkungen bedeuteten, dass nicht alle Nachrichten aus sämtlichen Quellen gleichzeitig heruntergeladen werden konnten, sondern pro Anfrage lediglich eine maximale Nachrichtenanzahl abgerufen werden konnte. Um dennoch eine ausreichende Menge an relevanten Daten zu erhalten, wurden 100 Schlüsselwörter ausgewählt, die eine politische Relevanz besitzen und von verschiedenen politischen Positionen unterschiedlich aufgegriffen und wiedergegeben werden. Beispiele für solche Schlagworte sind „AfD“, „Außenpolitik“, „Einwanderung“ oder „Polizei“. Diese Schlüsselwörter wurden verwendet, um die API gezielt nach Nachrichten zu durchsuchen, die diese Begriffe enthalten. Auf diese Weise konnten jeden Tag 100 Anfragen gestellt werden.

Das Skript sollte ursprünglich täglich automatisiert ausgeführt werden. Nach 4 Tagen hatte ich jedoch bereits 30.000 Einträge in der Datenbank gespeichert, sodass es nicht notwendig war, das Skript dauerhaft laufen zu lassen.

4.1.2 Speicherung der Daten

Die Daten aus dem Skript wurden automatisch in einer Datenbank gespeichert. Ich habe mich für eine MongoDB (dokumentenbasierte NoSQL-Datenbank) entschieden, die vorerst nur in einem lokalen Docker-Container lief (v.a. wegen unschätzbarer Datenmengen). Da die Daten von der API als JSON geliefert wurden, war dies eine ideale Lösung. Später wurde die Datenbank in einen MongoDB-Atlas (MongoDB, Inc., 2023) und eine Coolify Instanz (coolLabs Technologies Bt., 2024) verschoben. Mithilfe von Coolify können u. a. Datenbanken und Webseiten auf demselben Root-Server gehostet werden. Unter der Haube wird dafür Docker verwendet. Coolify bietet darüber hinaus z. B. Integrationen für GitHub an.

4.1.3 Evaluation der Datenbeschaffung

Die Ergebnisse des ersten Tests zeigten, dass die Datenspeicherung erfolgreich war. Innerhalb von 24 Stunden wurden etwa 10.000 Artikel erfasst. Es stellte sich jedoch heraus, dass der Inhalt der Artikel widererwartend nicht vollständig durch die API ausgeliefert wurde. Für jeden Artikel wurden nur die ersten ~30 Zeichen des Artikels mitgeliefert. Da die kurzen Texte für den Trainingsprozess ein Problem darstellen wird, ging ich wie folgt vor:

1. **Verwendung der ~30 Zeichen**
   Für die Darstellung auf der Webseite ist dies ausreichend. Um möglichst richtige Labels zu den Texten zu erhalten, ist allerdings möglichst viel Inhalt notwendig.
2. **Anfrage an Verlage gestellt**
   Ich habe alle Verlage, die bislang in der Datenbank gespeichert wurden (ca. 120), mittels Scripts eine Anfrage per E-Mail (Anhang 3) gestellt, ob diese mir Zugriff auf eine Schnittstelle bzw. die Daten geben können. Hierbei habe ich primär Absagen erhalten.
3. **Scraping der Nachrichten**
   Um herauszufinden, ob es legitim ist (z. B. wegen Urheberrechtsverletzungen), die Artikel über die mitgelieferte URL zu scrapen und in die Datenbank zu speichern, wurde ein SAE nahestehender Anwalt kontaktiert. Allerdings ohne Erfolg. Von einem Mitglied des Forscherteams der Bias Media Group wurde mir mitgeteilt, dass Data-Mining eine Best Practice für Forschungsarbeiten sei. Ähnliche Aussagen habe ich auch vom BR erhalten: „Die Einschätzung unseres zuständigen Juristen […] lautete, dass öffentlich zugängliche Inhalte via Text und Data Mining genutzt werden können […].“ (Loistl, 2024) (Anhang 4). Diese Aussagen ließen sich über das Internet bestätigen (uni-hamburg, 2020; Klawonn, 2024). Daher habe ich ab diesem Zeitpunkt Scraping aktiv eingesetzt.

Bei den Überprüfungen, die alle zwei Tage stattfanden, wurde die Anzahl der Artikel kontrolliert, jedoch konnte die Diversität der Artikelkategorien noch nicht festgestellt werden, da das erforderliche Label fehlte.

Nach fünf Tagen stellte sich heraus, dass die API viele irrelevante Artikel lieferte, darunter auch URLs wie Amazon.de. Diese irrelevanten Artikel und Duplikate wurden aus der Datenbank entfernt, sodass die bereinigte Anzahl der Artikel etwa 15.000 betrug.

4.2 Labeling

4.2.1 Umsetzung des Labelings

Die „Media Bias Group“ hat mir ein Python-Framework zur Verfügung gestellt, das mit unterschiedlichen KI-Modellen das Labeling von Daten in menschlicher Qualität durchführen kann (Media Bias Group, 2024). Für das Projekt habe ich eine Vorabversion bereitgestellt bekommen (Spinde, Mandl und Hilmer, 2024) (Anhang 5). Um die Daten labeln zu können, hatte ich verschiedene Möglichkeiten (Tabelle 7):

Tabelle 7: Möglichkeiten des Artikel-Labelings

Methode	Vorteil	Nachteil/ Probleme
Lokales Labeln auf einem privaten PC mit GPU (Nvidia GTX 1060)	Kostenlos; dauerhafter Zugriff auf den PC; Daten verlassen den privaten Raum nicht.	Sehr lange Dauer (> 300 Tage) beim Labeln, da die GPU nicht schnell genug ist.
Lokales Labeln auf dem PC der SAE mit GPU (Nvidia 2080 Turbo)	Kostenlos; Daten werden nicht in die Cloud geschickt.	Kein dauerhafter Zugriff auf die Ressourcen; lange Dauer des Labelings (> 20 Tage).
Labeln in der (Google) Cloud	Wenn die GPU reserviert ist, dauerhafter Zugriff; Auswahl an vielen versch. GPUs mit unterschiedlicher Leistung.	Kostenpflichtig; teilweise schwierig, an GPUs zu kommen.

Die Methoden 1 und 2 erschienen durch den hohen Zeitaufwand zunächst nicht sinnvoll. Die beste Möglichkeit schien daher das Labeling über die Google Cloud zu sein, auch wenn die Daten dadurch nicht mehr nur lokal gespeichert werden und der Google-Dienst kostenpflichtig ist (Google bietet ein Budget von 270 € an, das beantragt und genutzt wurde). Um die Labels für die Artikel über die Google Cloud GPUs laufen zu lassen, habe ich mein Python-Skript als Docker Container an die Artifact Regestry von Google gesendet. Von dort konnte es direkt verwendet werden, um eine Google Compute Engine zu erstellen. Auf diese habe ich eine Nvidia T4 (ca. 0,30ct/ h) und L4 (ca. 0,95ct/h) GPU ausgewählt und separat voneinander getestet. Es stellte sich heraus, dass die Konfiguration mit L4 ca. doppelt so schnell war wie die T4. Allerdings war die Auswahl oft durch die Verfügbarkeit der GPUs eingeschränkt.

4.2.2 Evaluation des Labelings

Nach dem ersten Test mit 100 Nachrichten zeigte sich, dass der Workflow für das Labeling funktionierte, aber die Inferenzgeschwindigkeit langsam war. Trotz der besseren GPUs in der Google Cloud war die Inferenzzeit länger als erwartet. Oft über 300s/Iteration (it), obwohl die Bach Size (Anzahl der Artikel, die in die GPU geladen und verarbeitet werden) bei einem Artikel pro Iteration lag. Außerdem schien die Inferenz langsamer zu werden, je länger sie lief. Nach einer Weile und nach Rücksprache mit Christoph Mandl vom Media Bias Research Team wurde dieses Problem auf ein Caching-Problem im Zusammenhang mit dem Modell „occiglot/occiglot-7b-de-en-instruct“ zurückgeführt (nach ca. 3.000 Artikeln). Nach dem Austausch des Modells auf „lex-hue/Delexa-7b“ konnte das Labeling auf dem privaten PC durchgeführt werden (Methode 1, um das Google Cloud Kontingent für das Training zu sparen). Die Inferenzzeit betrug ca. 50s/it. Nach ca. 10.000 Artikeln wurden Probleme mit der Länge der Artikel festgestellt, da der Cache überfüllt war. Daraufhin wurden die Artikel auf maximal 5.000 Zeichen gekürzt, was zu einer Verbesserung der Label-Performance führte, da weniger Daten in den (V)RAM geladen wurden.

Einschub: Während des Trainings (4.4.2.4 Kontext Problem) stellte sich außerdem heraus, dass die Kontextlänge des Etikettierungsmodells auf 512 Token begrenzt ist; alle weiteren Tokens wurden von dem Modell automatisch abgeschnitten.

Es zeigte sich auch, dass die Labels große Unterschiede in der Ausprägung aufwiesen. Der Bias der Artikel fand sich vor allem bei POLITISCH MITTEL. Auch die Labels POLITISCH LINKS und POLITISCH RECHTS erhielten einige Bewertungen. Die Labels POLITISCH LEICHT LINKS und POLITISCH LEICHT RECHTS erhielten wenige Labels (Abb. 14).

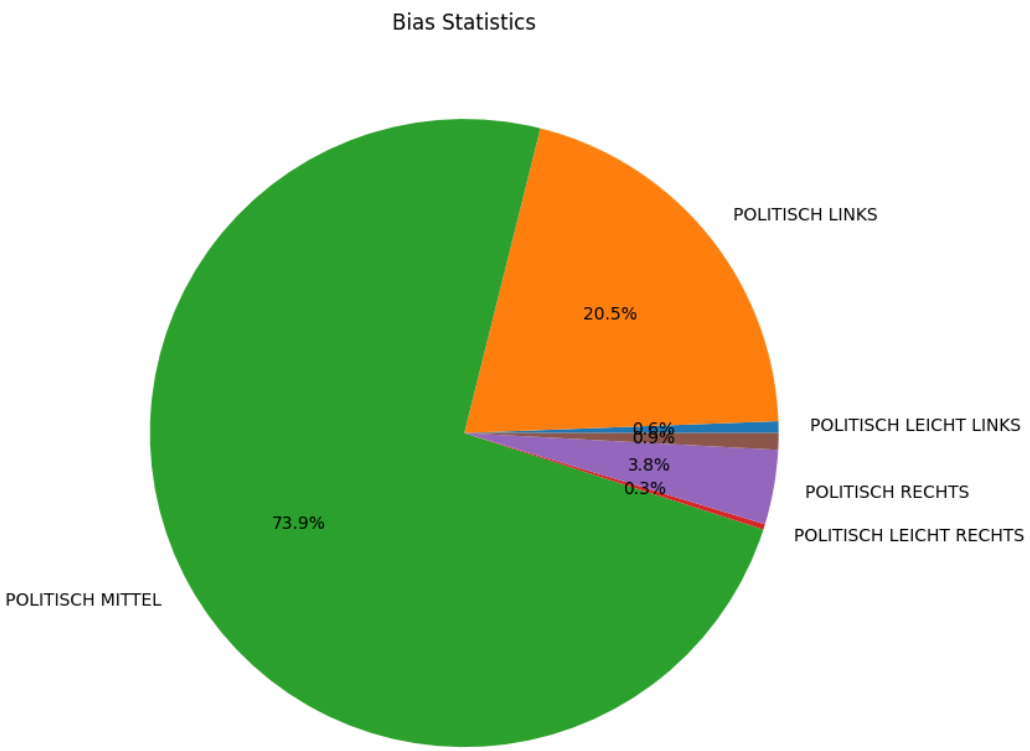


Abb. 14: Bias Statistik

Aus diesem Grund wurden Nachrichtensender (z. B. Die Welt, Junge Freiheit und FOCUS) ausgewählt, die typischerweise links und rechts stehen, um möglichst unterschiedliche Artikel und Labels zu finden (Anhäuser, 2017; Bundeszentrale für politische Bildung, 2024). Das neue Ziel ist es, die KI aufgrund der starken zeitlichen Einschränkung auf POLITISCH LINKS, POLITISCH MITTEL und POLITISCH RECHTS zu trainieren, da nicht genügend POLITISCH LEICHT LINKS und POLITISCH LEICHT RECHTS Artikel gefunden werden konnten. In einem längerfristigen Projekt könnte explizites Scraping für POLITISCH LEICHT LINKS und POLITISCH LEICHT RECHTS Artikel über einen längeren Zeitraum durchgeführt werden.

Aufgrund der zahlreichen Komplikationen konnte das Labeling erst 3 Wochen nach der eigentlichen Planung abgeschlossen werden. Weitere Labeling-Arbeiten zogen sich durch das gesamte Projekt bis hin zur finalen Version (Abb. 15).

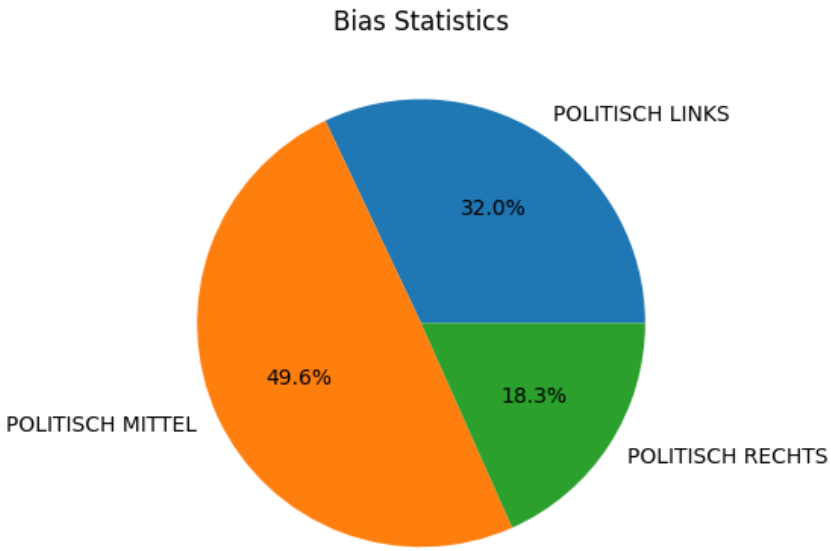


Abb. 15: Bias Statistik (Final) POLITISCH LINKS (11840), POLITISCH MITTEL (18343), POLITISCH RECHTS (6777)

4.3 Research: KI-Modell

4.3.1 Research des KI-Modells und Vergleich

Um das geeignete KI-Modell zu finden, wurden über Hugging Face Modelle gefiltert, die die Anforderungen (Art: Textklassifizierung) erfüllen konnten. Dadurch fielen bereits einige Modelle heraus. Hinsichtlich der Sprache wurden wie geplant deutsche, aber auch mehrsprachige und englische Modelle getestet, da die Modelle intern häufig Übersetzungsarbeiten durchführen können (Tabelle 8).

Tabelle 8: Vollständige Liste der verglichenen Modelle (Hugging Face, 2023)

Name	Größe	Sprache
albert-base-v2	11,8M	Englisch
albert-large-v2	17,9M	Englisch
bert-base-uncased	110M	Englisch
bert-large-uncased	336M	Englisch
dbmdz/bert-base-german-cased	111M	Deutsch
deepset/gelectra-base	110M	Deutsch
deepset/gelectra-large	336M	Deutsch

distilbert-base-multilingual-cased	135M	Multilanguage (inkl. Englisch und Deutsch)
distilbert-base-uncased	67M	Englisch
google/electra-base-discriminator	110M	Englisch
google/electra-small-discriminator	14M	Englisch
google-bert/bert-base-german-cased	110M	Deutsch
intfloat/multilingual-e5-large	560M	Multilanguage (inkl. Englisch und Deutsch)
intfloat/multilingual-e5-large-instruct	560M	Multilanguage (inkl. Englisch und Deutsch)
microsoft/deberta-base	86M	Englisch
microsoft/deberta-large	304M	Englisch
roberta-base	125M	Englisch
roberta-large	355M	Englisch
t5-base	223M	Multilanguage (inkl. Englisch und Deutsch)
t5-large	738M	Multilanguage (inkl. Englisch und Deutsch)
t5-small	60,5M	Multilanguage (inkl. Englisch und Deutsch)
xlnet-base-cased	110M	Englisch
xlnet-large-cased	340M	Englisch

Einige dieser Modelle sind bereits etwas älter. Andere „größer“ bzw. „kleiner“. In der Gesamtheit befinden sich alle ausgewählten Modelle unter den kleineren Modellen (< 1B Paremeter). Dafür habe ich mich entschieden, um das Modell mit möglichst geringen Ressourcen trainieren zu können.

Die Modelle wurden mittels Python-Skript miteinander verglichen. Ich wollte herausfinden, welches Modell die besten Klassifizierungsergebnisse ohne ein vorheriges Fine-Tuning liefern konnte. Deshalb habe ich aus meiner Datenbank an gelabelten Artikeln je Label 1.000 Artikel und die zugehörigen Labels exportiert und diese auf die Modelle getestet. Die Parameter Accuracy, Precision, Recall, Prediction Time, F1-Score wurden dabei miteinander verglichen (siehe: 2.2.2).

4.3.2 Evaluation des KI-Modells (Promt Engineering)

Für die Tests wurden alle ausgewählten Modelle mit denselben Daten (max. 512 Token Eingabegröße) auf zwei unterschiedlichen Systemen getestet. Das erste System ist Google Colab mit einer Nvidia T4 (folgend T4) und das zweite mit einer Nvidia GTX 1060 (folgend 1060).

4.3.2.1 Genauigkeit & Recall

Bei dem Vergleich der Modellgenauigkeit und der Recall ist sowohl bei der T4 als auch 1060 aufgefallen, dass alle Modelle einen ähnlichen Score bei ca. 33 % erzielten (Tabelle 9).

Tabelle 9: Modellgenauigkeit & Recall

Modell	**Modellgenauigkeit (%)**		**Recall (%)**	
	T4	**1060**	**T4**	**1060**
t5-large	35	33	33	33
distilbert-base-multilingual-cased	35	33	33	33
google-bert/bert-base-german-cased	34	33	33	33
bert-base-uncased	34	33	33	33
xlnet-large-cased	34	34	34	34
intfloat/multilingual-e5-large	34	32	32	32
google/electra-base-discriminator	33	34	34	34
deepset/gelectra-base	33	33	33	33
albert-large-v2	33	33	33	33
distilbert-base-uncased	33	33	33	33
google/electra-small-discriminator	33	33	33	33
dbmdz/bert-base-german-cased	33	33	33	33
roberta-base	33	33	33	33
albert-base-v2	33	33	33	33
microsoft/deberta-base	33	33	33	33
t5-base	33	34	34	34
deepset/gelectra-large	33	33	33	33
bert-large-uncased	33	34	34	34
microsoft/deberta-large	33	33	33	33
roberta-large	33	33	33	33
intfloat/multilingual-e5-large-instruct	33	33	33	33
t5-small	33	33	33	33
xlnet-base-cased	32	33	33	33

Die Vermutung lag nahe, dass die Modelle „raten“ und so eine Chance von 33 % haben, die richtige Kategorie zu treffen. Um meiner Annahme nachzugehen, habe ich weitere Experimente durchgeführt.

Experiment 1: Ein Label

In diesem Test habe ich 1.000 Artikel mit dem Label POLITISCH MITTEL aus der Datenbank exportiert und alle Modelle darauf getestet. Die Genauigkeit und der Recall verteilten sich bei diesem Test zwischen 0 % und 100 % ganz anders als bei dem zu vorigen Test. Um sicherzugehen, dass alles richtig gelaufen ist, habe ich den Test 3-mal durchgeführt. Auffallend war, dass es kaum Modelle gab, die nur „gut“ oder nur „schlecht“ abschnitten (Tabelle 10).

Tabelle 10: Experiment 1: Ein Label - Modellgenauigkeit & Recall

Modell	Modellgenauigkeit (%)			Recall (%)		
	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3
albert-base-v2	4	52	16	2	26	8
albert-large-v2	100	100	0	50	50	0
bert-base-uncased	0	100	0	0	100	0
bert-large-uncased	100	67	14	100	33	7
dbmdz/bert-base-german-cased	100	0	100	100	0	50
deepset/gelectra-base	99	93	7	50	46	4
deepset/gelectra-large	0	3	99	0	1	49
distilbert-base-multilingual-cased	0	100	0	0	100	0
distilbert-base-uncased	2	99	100	1	50	100
google/electra-base-discriminator	0	5	1	0	2	0
google/electra-small-discriminator	0	5	29	0	2	14
google-bert/bert-base-german-cased	80	99	32	40	49	16
intfloat/multilingual-e5-large	100	38	0	50	19	0
intfloat/multilingual-e5-large-instruct	68	99	0	34	50	0
microsoft/deberta-base	100	100	100	100	100	100
microsoft/deberta-large	100	100	0	100	100	0
roberta-base	0	50	100	0	25	100
roberta-large	100	0	0	100	0	0
t5-base	2	98	0	1	49	0
t5-large	100	0	100	100	0	100
t5-small	7	100	2	4	100	1
xlnet-base-cased	37	44	63	18	22	32
xlnet-large-cased	94	1	96	47	1	48

Experiment 2: Zwei Label

Wie bei dem Test mit einem Label habe ich daraufhin einen Test mit 2 Labeln (Links und Rechts – je1000 Labels) durchgeführt. Hierbei konnte ich eine Genauigkeit und einen Recall von 0 bis 50 % feststellen. Diesen Test habe ich zweimal durchgeführt, um festzustellen, ob hierbei Modelle in dem einen Durchlauf „gut“ und in dem nächsten „schlecht“ abschneiden (Tabelle 11). Dies konnte ich auch bei diesem Versuch feststellen. Zusätzlich waren die Modelle, die in Test 1 sehr gut (microsoft/deberta-base) oder schlecht (google/electra-base-discriminator) abgeschnitten haben, nicht die Modelle, die in Test 2 sehr gut (distilbert-base-uncased, roberta-base, microsoft/deberta-large) bzw. schlecht (deepset/gelectra-base, google/electra-small-discriminator, google-bert/bert-base-german-cased) abgeschnitten haben.

Tabelle 11: Experiment 2: Zwei Label - Modellgenauigkeit & Recall

Modell	Modellgenauigkeit (%)		Recall (%)	
	Test 1	Test 2	Test 1	Test 2
albert-base-v2	50	0	50	0
albert-large-v2	50	0	33	0
bert-base-uncased	0	49	0	33
bert-large-uncased	20	0	13	0
dbmdz/bert-base-german-cased	0	41	0	27
deepset/gelectra-base	1	7	1	5
deepset/gelectra-large	50	23	33	15
distilbert-base-multilingual-cased	1	45	1	30
distilbert-base-uncased	50	50	50	50
google/electra-base-discriminator	8	48	5	32
google/electra-small-discriminator	2	0	1	0
google-bert/bert-base-german-cased	1	0	1	0
intfloat/multilingual-e5-large	1	49	1	33
intfloat/multilingual-e5-large-instruct	14	50	9	33
microsoft/deberta-base	0	50	0	50
microsoft/deberta-large	50	50	50	33
roberta-base	50	50	50	50
roberta-large	50	0	33	0
t5-base	14	0	9	0
t5-large	0	43	0	29
t5-small	0	50	0	50
xlnet-base-cased	6	38	4	25

xlnet-large-cased	41	0	27	0

Bei einem Label lag die Genauigkeit der Modelle zwischen 0 und 100 %, bei zwei Labels zwischen 0 und 50 % und bei drei Labels zwischen 0 und 35 %. Daraus ergab sich eine direkte Proportionalität zwischen der Anzahl der Labels und der Testgenauigkeit.

Experiment 3: Predictions

Da die Zahlen bisher zwar in einen Zusammenhang gestellt werden konnten, aber nicht aussagekräftig erschienen, habe ich versucht herauszufinden, welche Voraussagen die Modelle machen, wenn ich einen Artikel verwende und ihn immer wieder an demselben Modell teste. Ich habe untersucht, ob Modelle bei wiederholter Eingabe desselben Artikels konsistente Voraussagen liefern. Dazu habe ich denselben Artikel 10-mal an dasselbe Modell gesendet und diesen Vorgang für jedes Modell wiederholt. Dabei stellte sich heraus, dass die Modelle in einer kontinuierlichen Schleife immer dasselbe Label ausgeben. Sobald die Schleife jedoch unterbrochen und neu gestartet wird, kann ein anderes Label generiert werden. Das galt für jedes Modell und jedes Label, die in den Tests verwendet wurden.
Diese Erkenntnisse deckten sich mit den Tests zuvor und der Tatsache, dass jedes Modell bei jedem Durchlauf unterschiedliche „Leistungen“ erbracht hat. Dies kann darauf zurückzuführen sein, dass der Zusammenhang zwischen der Sprache und den Labels, die eine stark vereinfachte Realität darstellen sollen, sehr komplex ist. Zudem wurden die Modelle noch nicht auf diese Daten trainiert und haben vermutlich zu wenig „Wissen“, um die Aufgaben verlässlich durchzuführen. Ob ein Modell in einem Durchlauf gut oder schlecht abschneidet, schien daher bei diesen Metriken zu diesem Zeitpunkt zufällig zu sein.

4.3.2.2 Präzision

Die Präzision unterschied sich deutlich je Modell. Modelle wie deepset/gelectra-base (78 %) oder roberta-base (78 %) hatten eine deutlich höhere Präzision, als beispielsweise google-bert/bert-base-german-cased (T4: 52 %, 1080: 45 %) oder dbmdz/bert-base-german-cased (44 %). Auffallend war, dass die Modellgröße nicht unbedingt einen signifikanten Unterschied in der Präzision des Modells zu machen schien. Zu erkennen war außerdem, dass derselbe Test nicht dieselben Metriken erzeugte. Hier unterschieden sich die Modelle in den Tests teilweise erheblich (roberta-large: T4 (47 %), 1060 (78 %)) (Abb. 16, Tabelle 12). Ich habe mich im weiteren Verlauf vor allem auf die Modelle konzentriert, die ähnliche Ergebnisse bei beiden Systemen liefern konnten.

Da für die Anwendung vor allem die Präzision relevant ist, habe ich auf diese Metrik einen besonderen Wert gelegt.

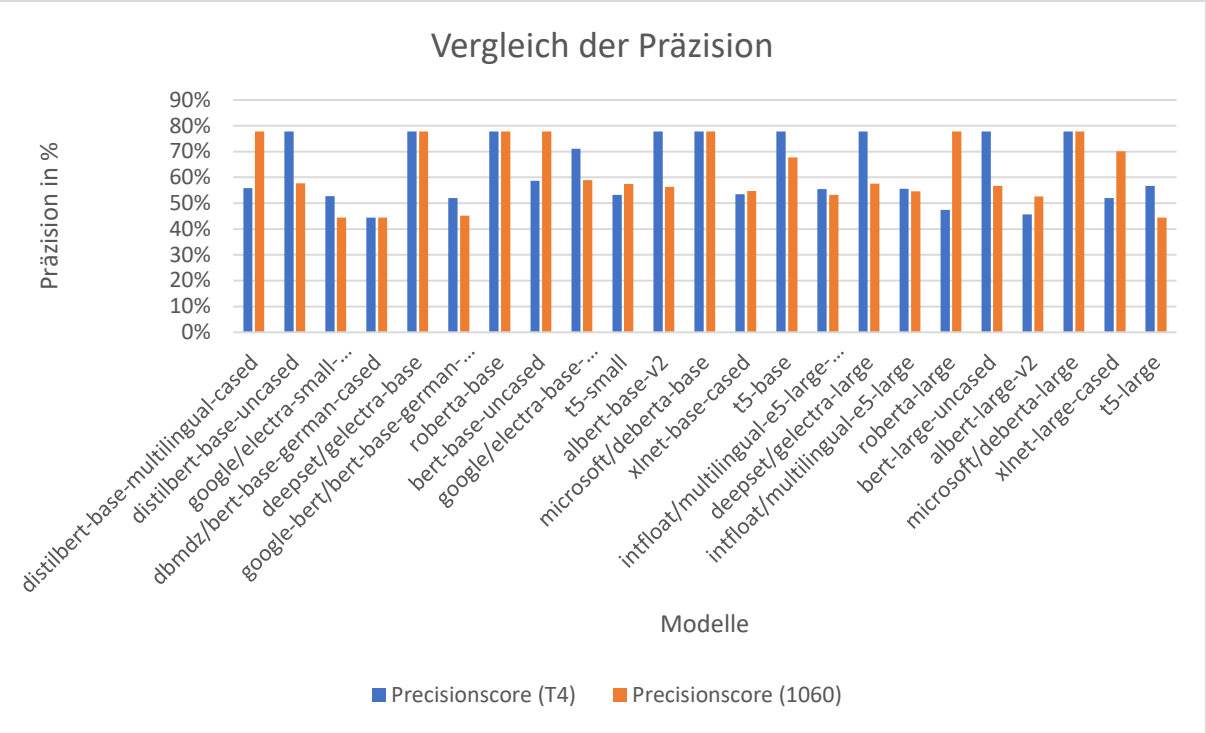


Abb. 16: Vergleich der Modellpräzision

Tabelle 12: Modellpräzision

Modell	**Modellpräzision (%)**	
	T4	**1060**
deepset/gelectra-base	78	78
distilbert-base-uncased	78	58
roberta-base	78	78
albert-base-v2	78	56
microsoft/deberta-base	78	78
t5-base	78	68
deepset/gelectra-large	78	58
bert-large-uncased	78	57
microsoft/deberta-large	78	78
google/electra-base-discriminator	71	59
bert-base-uncased	59	78
t5-large	57	44
distilbert-base-multilingual-cased	56	78
intfloat/multilingual-e5-large	56	55
intfloat/multilingual-e5-large-instruct	56	53
xlnet-base-cased	53	55
t5-small	53	57

google/electra-small-discriminator	53	44
xlnet-large-cased	52	70
google-bert/bert-base-german-cased	52	45
roberta-large	47	78
albert-large-v2	46	53
dbmdz/bert-base-german-cased	44	44

4.3.2.3 F1-Score

Die Distribution der F1-Werte erstreckte sich in einem sehr eingeschränkten Bereich von 28 – 17 %. Auffällig war, dass sich die beiden Tests stark unterschieden (Abb. 17, Tabelle 13). Bei dieser Metrik konnten vor allem die Modelle intfloat/multilingual-e5-large (T4: 27 %, 1060: 26 %) und xlnet-base-cased (T4: 24 %, 1060: 26 %) überzeugen.

Tabelle 13: F1-Score der Modelle

Modell	**F1-Score (%)**	
	T4	**1060**
distilbert-base-multilingual-cased	27	17
t5-large	27	17
intfloat/multilingual-e5-large	27	26
bert-base-uncased	26	17
t5-small	25	19
xlnet-base-cased	24	26
google-bert/bert-base-german-cased	23	17
xlnet-large-cased	22	17
intfloat/multilingual-e5-large-instruct	21	23
albert-large-v2	18	20
google/electra-base-discriminator	17	27
roberta-large	17	17
deepset/gelectra-base	17	17
google/electra-small-discriminator	17	17
dbmdz/bert-base-german-cased	17	17
distilbert-base-uncased	17	17
roberta-base	17	17
albert-base-v2	17	27
microsoft/deberta-base	17	17
t5-base	17	18
deepset/gelectra-large	17	18

bert-large-uncased	17	28
microsoft/deberta-large	17	17

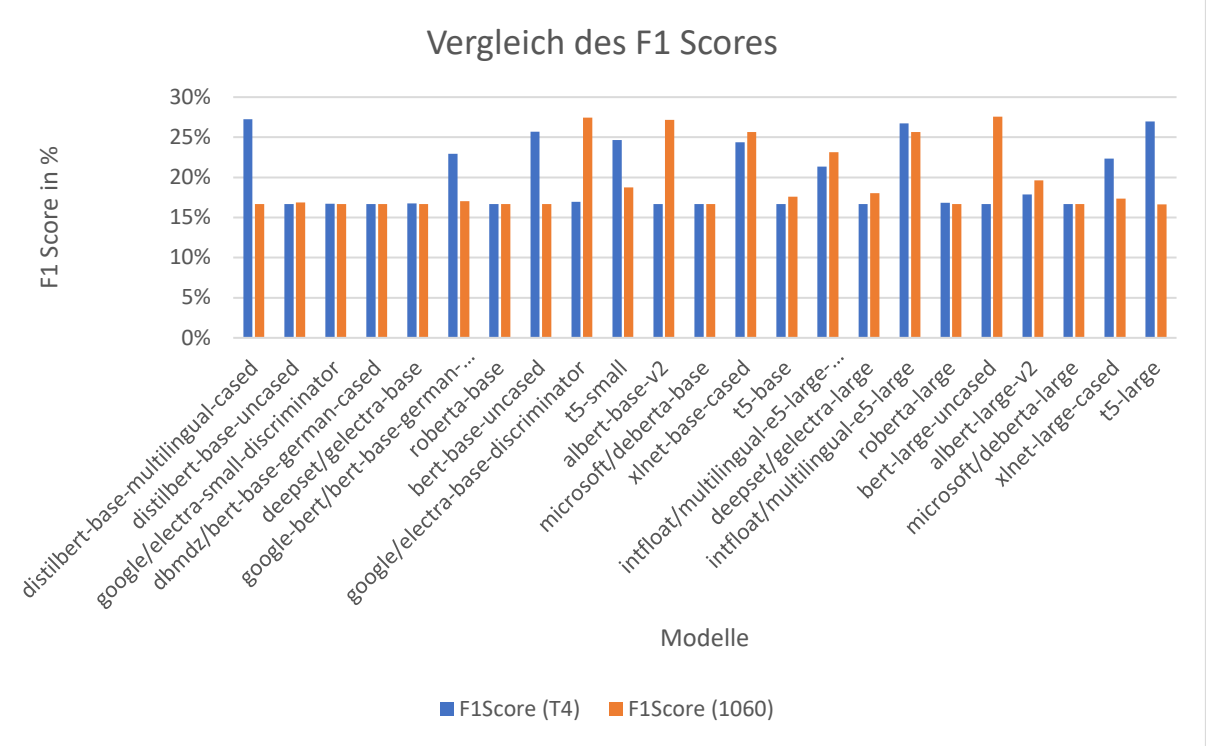


Abb. 17: Vergleich des F1 Scores

4.3.2.4 Ausführungszeit

Die Ausführungszeiten waren sehr unterschiedlich und reichte je nach Modell und System von 34 bis 941 Sekunden (Abb. 18, Tabelle 14). In den Tests konnte außerdem ein direkter Zusammenhang mit dem Modell, der Größe und der Geschwindigkeit festgestellt werden. Kleine und mittelgroße Modelle wie distilbert-base-multilingual-cased und google/electra-small-discriminator waren deshalb tendenziell auch schneller als große Modelle wie xlnet-large-cased und t5-large. In dieser Metrik konnten des Weiteren direkte Vergleiche zwischen den beiden Systemen gezogen werden. Es ließ sich erkennen, dass die Grafikkarte T4 ca. 40 % schneller ist als die 1060, die in den Tests verwendet wurde.

Da die für diese Arbeit zur Verfügung stehende Zeit und die benötigten Hardware-Ressourcen begrenzt sind, hatte die Geschwindigkeit der Modelle einen hohen Stellenwert.

Tabelle 14: Ausführungszeit der Modelle

Modell	**Zeit in Sekunden**	
	T4	**1060**
distilbert-base-multilingual-cased	34	49
distilbert-base-uncased	34	54

google/electra-small-discriminator	39	53
dbmdz/bert-base-german-cased	52	82
deepset/gelectra-base	54	83
google-bert/bert-base-german-cased	54	75
roberta-base	58	103
bert-base-uncased	59	97
google/electra-base-discriminator	59	93
t5-small	64	96
albert-base-v2	70	111
microsoft/deberta-base	106	190
xlnet-base-cased	119	208
t5-base	150	406
intfloat/multilingual-e5-large-instruct	154	300
deepset/gelectra-large	156	183
intfloat/multilingual-e5-large	159	223
roberta-large	174	241
bert-large-uncased	179	222
albert-large-v2	181	245
microsoft/deberta-large	285	416
xlnet-large-cased	323	501
t5-large	425	941

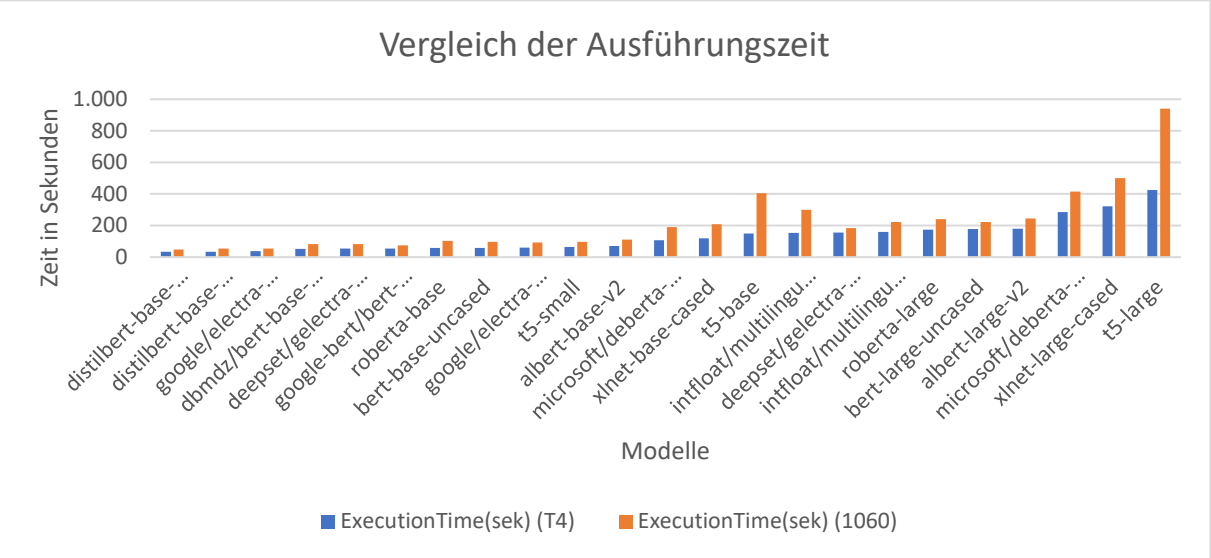


Abb. 18: Vergleich der Ausführungszeit der Modelle

4.3.2.5 Kontext Problem

Durch weitere Untersuchungen und bereits durchgeführtes Training (siehe 4.4.2.4), stellte sich heraus, dass zu den ursprünglichen Vermutungen noch ein weiteres Problem dafür sorgen könnte, dass die Genauigkeit & Recall bei allen Modellen Ähnlichkeiten aufwiesen: „der Kontext“. Der Kontext beschreibt bei einem Modell die Länge der Token, die das Modell als Wissensspeicher verwenden kann. Dieser liegt bei kleinen Modellen, von welchen auch viele für diesen Test verwendet wurden, bei 512 Tokens (entspricht < 512 Wörtern). Das Problem könnte deshalb sein, dass die Größe des Kontextes ausschlaggebend dafür ist, wie gut die Artikel in die politischen Kategorien eingeordnet werden konnten. Bis zu diesem Zeitpunkt wurden Texte, die länger als 512 Token waren, automatisch abgeschnitten. Es gab verschiedene Ansätze, dieses Problem zu lösen. Der erste bestand darin, größere Modelle (nicht Bert) mit einem größeren Kontext zu verwenden. Alternativ wurde bereits untersucht, wie ein Bert-Modell mit kleinem Kontext gut trainiert werden kann (Sun u. a., 2020).

Ich habe mich zunächst auf den Vergleich mit größeren Modellen konzentriert, um eine Einschätzung zu erhalten, ob diese ein besseres Ergebnis erzielen können. Wichtig anzumerken sind hierbei begrenzte monetäre und zeitliche Mittel. Daher habe ich mich weiterhin nicht für Modelle wie meta-llama/Meta-Llama-3-8B oder mistralai/Mistral-7B-Instruct-v0.3 entschieden. Diese Modelle sind mit 7 bzw. 8 Milliarden Parametern deutlich größer als die zuvor getesteten Modelle mit maximal 560 Millionen Parametern. Zeitgleich benötigen diese deutlich mehr Ressourcen während des Trainings und der Inference. Hierbei gibt es zwar Möglichkeiten, die Modelle auch mit einer kleinen GPU zu laden (z. B. Quantisierung über bitsandbytes), jedoch benötigt das Training und die Inference sehr viel Zeit. Bei dem Experiment habe ich mich deshalb auf kleine Modelle mit einem größeren Kontextfenster konzentriert (Tabelle 15):

Tabelle 15: Vergleich von Modellen mit größerem Kontextfenster (Parameter- & Kontextlänge)

Modell	Paramente Anzahl	Kontext Länge
EleutherAI/gpt-neo-1.3B	1,3 Milliarden	2048
l-yohai/bigbird-roberta-base-mnli	110 Millionen	4096
allenai/longformer-base-4096	149 Millionen	4096
google/bigbird-roberta-base	110 Millionen	4096

Um die Modelle miteinander vergleichen zu können, wurde das ursprüngliche Skript um eine qLora 4bit Quantisierung erweitert (Belkada u. a., 2023; Dettmers u. a., 2023). Dies ermöglichte größere Modelle auf kleinen Hardwareressourcen auszuführen. Außerdem wurde

die zu testende Artikelzahl von 1.000 Artikel pro Parameter auf 100 Artikel pro Parameter reduziert, da die Laufzeit der Modelle wesentlich länger war. google/bigbird-roberta-base benötigte 248 Sekunden, EleutherAI/gpt-neo-1.3B 1046 Sekunden, l-yohai/bigbird-roberta-base-mnli 245 Sekunden und ai/longformer-base-4096 294 Sekunden auf einer Nvidia GeForce 1060.

Die Ergebnisse des Experiments zeigten, dass alle 4 Modelle im Bereich Accuracy und Recall Score ähnlich bei ca. 33 % erhielten. Bei der Percission lagen die Modelle l-yohai/bigbird-roberta-base-mnli und allenai/longformer-base-4096 mit 77 % vorne. Beim F1-Score schnitt das Modell EleutherAI/gpt-neo-1.3B mit 32 % am besten ab (Abb. 19).

Diese Ergebnisse waren den bereits getesteten Modellen sehr ähnlich. Mit dem Unterschied, dass die Laufzeit deutlich länger war. Aus diesem Grund habe ich versucht, den Kontext während des Trainings weniger wichtig werden zu lassen (Sun u. a., 2020) und meine Ziele mit einem kleineren Modell zu erreichen (siehe: 1.3).

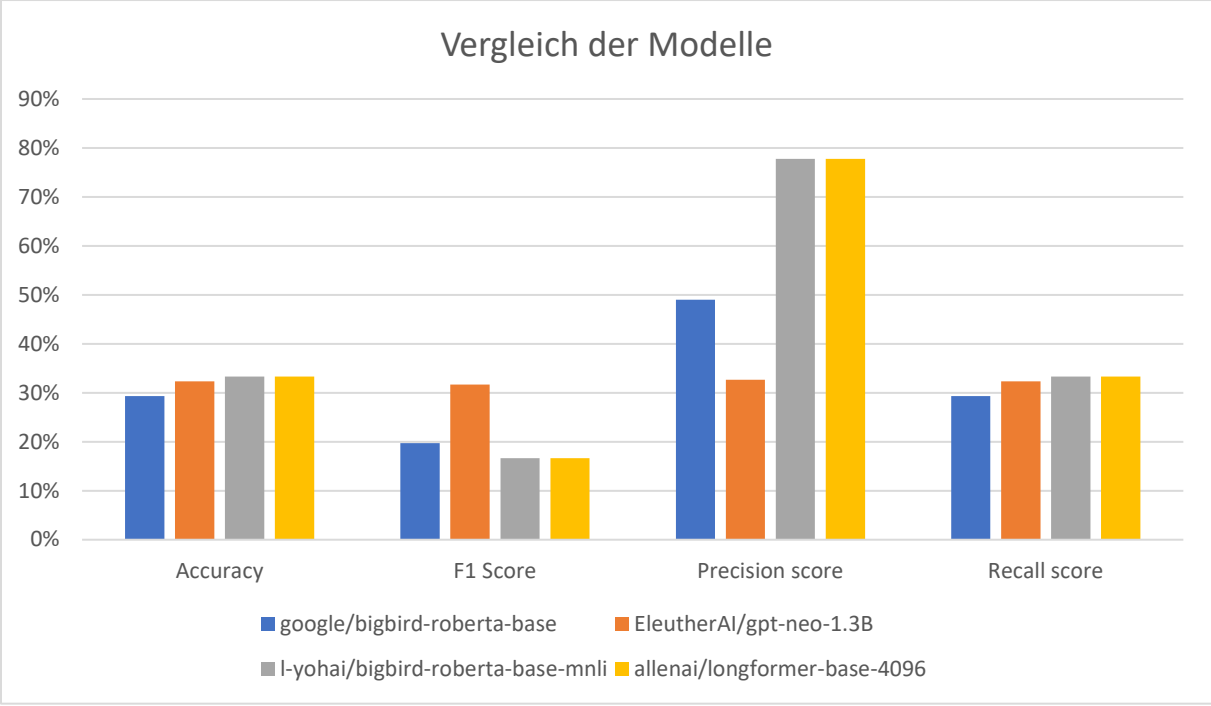


Abb. 19: Vergleich von Modellen mit großem Kontextfenster

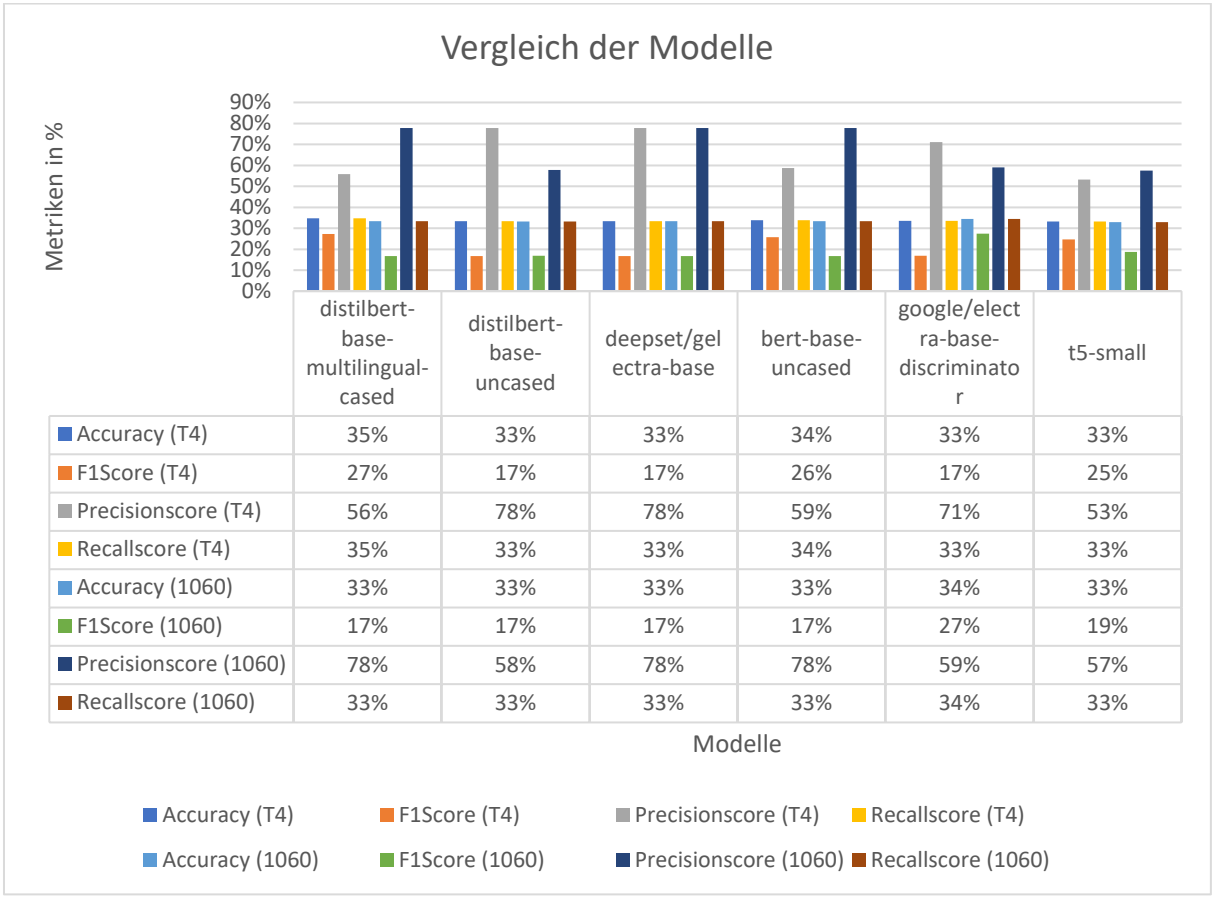
4.3.2.6 Zusammenfassung

Da für diese Arbeit ein strikter Zeitrahmen vorgegeben war und das Modell vor allem ein gutes Verhältnis zwischen Genauigkeit und Trainingszeit/ Inferenz aufweisen musste, wurden die Metriken Precision Score und Execution Time am stärksten gewichtet. Der F1 Score spielte ebenfalls eine Rolle, war allerdings nicht allzu relevant, da die Trainingsdaten (1.000 Artikel/ Label) gleichmäßig aufgeteilt wurden. Die Metriken Genauigkeit und Recall spielten für die Modellentscheidung wenig Rolle, da sie keine signifikanten Unterschiede aufwiesen.

Unter diesen Voraussetzungen habe ich meine Entscheidung auf die folgenden Modelle eingeschränkt (Abb. 20):

- distilbert-base-multilingual-cased
- distilbert-base-uncased
- deepset/gelectra-base
- bert-base-uncased
- google/electra-base-discriminator
- t5-small

Letztlich habe ich mich für das Modell **distilbert-base-multilingual-cased** (Sprache: Multilingual inkl. Deutsch) entschieden, da dieses über beide Tests hinweg eine der besten Geschwindigkeiten (Abb. 21) und Präzisionen aufwies. Des Weiteren hat das Modell die beste Genauigkeit und den besten Recal Score im ersten Test aufweisen können. Es steht unter der Lizenz „Apache License 2.0“ zur Verfügung und war deshalb auch für dieses Projekt uneingeschränkt einsetzbar.



	distilbert-base-multilingual-cased	distilbert-base-uncased	deepset/gelectra-base	bert-base-uncased	google/electra-base-discriminator	t5-small
Accuracy (T4)	35%	33%	33%	34%	33%	33%
F1Score (T4)	27%	17%	17%	26%	17%	25%
Precisionscore (T4)	56%	78%	78%	59%	71%	53%
Recallscore (T4)	35%	33%	33%	34%	33%	33%
Accuracy (1060)	33%	33%	33%	33%	34%	33%
F1Score (1060)	17%	17%	17%	17%	27%	19%
Precisionscore (1060)	78%	58%	78%	78%	59%	57%
Recallscore (1060)	33%	33%	33%	33%	34%	33%

Abb. 20: Vergleich der besten Modelle

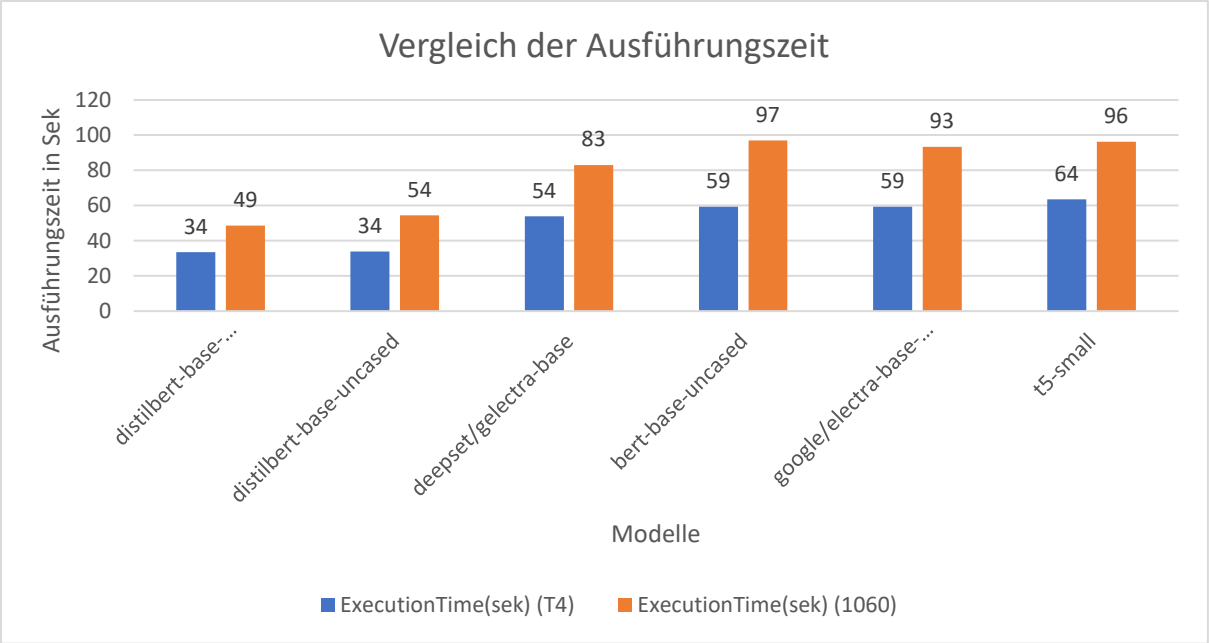


Abb. 21: Vergleich der Ausführungszeit der Modelle

4.4 Training & Fine-Tuning

4.4.1 Training (Fine-Tuning)

Um mit dem Training zu beginnen, habe ich auf Basis von Hugging Face ein Trainingsprogramm geschrieben, welches das ausgewählte Base-Modell auf die zuvor gelabelten Daten fine-tuned. Die während des Trainings anfallenden Daten wurden automatisch an die verknüpfte Schnittstelle von Weights&Biases (wandb) (Weights and Biases, Inc., 2024) übertragen. Diese Seite bereitet die Metriken live während des Trainings in Form von Graphen auf. Zu diesem Zeitpunkt fand das Training auf meiner privaten Hardware (Nvidia GTX1060) und Google Colab (Nvidia T4) statt. Später auch mit einer Google Cloud Instanz (Nvidia L4, T4) und Vast.ai GPUs (VAST.AI, 2024).

4.4.2 Evaluation des Finetunings

Im ersten Trainingsdurchlauf wurde ein komplettes Finetuning durchgeführt. Das Modell wurde mit 1.000 Artikel pro Label trainiert (insgesamt 3.000 Artikel). Zu erkennen war, dass der train/loss kleiner wurde, bis er schwindend gering war (Abb. 22). Im Gegensatz dazu stieg der eval/loss kontinuierlich von knapp unter 1 auf über 2,5 (je kleiner, desto besser) (Abb. 23). Diese Metriken deuteten darauf hin, dass das Modell die Trainingsdaten auswendig lernte. Das Modell konnte die Trainingsdaten sehr gut in die Labels einordnen, aber das Gelernte nicht auf die neuen Daten anwenden (Overfitting, siehe 2.2.3.2).

Abb. 22: Erster Trainingsdurchlauf (train/loss)

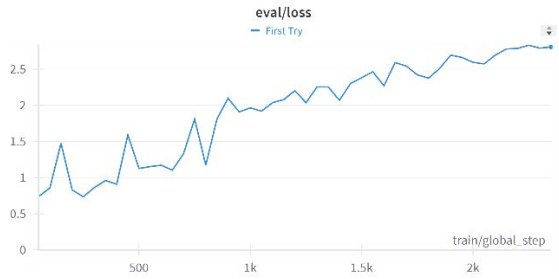


Abb. 23: Erster Trainingsdurchlauf (eval/loss)

4.4.2.1 PEFT-Anpassungen

Um ein Overfitting zu verhindern, habe ich daraufhin eine PEFT-Anpassung mit Lora durchgeführt (siehe 2.2.3.3 Parameter-Efficient Fine-Tuning). Es war zu bemerken, dass das Modell durch diese Anpassung deutlich länger lernte. Allerdings konnte ich beobachten, dass das Training sehr langsam vor sich ging. Außerdem war bei Iteration 1.1k ein Knick sowohl im train/loss (Abb. 24) als auch im eval/loss (Abb. 25) zu erkennen. Auch hier schien das Modell an Overfitting zu scheitern.

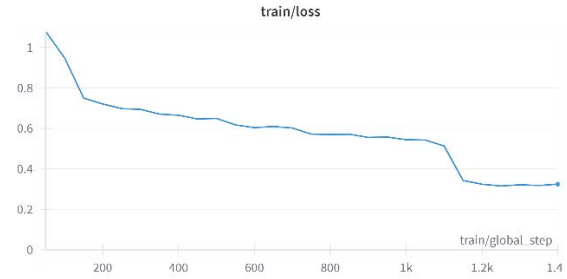


Abb. 24: PEFT-Anpassung (train/loss)

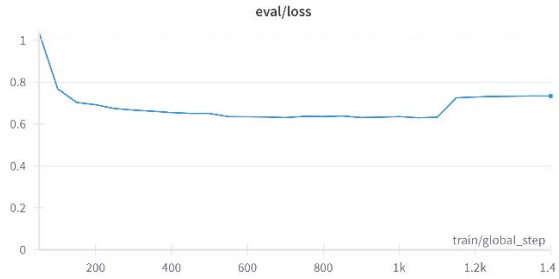


Abb. 25: PEFT-Anpassung (eval/loss)

4.4.2.2 Early Stopping

Zusätzlich zu Lora habe ich daraufhin eine Early Stopping Funktionalität eingebaut. Diese kann das Overfitting weiter reduzieren (siehe 2.2.3.2 Probleme beim Finetuning). Dieser Test sah ähnlich aus wie der vorherige, mit dem Unterschied, dass das Training nur etwa halb so lang durchgeführt wurde (Abb. 26). Bei 800 Steps wurde das Overfitting erkannt und das Training beendet. Anhand der Graphen ist auch zu erkennen, dass das Training nie unter eine Schwelle von 0,65 - 0,75 eval/loss kam. Dies kann z. B. daran liegen, dass die Daten zu komplex sind oder zu wenig Trainingsdaten zur Verfügung stehen.

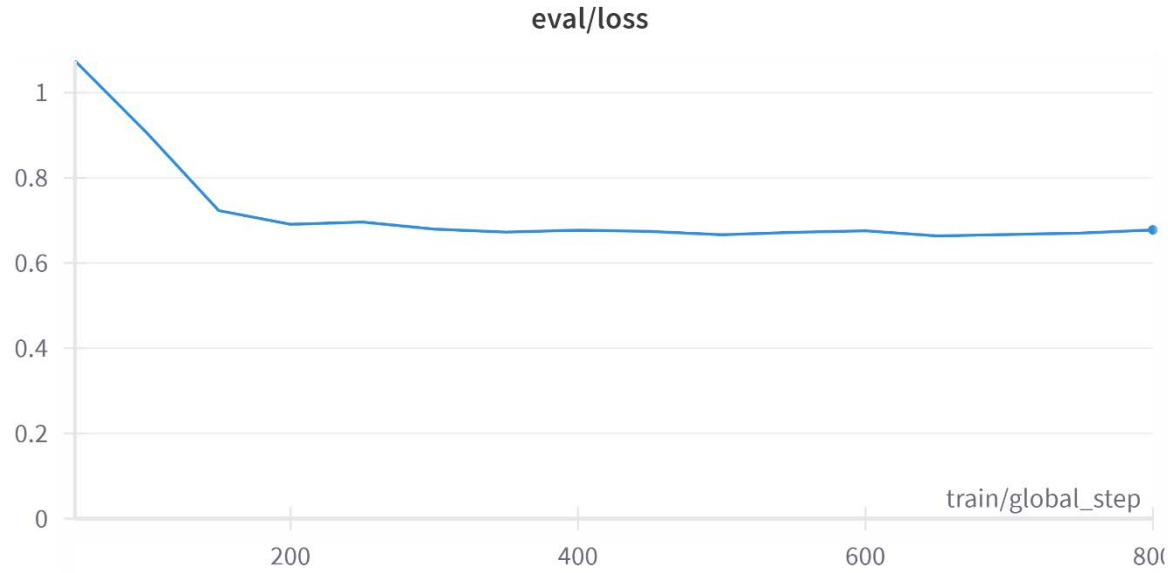


Abb. 26: Early Stopping (eval/loss)

4.4.2.3 Erhöhung der Trainingsdaten

Um festzustellen, ob die Trainingsdaten für das Feintuning nicht ausreichten, habe ich die Anzahl der Daten von 1000 auf 1710 Labels pro Kategorie erhöht (+70 %). Während des Trainings konnte ich eine Verbesserung feststellen, und der eval/loss erreichte etwa 0,6. Dieses Training wurde mit Lora, einem Dropout von 0,05 und Early Stopping durchgeführt. Da die Ergebnisse gut waren, erhöhte ich die Datenmenge weiter auf 3.773 Labels pro Kategorie. Der eval/loss verbesserte sich erneut und fiel unter 0,6. Anschließend erhöhte ich die Trainingsdaten auf 5.071 Labels pro Kategorie. Auch hier verbesserte sich der eval/loss auf ca. 0,51 (Abb. 27). Daraus schloss ich, dass weitere Daten nötig waren, um die Ergebnisse weiter zu verbessern.

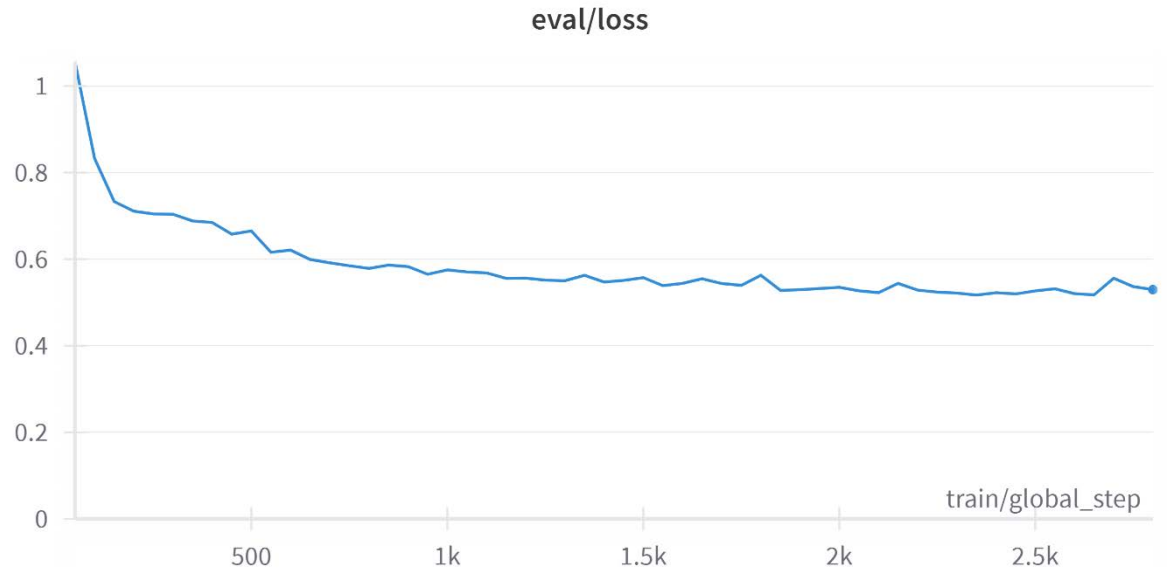


Abb. 27: 5071 Trainingsdaten (eval/loss)

4.4.2.4 Kontext & Hyperparameter

An dieser Stelle ist mir das Kontextproblem aufgefallen, auf das ich bereits in Kapitel 4.3.2.5 eingegangen bin. Aufgrund der langen Trainingszeiten und des hohen Ressourcenverbrauchs habe ich mich dazu entschlossen, das ausgewählte Modell so gut wie möglich zu trainieren. Dafür habe ich mich an dem Paper „How to Fine-Tune BERT for Text Classification?“ (Sun u. a., 2020) orientiert, das sich bereits mit den genannten Problemen beschäftigt hatte.

Zuerst habe ich die Daten weiter normalisiert. Dazu habe ich kurze Texte (< 500 Zeichen) aus der Datenbank gefiltert und die vollständigen Nachrichtenartikel gescraped (über 14.000 Artikel). Außerdem habe ich zwei weitere Labels hinzugefügt: „POLITISCH“ und „UNPOLITISCH“, um unpolitische Texte aus den Trainingsdaten zu entfernen und „noise“ zu reduzieren. Zusätzlich habe ich Algorithmen verwendet, um Werbung, Emojis und irrelevante Inhalte aus den Daten zu entfernen. Zuletzt wurden die Texte in Kleinschreibung umgewandelt, um die Relevanz von Groß- und Kleinschreibung während des Trainings zu reduzieren.

Neben der Normalisierung der Daten wollte ich gleichzeitig auch die Hyperparameter (Stellschrauben, die das Training beeinflussen können) anpassen. Um die richtigen Parameter auszuwählen, entschied ich mich, die besten Parameter mithilfe von Weights&Biases Sweeps zu finden. Dafür legte ich einige Grenzwerte fest (siehe Anhang: Anhang 7). Das Experiment wurde mit 30-mal mit zufällig ausgewählten Werten durchgeführt. Dabei wurde das Modell über wenige Iterationen trainiert. Anschließend habe ich die Ergebnisse mit wandb ausgewertet (Abb. 28). Die besten Hyperparameter (siehe Anhang: 0) habe ich für einen vollständigen Test verwendet.

Die Ergebnisse der Datenbereinigung und Anpassung der Hyperparameter waren mit einem eval/loss von 0,40 (eval/f1: 0,81, recal: 0,83, eval/accuracy: 0,83, eval/precision: 0,83) die bisher besten Ergebnisse.

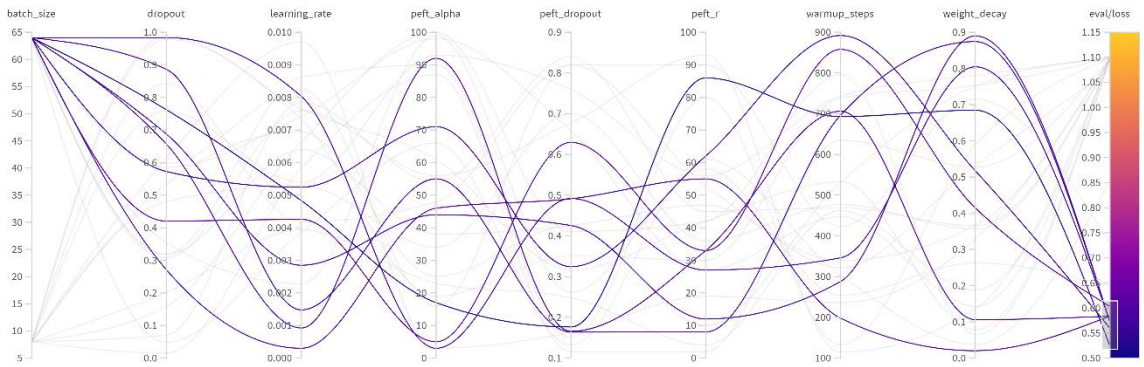


Abb. 28: Hyperparameter Tuning über WandB

4.4.2.5 Gruppen Label

Um herauszufinden, ob bessere Trainingsergebnisse erzielt werden können, wenn die Labels nicht von einer KI generiert werden, habe ich verallgemeinerte Labels erstellt. Dafür erhält ein Verlag ein Label und alle zugehörigen Artikel erben dieses. Für die zugrundeliegenden Labels habe ich mich an eine Einordnung von (polisphere GmbH, 2024) gehalten (Abb. 29) und zusätzliche Inhalte gescraped. Mit diesen Daten konnte ich in 2h 12.000 Labels pro Kategorie (36.000 Samples) erstellen.

Die Einordnung der Verlage in Abb. 29 ist laut polisphere „nur eine Diskussionsgrundlage und keine wissenschaftliche Ausarbeitung, [sie] beruht auf persönlichen Einschätzungen (nicht nur des polisphere-Teams)“ (Anhäuser, 2017). Zudem ist die Grafik bereits sehr alt. Die angeblich veröffentlichte Aktualisierung konnte ich nicht finden. Aus diesen Gründen ist die Einordnung nicht 100 % korrekt und enthält ebenfalls Bias (siehe 2.1.1). Trotzdem ist sie eine gute Grundlage für mein Experiment.

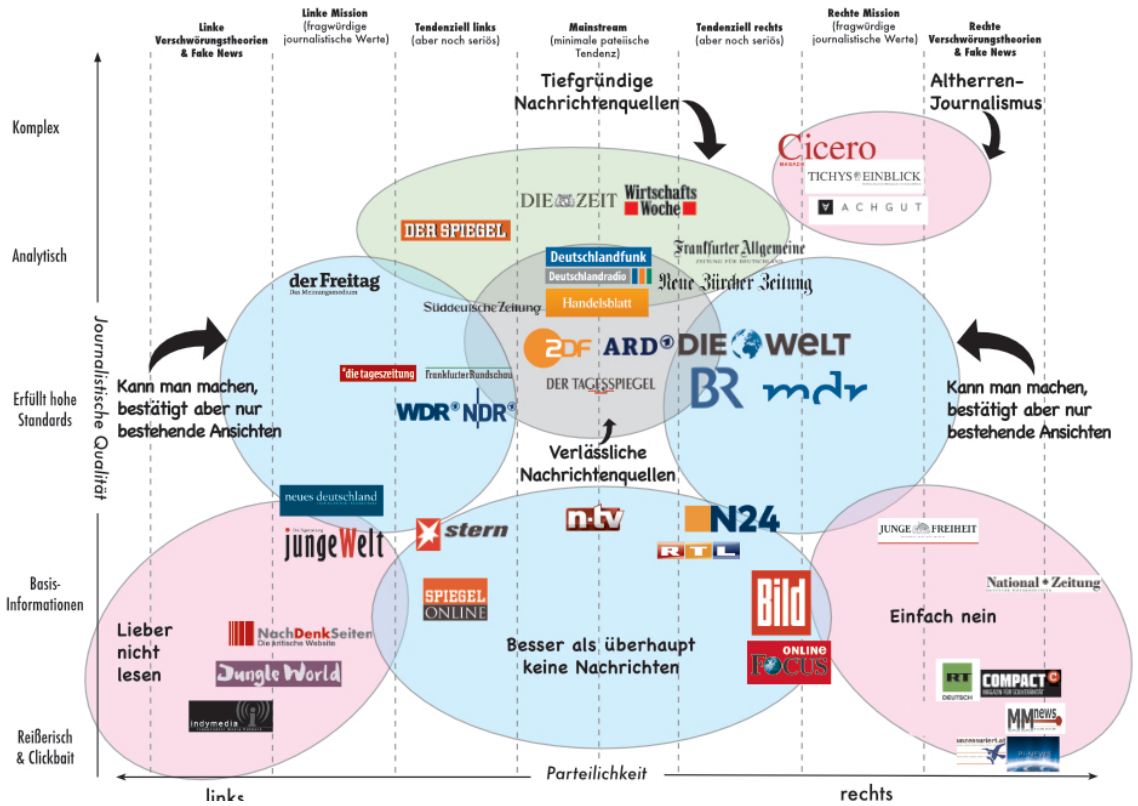


Abb. 29: Politische Einordnung von Medien (Anhäuser, 2017)

Das Ergebnis des Experiments war ein eval/loss von 0,20, eval/accuracy von 0,94, eval/percission von 0,948, eval/recal von 0,94 und eval/f1 von 0,94. Die Ergebnisse waren besser als im Training zuvor. Einerseits kann das Ergebnis mit der größeren Datenmenge zusammenhängen, andererseits kann das Labeling der Daten durch die Generalisierungen einheitlicher sein als die Labeling-KI. Das bewies auch das Training mit 20.100 Sampels, welches ähnliche Ergebnisse erzielte wie das mit 36.000 Sampels (Abb. 30).

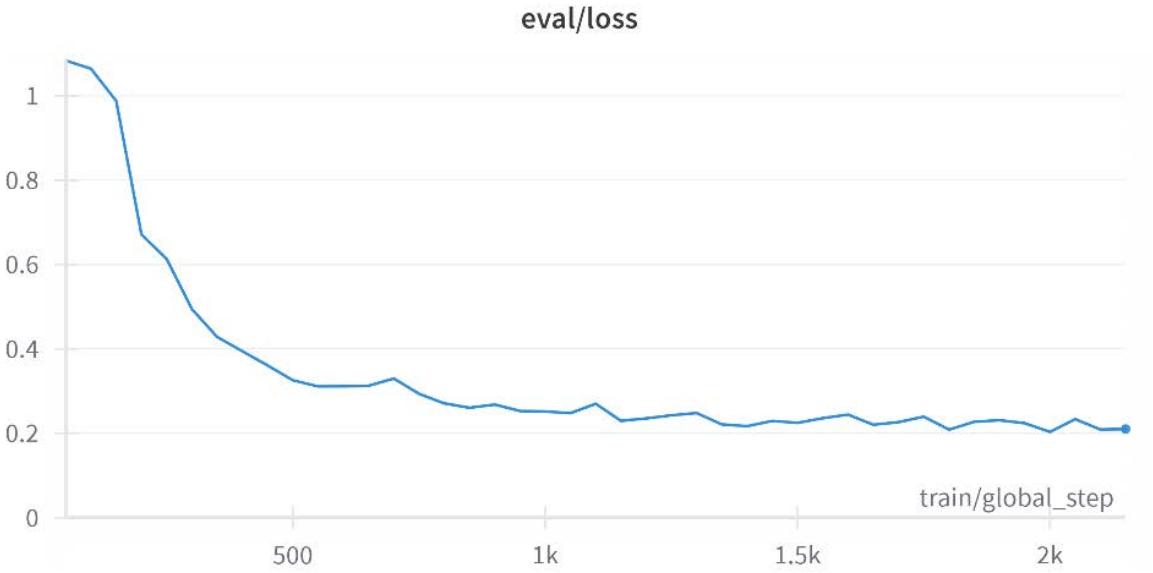


Abb. 30: eval/ loss - Training mit einem manuellen Datensatz (20.100 Samples)

4.5 Entwicklung der Applikation

4.5.1 Umsetzung der Webapplikation

Die Web-App habe ich mittels des T3-Stacks (Browne, 2024) aufgesetzt. Die Installation umfasste NextJS als Basis, NextAuth (Collins u. a., 2024) als Authentifizierungsbibliothek und Prisma (Prisma Data, Inc., 2023) als ORM (Object Relational Mapping) für die Datenbank. Außerdem wurde TypeScript und TailwindCSS (Tailwind Labs Inc., 2024) verwendet. Ich habe Shadcn/ui (shadcn/ui, 2024) als Komponentenbibliothek eingesetzt. Bei der Datenbank wurde weiterhin MongoDB verwendet.

Da sich die Webapplikation vor allem auf die KI fokussiert, habe ich Google News als Vorbild für das schlichte Design verwendet und keine extra Mockups erstellt. Die Web-App wurde einfach aufgebaut. Die aktuellsten Nachrichten werden automatisch aus der Datenbank geholt und in kleinen Kacheln angezeigt. Oben rechts in jeder Kachel befindet sich die durch die KI generierte politische Einordnung. Zusätzlich wird ein Bild, der Verlag sowie ein Ausschnitt des Textes angezeigt. Ein Link führt zum Originalartikel. Ein Klick auf „KI-Rating“ schickt eine Anfrage an OpenAI und generiert ein „Fakenews“-Rating für den Artikel. Der Button „KI-Zusammenfassung“ wird eine Zusammenfassung des Artikels über ein OpenAI-Modell generiert. Neue Nachrichten werden beim Scrollen automatisch durch einen Intersection Observer und Pagination nachgeladen. Die Web-App wird responsiv über alle Gerätegrößen hinweg dargestellt (Abb. 31, Abb. 32). Da die Webanwendung nur eingeschränkt verfügbar sein soll, wurde eine Authentifizierung (oAuth über GitHub) implementiert, die auf ausgewählte

Benutzer beschränkt ist. Für die Abgabe dieser Arbeit wurde zusätzlich ein Credential (E-Mail und Passwort) Login implementiert.

Die Web-App wurde über einen Hetzner VPS (Hetzner Online GmbH, 2024) auf einer Domain bereitgestellt. Dabei zeigt die Subdomain auf die entsprechende IP-Adresse, um die Webseite darzustellen. Zum Aufbau der Webseite auf dem Server wurde Coolify verwendet.

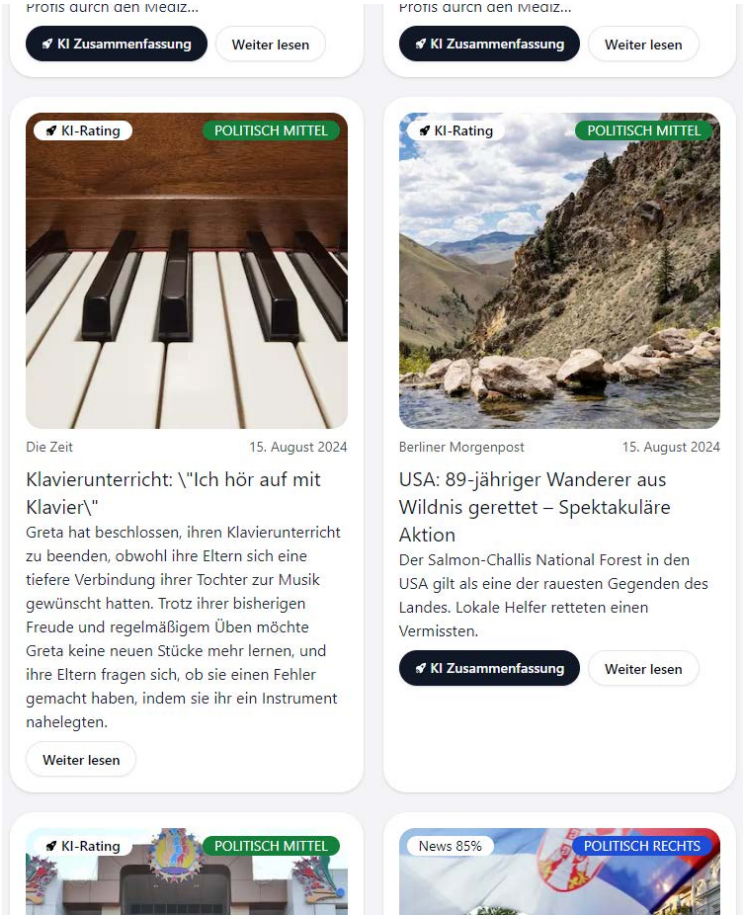


Abb. 31: Unbiased News Web-App (Desktop)

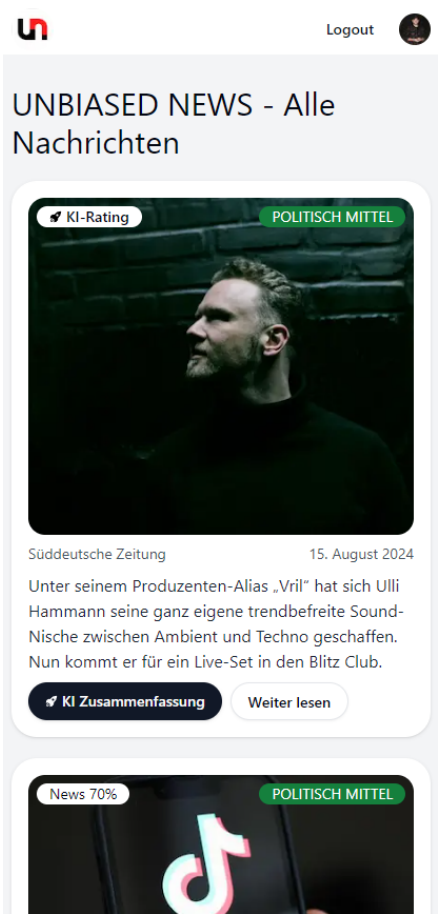


Abb. 32: Unbiased News Web-App (Mobil)

4.5.2 Evaluation der Webapplikation

Um Fehler im Vorfeld zu reduzieren, wurde TypeScript verwendet. In Verbindung mit NextJS und Prisma konnten dadurch Probleme mit der Datenbank und dem Datentransfer vom Backend zum Frontend vermieden werden.

Außerdem habe ich während der Programmierung manuelle Funktionstests durchgeführt, um sicherzustellen, dass alle Inhalte korrekt geladen werden und funktionieren. Ich habe sowohl die funktionalen Bestandteile wie den Login oder das Laden der Artikel als auch die optischen Merkmale auf Ihre Richtigkeit überprüft. Im Falle eines Fehlers konnte ich diesen schnell finden und beheben.

Zuletzt wurden Unit-Tests mit Vitest durchgeführt. Die Tests wurden unabhängig von der Anwendung geschrieben. Ich habe für jede Komponente (z. B. Button) einen Test hinzugefügt (Abb. 33, Abb. 34). Da es sich um eine kleine Anwendung handelt, wurden bei den Tests wenige kleine Fehler entdeckt. Wenn die Anwendung weiterentwickelt wird, können die Tests sehr hilfreich werden. In diesem Fall war vor allem das Schreiben der Tests eine Herausforderung. Primär die Mock Implementierung der NextAuth Server Session-Funktion stellte mich vor einige Probleme. Hierbei konnte der Test nicht zwischen eingeloggt und ausgeloggt unterscheiden. Dieses Problem konnte ich über ein Trennen der Tests in separate Dateien lösen.

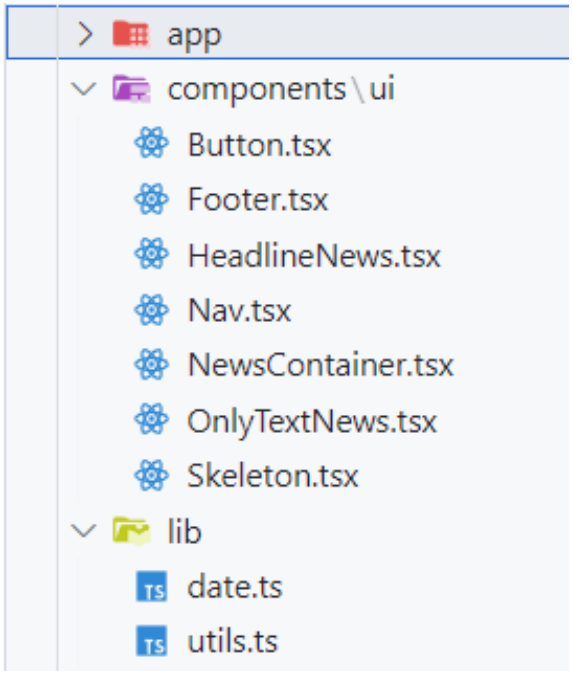


Abb. 33: Komponenten und Funktionen des Projekts

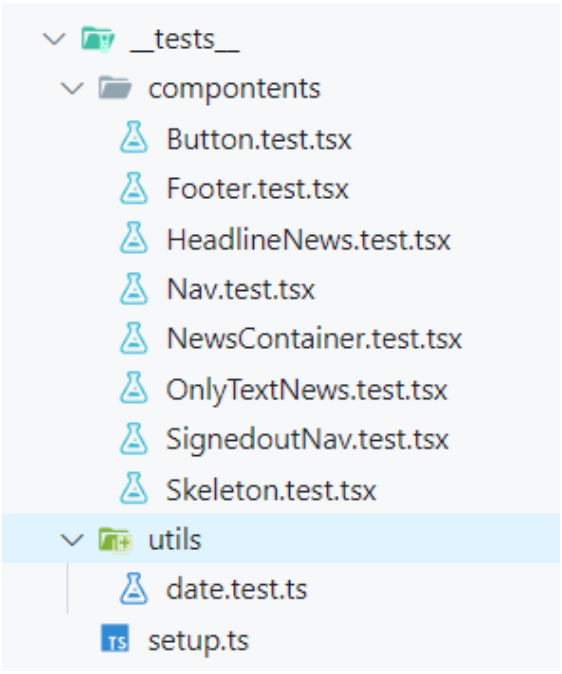


Abb. 34: Tests des Projekts

4.6 Finalisierung

4.6.1 Umsetzung der Finalisierung

In diesem Teil der Arbeit wurden alle Bausteine zusammengesetzt (Abb. 35). Die Automatisierung wird zweimal täglich über einen Cronjob ausgeführt. Alle aufgeführten Schritte (bis auf 6.), sind Funktionen oder Methoden im Python Backend, die nacheinander ausgeführt werden.

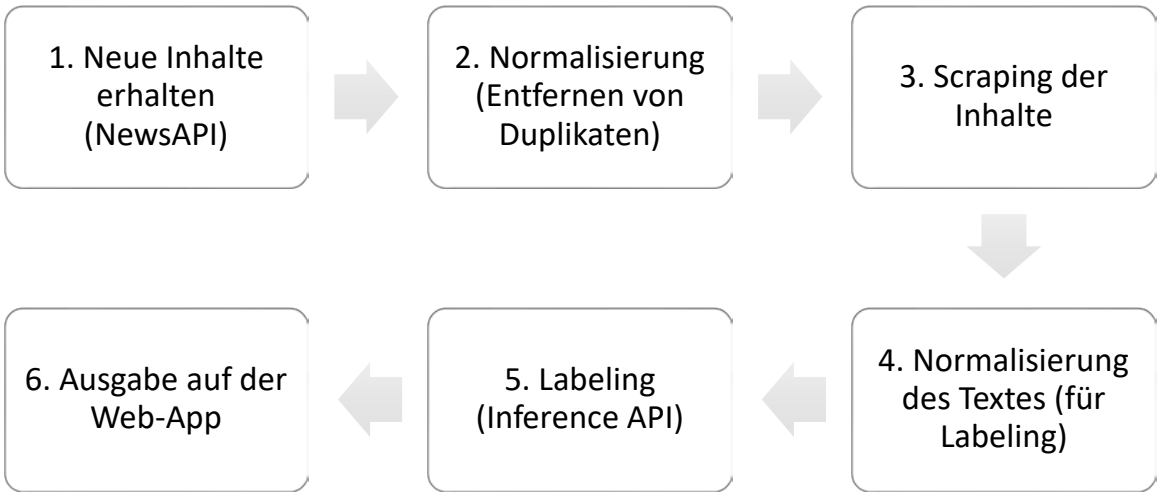


Abb. 35: Ablauf der Automatisierung

4.6.2 Evaluierung der Finalisierung

Während der Umsetzung der Finalisierung musste ich einen Teil des Codes im Python-Backend umstrukturieren und refaktorisieren, damit dieser automatisiert ablaufen konnte. Fehler mussten so abgefangen werden, dass sie die Funktionen nicht unterbrechen (z. B., wenn keine Daten in der Datenbank gefunden werden können). Ein besonderes Beispiel dafür ist der Aufruf der Inference API, die „schläft“ (Scaled to Zero), wenn sie nicht verwendet wird. Dadurch benötigt die API einige Sekunden, um „aufzuwachen“, wenn diese angefragt wird. Um das Problem zu lösen, dass zumindest bei der ersten Anfrage ein Fehler von der API zurückgegeben wird, wird der Fehler abgefangen und die API 30 Sekunden später erneut angefragt (maximal 5-mal). Auf diese Weise kann die API gestartet und verwendet werden.

Diese und ähnliche Probleme habe ich mittels Funktionstests, Unit-Tests, Integration/API-Tests (Abb. 36, orange Pfeile) und End-2-End-Tests überprüft und im Laufe der Arbeit verbessert (Abb. 36).

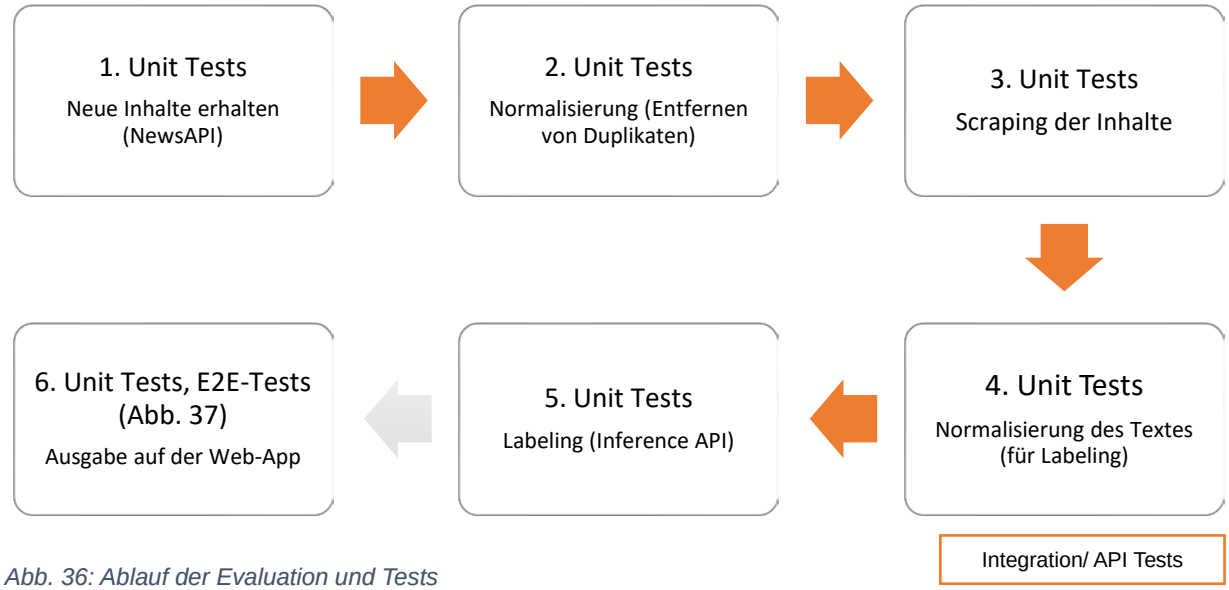


Abb. 36: Ablauf der Evaluation und Tests

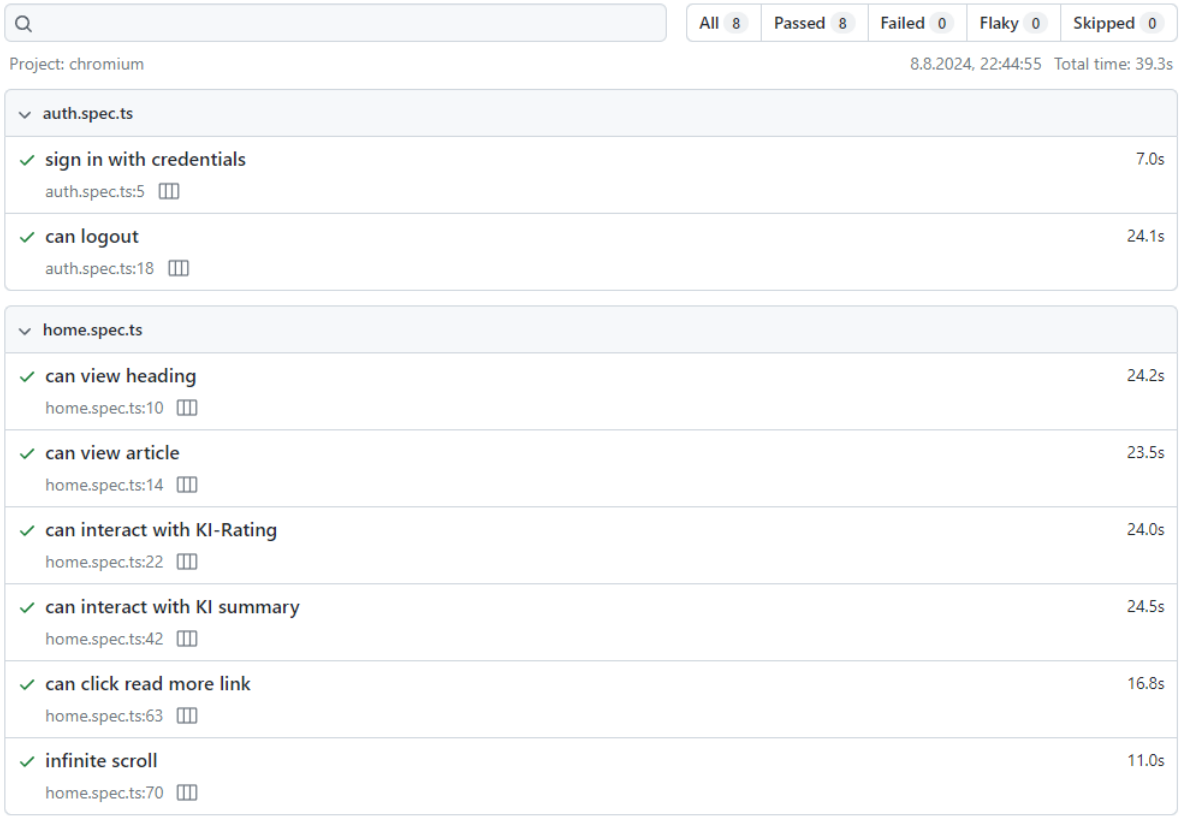


Abb. 37: e2e Tests mit Playwright

5 Ergebnisse, Evaluation und Ausblick

5.1 Ergebnisse

Das fertige Ergebnis des Medienprojekts ist auf der Website https://newsai.danielhilmer.de/ zu finden. Folgend sind alle Informationen zu den veröffentlichten Projektinhalten. Da der Datensatz und das Modell nicht weitergegeben werden darf (durch das Scraping der Daten) sind alle Inhalte nicht öffentlich aufrufbar und mindestens durch eine Authentifizierung gesichert.

5.1.1 Übersicht der Ergebnisse

Tabelle 16: Veröffentlichungen und Ergebnisse des Projekts

Name	**Ort/ URL**
Web-App (Online)	Coolify: https://newsai.danielhilmer.de/
Web-App (Lokal)	Lokal kann die Web-App über die Anleitung im Anhang (Anhang 10) ausgeführt werden. Alle notwendigen Inhalte befinden sich im digitalen Anhang (Anhang 1.3).
Space zum Testen (Manuelle Label)	Hugingface Spaces: DackJan/unbiased-news-manu – es gibt keine Möglichkeit, einen privaten Link zu teilen. Die Inhalte befinden sich im digitalen Anhang (0).
Space zum Testen (AI-Label)	Hugingface Spaces: DackJan/unbiased-news – es gibt keine Möglichkeit, einen privaten Link zu teilen. Die Inhalte befinden sich im digitalen Anhang (0).
Datensatz (Verschiedene Größen, AI-Label und manuelle Labels)	Huggingface: DackJan/news – Es gibt keine Möglichkeit, einen privaten Link zu teilen. Die Inhalte befinden sich im Digitalen Anhang (Anhang 1.4).
Modell Weights mit AI-Labeln	Huggingface: DackJan/distilbert-base-uncased-finetuned-news-bias-de – Es gibt keine Möglichkeit, einen privaten Link zu teilen. Die Inhalte befinden sich im Digitalen Anhang (Anhang 1.5).
Modell Weights mit manuellen Labeln	Huggingface: DackJan/distilbert-base-uncased-finetuned-news-bias-de-manu – es gibt keine

	Möglichkeit, einen privaten Link zu teilen. Die Inhalte befinden sich im digitalen Anhang (Anhang 1.5).
Modell API (manuelle Label)	Dedicated Endpoint: https://u9a0wvmaa4di7jo0.eu-west-1.aws.endpoints.huggingface.cloud
Python Backend (Lokal)	Lokal kann das Python-Backend über die Anleitung im Anhang (Anhang 9) ausgeführt werden. Alle notwendigen Inhalte befinden sich im digitalen Anhang (Anhang 1.2).
Ganzer Code & Entwicklung (Web-App und KI-Modell)	GitHub: https://github.com/dackJaniel/major-project und an den entsprechenden Stellen im digitalen Anhang.

5.1.2 Auswertung der Zielsetzung

Tabelle 17: Auswertung der Zielsetzung

Ziele	Art	Evaluierung
Inhaltliche Ziele		
News-Datenmanagement Automatischer Erhalt von neuen News, Speicherung der News.	Muss	Obwohl dieser Schritt viel mehr Zeit in Anspruch nahm als ursprünglich geplant, konnten im Laufe der Zeit (fast während des gesamten Projekts) viele Daten über die API und später über Scraping in einer Datenbank gespeichert werden. Das Ergebnis sind drei Datensätze. Einer mit 20.100 Artikeln und KI-generierten Labeln und zwei weitere mit manuellen Labeln, mit 20.100 und 36.000 Artikeln. Die Daten enthalten Bias, welche sie während des Trainings an die KI weitergegeben haben. Dieses Problem wurde bereits in 2.1 Bias in künstlichen Intelligenzen angesprochen.
Labeling Labeling der Nachrichtenartikel, damit die KI mit Nachrichten und Labeln (POLITISCH LEICHT LINKS – POLITISCH LINKS – POLITISCH MITTEL – POLITISCH LEICHT	Muss	Die politische Einordnung der Artikel erfolgte schließlich auf zwei verschiedene Arten. Zum einen über das Labeling der Artikel durch die KI und das Framework des Forschungsteams

RECHTS – POLITISCH RECHTS) trainiert werden kann.		und zum anderen über eine Generalisierung. Das Labeling über die KI hat sehr viel Zeit in Anspruch genommen. Die Generalisierung ist skalierbarer, auch wenn dabei nicht mehr jeder einzelne Artikel geprüft und gelabelt wird. Außerdem habe ich mich im Laufe der Zeit dazu entschlossen, nur die Labels POLITISCH LINKS; POLITISCH MITTEL und POLITISCH RECHTS zu bedienen, da in den anderen Bereichen nicht genügend Artikel gefunden werden konnten. Hätte ich die Generalisierung früher durchgeführt, hätte ich auch die Bereiche POLITISCH LEICHT LINKS und POLITISCH LEICHT RECHTS abdecken können.
Fake News Detektion Scannen der Nachrichten auf Fake News, bevor Sie für das Training verwendet werden, Scannen der Nachricht auf Fake News und Tagging/ Kennzeichnung auf der Web-App.	Soll	Nicht vor dem Training, aber auf der Webseite auf Knopfdruck. Der Knopfdruck ist v.a. eine Sparmaßnahme, damit nicht alle Artikel automatisch durch OpenAI verarbeitet werden und Kosten verursachen. Eine Automatisierung wäre problemlos möglich. Die Umsetzung der Fake-News-Detektion vor dem Training war an den Zeitplan gekoppelt und die Umsetzung sollte nur erfolgen, wenn das Labeling rechtzeitig umgesetzt werden konnte. Da dies nicht der Fall war, wurde keine Fake-News-Detektion vor dem Training durchgeführt. Der Fake News Scanner wurde über die API von OpenAI implementiert. Dabei werden die ersten 1000 Zeichen (Sparmaßnahme) des Artikels und der Verlag übergeben. Generative KIs sind für diese Aufgabe nicht optimal geeignet, da diese in der Regel nicht auf aktuelle Inhalte zugreifen können. Für eine produktive Anwendung müsste ein anderes

		System oder eine KI mit Internetzugriff verwendet werden.
Politische Einordnung und Analyse (Inference) Analyse der News und politische Einordnung anhand der verwendeten Sprache.	Muss	Entgegen der Planung wurden zwei Modelle trainiert und zur Verfügung gestellt. Eines, das mit den manuellen Labels trainiert und eines, das mit den KI-generierten Labels trainiert wurde. Für die Inference wurde das Modell mit den manuell gelabelten und verallgemeinerten Daten verwendet, da dieses das bessere Ergebnis lieferte.
Entwicklung der Web-Anwendung **Web-Anwendung zur Bereitstellung der gelabelten Nachrichten.**	Muss	Die Webanwendung konnte erfolgreich implementiert werden. Zusätzlich wurden Hugging Face Spaces zum Testen von eigenen Nachrichtenartikeln und Ausgaben zur Verfügung gestellt. Negativ ist hierbei, dass sich die Hugging Face Spaces nicht durch einen privaten Link teilen lassen.
Zusammenfassungen der News AI generierte Zusammenfassung aus News Artikeln.	Kann	Zusammenfassung über OpenAI per Knopfdruck. Der Knopfdruck ist v.a. eine Sparmaßnahme, damit nicht alle Artikel automatisch durch OpenAI bearbeitet werden und Kosten verursachen. Eine Automatisierung wäre problemlos möglich.
Qualitative Ziele		
Design und Layout nach Vorbild von Google News	Soll	Das Design ist einfach gehalten und ähnelt den Google News.
Optimierung primär auf mobile Geräte	Soll	Die Webanwendung ist für mobile Geräte optimiert, kann aber auch auf größeren Bildschirmen genutzt werden.
Politische Einordnung nach Vorbild von Ground News	Kann	Ground News zeigt die prozentuale politische Einordnung (z. B. 20 % Links, 45 % Mitte, 35 % Rechts) sowie Vergleiche zwischen ähnlichen Artikeln an. Meine News-KI gibt einen Wert zurück, der die politische Einordnung darstellt. Dies ist die

		prozentuale Wahrscheinlichkeit, dass dieser Artikel zu einer politischen Kategorie gehört. Die Überprüfung, wie viel Prozent des Textes eine politische Einordnung hat, wird aus Zeitgründen nicht durchgeführt.
Zusammenfassung der Artikel nach Vorbild von Ground News	Kann	Zusammenfassung über OpenAI per Knopfdruck. Die Nachrichten von Ground News werden in 3 Stichpunkten zusammengefasst. In meiner Version werden die Nachrichten in 2 Sätzen zusammengefasst.
Mindestens 75 % accuracy = < 25 % loss	Muss	Das von der KI gelabelte Modell erreichte ein eval/loss von 0,4 und ein eval/accuracy von 0,8. Das manuell gelabelte Modell erreichte ein eval/loss von 0,2 und ein eval/accuracy von über 0,9.
Quantitative Ziele		
Deutsche News	Muss	Für das Training wurden nur deutschsprachige Nachrichten verwendet. Andere Sprachen, aber auch Werbung, wurden aus den Daten herausgefiltert.
Training/ Finetuning der KI mit ~ 5.000 – 50.000 Datensätzen	Muss	Das abschließende Training wurde mit 20.100 Artikeln (KI-Label) und 36.000 Artikeln (manuelles Label) durchgeführt. Diese Werte stimmen sehr gut mit den ursprünglichen Schätzungen überein.

Legende: Grün: Vollständig erfüllt, Orange: Teilweise erfüllt, Rot: Nicht erfüllt

5.2 Evaluation

Ich habe das Projekt iterativ durchgeführt und während der Durchführung recherchiert. Das hatte den Vorteil, dass ich genau wusste, welche Probleme zu lösen waren und welche Informationen für mich relevant waren. Andererseits stößt man bei dieser Vorgehensweise oft auf Situationen, die noch nicht gelöst werden können. Daher musste ich viel Zeit aufwenden, um diese Situationen durch Ausprobieren zu lösen.

Unter diesen Voraussetzungen verlief das Projekt größtenteils gut. Die größten Probleme traten bei den Daten und dem Labeling auf. Das waren auch die Bereiche, in denen ich noch

keine Erfahrung hatte. Die Daten hätten z. B. früher bereinigt werden müssen und auch das Labeling hätte ich im Nachhinein wahrscheinlich nicht mit einer KI durchgeführt. Stattdessen ist das Clustern von Daten, wie ich später herausfand, eine skalierbare Lösung. Auch wenn sie andere Nachteile hat. Die Vermutung liegt nahe, dass die KI-Labels nicht konsistent waren. Um das mit Sicherheit zu wissen, müssen jedoch weitere Untersuchungen durchgeführt werden. Es stellte sich heraus, dass die Datenbasis für das Training einer der entscheidenden Punkte für das Trainingsergebnis ist.

Die Implementierung der Webanwendung (die nur einen kleinen Teil der Arbeit ausmachte), aber auch das Fine-Tuning verliefen hingegen gut. Das einmal geschriebene Skript konnte immer wieder verwendet werden. Erstaunlich war für mich die kurze Trainingszeit, die je nach GPU zwischen 10 und 60 Minuten lag. Dies ist sicherlich auf das gewählte Modell zurückzuführen. Es war recht klein und konnte bereits auf einer GTX 3090 mit 24 GB VRAM schnell trainiert werden. Die Inference läuft auf einer CPU (Intel Sapphire Rapids) mit 2 GB RAM. Bei einem Projekt mit mehr Zeit und größeren finanziellen Mitteln würde ich trotzdem auf ein größeres Modell (z. B. llama-3.1-8B oder llama-3.1-70B) zurückgreifen, um mehr Kontext des Artikels analysieren zu können und die Fehlerquote noch weiter zu reduzieren.

Für mich persönlich war das Projekt sehr erfolgreich. Ich habe sehr viel gelernt und konnte auftretende Probleme lösen. Dies habe ich sowohl durch viel Recherche als auch durch die regelmäßigen Feedbackgespräche erreicht. Dabei konnte ich wertvolle Informationen erhalten und umsetzen. Tipps und Hilfestellungen habe ich auch von einem Mitglied des Forschungsteams (Media Bias Group) erhalten. Manchmal konnten auch Freunde, die die richtigen Fragen stellten, einen Anstoß in die richtige Richtung geben.

Ich habe gemerkt, dass ich den Zeitplan nicht einhalten konnte. Durch verschiedene Verzögerungen war der Zeitplan am Anfang noch brauchbar, aber im Laufe des Projektes nicht mehr. Deshalb habe ich ihn als Orientierungshilfe gesehen. Durch das iterative Vorgehen (auch mit viel Puffer) konnte ich die Termine in den meisten Fällen nicht einhalten. Festzuhalten ist auch, dass vor allem die Zeiten für das Labeling und die Auswahl des Basismodells (Modellvergleich) nicht realistisch waren. Gerade auf diese Bereiche würde ich bei zukünftigen Projekten mehr Wert legen und mindestens 50 % mehr Zeit für die Generierung der Datenbasis einplanen.

Meine Motivation konnte ich während der vielen Arbeitsschritte gut aufrechterhalten. Regelmäßige Fortschritte haben diese unterstützt und gefördert. Es gab aber auch Phasen, in denen kein Fortschritt erkennbar war. In diesen Zeiten fiel es mir oft schwer, mich nicht von einfacheren Arbeiten ablenken zu lassen. Auszeiten, z. B. Unternehmungen und Sport, haben mir geholfen, den Kopf wieder freizubekommen und effektiv am Projekt weiterzuarbeiten.

5.3 Ausblick

Aufgrund der verwendeten Daten (Scraping) kann weder der Datensatz noch das Modell oder die Webseite veröffentlicht werden. Dies würde wahrscheinlich zu Urheberrechtsverletzungen führen. Außerdem ist mein Vertrauen in die Ausgaben des Modells aufgrund der Datenbasis sehr begrenzt. Dennoch halte ich das Thema in der heutigen Zeit, die nicht nur durch menschliche, sondern auch durch KI-generierte Fehlinformationen und Verzerrungen geprägt ist, für äußerst relevant. Um die Idee dieser Webapplikation umzusetzen, braucht es neben der rechtlichen Absicherung auch entsprechende Mittel, um die KI mit einem großen Modell zu trainieren und weitere Funktionalitäten wie Fake-News-Scanner, Artikelvergleiche und ähnliche Features bereitzustellen. Der wichtigste Baustein ist zudem der zugrundeliegende Datensatz, auf dem das Modell trainiert wird. Die Labels müssen möglichst neutral auf die Daten angewendet werden. Dazu müssen mehrere Prüfinstanzen wie externe Prüfer, Anwender und prüfende KIs eingesetzt werden, um den Bias im Datensatz und in der KI so gering wie möglich zu halten. Zudem sollten die Labels erweitert werden, um die Komplexität der Realität widerzuspiegeln, die nicht nur schwarz und weiß ist. Dies ist mit Kosten verbunden, die gedeckt werden müssen.

Unter diesen Voraussetzungen ist es schwierig, dieses Projekt in der Zukunft erfolgreich umzusetzen. Mit der richtigen Organisation und dem richtigen Team ist es jedoch möglich.

Quellenverzeichnis

AllSides, 2019. *AllSides | Balanced news via media bias ratings for an unbiased news perspective*. [online] AllSides. Verfügbar unter: <https://www.allsides.com/unbiased-balanced-news> [Zugegriffen 8 Oktober 2023].

Anhäuser, M., 2017. *Welche Medien stehen wo, welche sind verlässlich? (Nachtrag August 2022)*. [online] Plazeboalarm. Verfügbar unter: <https://scienceblogs.de/plazeboalarm/index.php/welche-medien-stehen-wo-welche-sind-verlaesslich/> [Zugegriffen 12 Mai 2024].

Anil, R., Borgeaud, S., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A.M., Hauth, A., Millican, K., Silver, D., Johnson, M., Antonoglou, I., Schrittwieser, J., Glaese, A., Chen, J., Pitler, E., Lillicrap, T., Lazaridou, A., Firat, O., Molloy, J., Isard, M., Barham, P.R., Hennigan, T., Lee, B., Viola, F., Reynolds, M., Xu, Y., Doherty, R., Collins, E., Meyer, C., Rutherford, E., Moreira, E., Ayoub, K., Goel, M., Krawczyk, J., Du, C., Chi, E., Cheng, H.-T., Ni, E., Shah, P., Kane, P., Chan, B., Faruqui, M., Severyn, A., Lin, H., Li, Y., Cheng, Y., Ittycheriah, A., Mahdieh, M., Chen, M., Sun, P., Tran, D., Bagri, S., Lakshminarayanan, B., Liu, J., Orban, A., Güra, F., Zhou, H., Song, X., Boffy, A., Ganapathy, H., Zheng, S., Choe, H., Weisz, Á., Zhu, T., Lu, Y., Gopal, S., Kahn, J., Kula, M., Pitman, J., Shah, R., Taropa, E., Merey, M.A., Baeuml, M., Chen, Z., Shafey, L.E., Zhang, Y., Sercinoglu, O., Tucker, G., Piqueras, E., Krikun, M., Barr, I., Savinov, N., Danihelka, I., Roelofs, B., White, A., Andreassen, A., von Glehn, T., Yagati, L., Kazemi, M., Gonzalez, L., Khalman, M., Sygnowski, J., Frechette, A., Smith, C., Culp, L., Proleev, L., Luan, Y., Chen, X., Lottes, J., Schucher, N., Lebron, F., Rrustemi, A., Clay, N., Crone, P., Kocisky, T., Zhao, J., Perz, B., Yu, D., Howard, H., Bloniarz, A., Rae, J.W., Lu, H., Sifre, L., Maggioni, M., Alcober, F., Garrette, D., Barnes, M., Thakoor, S., Austin, J., Barth-Maron, G., Wong, W., Joshi, R., Chaabouni, R., Fatiha, D., Ahuja, A., Tomar, G.S., Senter, E., Chadwick, M., Kornakov, I., Attaluri, N., Iturrate, I., Liu, R., Li, Y., Cogan, S., Chen, J., Jia, C., Gu, C., Zhang, Q., Grimstad, J., Hartman, A.J., Garcia, X., Pillai, T.S., Devlin, J., Laskin, M., Casas, D. de L., Valter, D., Tao, C., Blanco, L., Badia, A.P., Reitter, D., Chen, M., Brennan, J., Rivera, C., Brin, S., Iqbal, S., Surita, G., Labanowski, J., Rao, A., Winkler, S., Parisotto, E., Gu, Y., Olszewska, K., Addanki, R., Miech, A., Louis, A., Teplyashin, D., Brown, G., Catt, E., Balaguer, J., Xiang, J., Wang, P., Ashwood, Z., Briukhov, A., Webson, A., Ganapathy, S., Sanghavi, S., Kannan, A., Chang, M.-W., Stjerngren, A., Djolonga, J., Sun, Y., Bapna, A., Aitchison, M., Pejman, P., Michalewski, H., Yu, T., Wang, C., Love, J., Ahn, J., Bloxwich, D., Han, K., Humphreys, P., Sellam, T., Bradbury, J., Godbole, V., Samangooei, S., Damoc, B., Kaskasoli, A., Arnold, S.M.R., Vasudevan, V., Agrawal, S., Riesa, J., Lepikhin, D., Tanburn, R., Srinivasan, S., Lim, H., Hodkinson, S., Shyam, P., Ferret, J., Hand, S., Garg, A., Paine, T.L., Li, J., Li, Y., Giang, M., Neitz, A., Abbas, Z., York, S., Reid, M., Cole, E., Chowdhery, A., Das, D., Rogozińska, D., Nikolaev, V., Sprechmann, P., Nado, Z., Zilka, L., Prost, F., He, L., Monteiro, M., Mishra, G., Welty, C., Newlan, J., Jia, D., Allamanis, M., Hu, C.H., de Liedekerke, R., Gilmer, J., Saroufim, C., Rijhwani, S., Hou, S., Shrivastava, D., Baddepudi, A., Goldin, A., Ozturel, A., Cassirer, A., Xu, Y., Sohn, D., Sachan, D., Amplayo, R.K., Swanson, C., Petrova, D., Narayan, S., Guez, A., Brahma, S., Landon, J., Patel, M., Zhao, R., Villela, K., Wang, L., Jia, W., Rahtz, M., Giménez, M., Yeung, L., Keeling, J., Georgiev, P., Mincu, D., Wu, B., Haykal, S., Saputro, R., Vodrahalli, K., Qin, J., Cankara, Z., Sharma, A., Fernando, N., Hawkins, W., Neyshabur, B., Kim, S., Hutter, A., Agrawal, P., Castro-Ros, A., Driessche, G. van den, Wang, T., Yang, F., Chang, S., Komarek, P., McIlroy, R., Lučić, M., Zhang, G., Farhan, W., Sharman, M., Natsev, P., Michel, P., Bansal, Y., Qiao, S., Cao, K., Shakeri, S., Butterfield, C., Chung, J., Rubenstein, P.K., Agrawal, S., Mensch, A., Soparkar, K., Lenc, K., Chung, T., Pope, A., Maggiore, L., Kay, J., Jhakra, P., Wang, S., Maynez, J., Phuong, M., Tobin, T., Tacchetti, A., Trebacz, M., Robinson, K., Katariya, Y., Riedel, S., Bailey, P., Xiao, K., Ghelani, N., Aroyo, L., Slone, A., Houlsby, N., Xiong, X., Yang, Z., Gribovskaya, E., Adler, J., Wirth, M., Lee, L., Li, M., Kagohara, T., Pavagadhi, J., Bridgers, S., Bortsova, A., Ghemawat, S., Ahmed, Z., Liu, T., Powell, R., Bolina, V., Iinuma, M., Zablotskaia, P., Besley, J., Chung, D.-W., Dozat, T., Comanescu, R., Si, X., Greer, J., Su, G., Polacek, M., Kaufman, R.L., Tokumine, S., Hu, H., Buchatskaya, E., Miao,

Y., Elhawaty, M., Siddhant, A., Tomasev, N., Xing, J., Greer, C., Miller, H., Ashraf, S., Roy, A., Zhang, Z., Ma, A., Filos, A., Besta, M., Blevins, R., Klimenko, T., Yeh, C.-K., Changpinyo, S., Mu, J., Chang, O., Pajarskas, M., Muir, C., Cohen, V., Lan, C.L., Haridasan, K., Marathe, A., Hansen, S., Douglas, S., Samuel, R., Wang, M., Austin, S., Lan, C., Jiang, J., Chiu, J., Lorenzo, J.A., Sjösund, L.L., Cevey, S., Gleicher, Z., Avrahami, T., Boral, A., Srinivasan, H., Selo, V., May, R., Aisopos, K., Hussenot, L., Soares, L.B., Baumli, K., Chang, M.B., Recasens, A., Caine, B., Pritzel, A., Pavetic, F., Pardo, F., Gergely, A., Frye, J., Ramasesh, V., Horgan, D., Badola, K., Kassner, N., Roy, S., Dyer, E., Campos, V.C., Tomala, A., Tang, Y., Badawy, D.E., White, E., Mustafa, B., Lang, O., Jindal, A., Vikram, S., Gong, Z., Caelles, S., Hemsley, R., Thornton, G., Feng, F., Stokowiec, W., Zheng, C., Thacker, P., Ünlü, Ç., Zhang, Z., Saleh, M., Svensson, J., Bileschi, M., Patil, P., Anand, A., Ring, R., Tsihlas, K., Vezer, A., Selvi, M., Shevlane, T., Rodriguez, M., Kwiatkowski, T., Daruki, S., Rong, K., Dafoe, A., FitzGerald, N., Gu-Lemberg, K., Khan, M., Hendricks, L.A., Pellat, M., Feinberg, V., Cobon-Kerr, J., Sainath, T., Rauh, M., Hashemi, S.H., Ives, R., Hasson, Y., Noland, E., Cao, Y., Byrd, N., Hou, L., Wang, Q., Sottiaux, T., Paganini, M., Lespiau, J.-B., Moufarek, A., Hassan, S., Shivakumar, K., van Amersfoort, J., Mandhane, A., Joshi, P., Goyal, A., Tung, M., Brock, A., Sheahan, H., Misra, V., Li, C., Rakićević, N., Dehghani, M., Liu, F., Mittal, S., Oh, J., Noury, S., Sezener, E., Huot, F., Lamm, M., De Cao, N., Chen, C., Mudgal, S., Stella, R., Brooks, K., Vasudevan, G., Liu, C., Chain, M., Melinkeri, N., Cohen, A., Wang, V., Seymore, K., Zubkov, S., Goel, R., Yue, S., Krishnakumaran, S., Albert, B., Hurley, N., Sano, M., Mohananey, A., Joughin, J., Filonov, E., Kępa, T., Eldawy, Y., Lim, J., Rishi, R., Badiezadegan, S., Bos, T., Chang, J., Jain, S., Padmanabhan, S.G.S., Puttagunta, S., Krishna, K., Baker, L., Kalb, N., Bedapudi, V., Kurzrok, A., Lei, S., Yu, A., Litvin, O., Zhou, X., Wu, Z., Sobell, S., Siciliano, A., Papir, A., Neale, R., Bragagnolo, J., Toor, T., Chen, T., Anklin, V., Wang, F., Feng, R., Gholami, M., Ling, K., Liu, L., Walter, J., Moghaddam, H., Kishore, A., Adamek, J., Mercado, T., Mallinson, J., Wandekar, S., Cagle, S., Ofek, E., Garrido, G., Lombriser, C., Mukha, M., Sun, B., Mohammad, H.R., Matak, J., Qian, Y., Peswani, V., Janus, P., Yuan, Q., Schelin, L., David, O., Garg, A., He, Y., Duzhyi, O., Älgmyr, A., Lottaz, T., Li, Q., Yadav, V., Xu, L., Chinien, A., Shivanna, R., Chuklin, A., Li, J., Spadine, C., Wolfe, T., Mohamed, K., Das, S., Dai, Z., He, K., von Dincklage, D., Upadhyay, S., Maurya, A., Chi, L., Krause, S., Salama, K., Rabinovitch, P.G., M, P.K.R., Selvan, A., Dektiarev, M., Ghiasi, G., Guven, E., Gupta, H., Liu, B., Sharma, D., Shtacher, I.H., Paul, S., Akerlund, O., Aubet, F.-X., Huang, T., Zhu, C., Zhu, E., Teixeira, E., Fritze, M., Bertolini, F., Marinescu, L.-E., Bölle, M., Paulus, D., Gupta, K., Latkar, T., Chang, M., Sanders, J., Wilson, R., Wu, X., Tan, Y.-X., Thiet, L.N., Doshi, T., Lall, S., Mishra, S., Chen, W., Luong, T., Benjamin, S., Lee, J., Andrejczuk, E., Rabiej, D., Ranjan, V., Styrc, K., Yin, P., Simon, J., Harriott, M.R., Bansal, M., Robsky, A., Bacon, G., Greene, D., Mirylenka, D., Zhou, C., Sarvana, O., Goyal, A., Andermatt, S., Siegler, P., Horn, B., Israel, A., Pongetti, F., Chen, C.-W. „Louis“, Selvatici, M., Silva, P., Wang, K., Tolins, J., Guu, K., Yogev, R., Cai, X., Agostini, A., Shah, M., Nguyen, H., Donnaile, N.Ó., Pereira, S., Friso, L., Stambler, A., Kurzrok, A., Kuang, C., Romanikhin, Y., Geller, M., Yan, Z.J., Jang, K., Lee, C.-C., Fica, W., Malmi, E., Tan, Q., Banica, D., Balle, D., Pham, R., Huang, Y., Avram, D., Shi, H., Singh, J., Hidey, C., Ahuja, N., Saxena, P., Dooley, D., Potharaju, S.P., O’Neill, E., Gokulchandran, A., Foley, R., Zhao, K., Dusenberry, M., Liu, Y., Mehta, P., Kotikalapudi, R., Safranek-Shrader, C., Goodman, A., Kessinger, J., Globen, E., Kolhar, P., Gorgolewski, C., Ibrahim, A., Song, Y., Eichenbaum, A., Brovelli, T., Potluri, S., Lahoti, P., Baetu, C., Ghorbani, A., Chen, C., Crawford, A., Pal, S., Sridhar, M., Gurita, P., Mujika, A., Petrovski, I., Cedoz, P.-L., Li, C., Chen, S., Santo, N.D., Goyal, S., Punjabi, J., Kappaganthu, K., Kwak, C., LV, P., Velury, S., Choudhury, H., Hall, J., Shah, P., Figueira, R., Thomas, M., Lu, M., Zhou, T., Kumar, C., Jurdi, T., Chikkerur, S., Ma, Y., Yu, A., Kwak, S., Ähdel, V., Rajayogam, S., Choma, T., Liu, F., Barua, A., Ji, C., Park, J.H., Hellendoorn, V., Bailey, A., Bilal, T., Zhou, H., Khatir, M., Sutton, C., Rzadkowski, W., Macintosh, F., Shagin, K., Medina, P., Liang, C., Zhou, J., Shah, P., Bi, Y., Dankovics, A., Banga, S., Lehmann, S., Bredesen, M., Lin, Z., Hoffmann, J.E., Lai, J., Chung, R., Yang, K., Balani, N., Bražinskas, A., Sozanschi, A., Hayes, M., Alcalde, H.F., Makarov, P., Chen, W., Stella, A., Snijders, L., Mandl, M., Kärrman, A., Nowak, P., Wu, X., Dyck, A., Vaidyanathan, K., R, R., Mallet, J., Rudominer, M., Johnston, E., Mittal, S., Udathu, A., Christensen, J., Verma, V., Irving, Z., Santucci, A., Elsayed, G., Davoodi, E., Georgiev, M., Tenney, I., Hua, N., Cideron, G., Leurent, E., Alnahlawi,

M., Georgescu, I., Wei, N., Zheng, I., Scandinaro, D., Jiang, H., Snoek, J., Sundararajan, M., Wang, X., Ontiveros, Z., Karo, I., Cole, J., Rajashekhar, V., Tumeh, L., Ben-David, E., Jain, R., Uesato, J., Datta, R., Bunyan, O., Wu, S., Zhang, J., Stanczyk, P., Zhang, Y., Steiner, D., Naskar, S., Azzam, M., Johnson, M., Paszke, A., Chiu, C.-C., Elias, J.S., Mohiuddin, A., Muhammad, F., Miao, J., Lee, A., Vieillard, N., Park, J., Zhang, J., Stanway, J., Garmon, D., Karmarkar, A., Dong, Z., Lee, J., Kumar, A., Zhou, L., Evens, J., Isaac, W., Irving, G., Loper, E., Fink, M., Arkatkar, I., Chen, N., Shafran, I., Petrychenko, I., Chen, Z., Jia, J., Levskaya, A., Zhu, Z., Grabowski, P., Mao, Y., Magni, A., Yao, K., Snaider, J., Casagrande, N., Palmer, E., Suganthan, P., Castaño, A., Giannoumis, I., Kim, W., Rybiński, M., Sreevatsa, A., Prendki, J., Soergel, D., Goedeckemeyer, A., Gierke, W., Jafari, M., Gaba, M., Wiesner, J., Wright, D.G., Wei, Y., Vashisht, H., Kulizhskaya, Y., Hoover, J., Le, M., Li, L., Iwuanyanwu, C., Liu, L., Ramirez, K., Khorlin, A., Cui, A., LIN, T., Wu, M., Aguilar, R., Pallo, K., Chakladar, A., Perng, G., Abellan, E.A., Zhang, M., Dasgupta, I., Kushman, N., Penchev, I., Repina, A., Wu, X., van der Weide, T., Ponnapalli, P., Kaplan, C., Simsa, J., Li, S., Dousse, O., Yang, F., Piper, J., Ie, N., Pasumarthi, R., Lintz, N., Vijayakumar, A., Andor, D., Valenzuela, P., Lui, M., Paduraru, C., Peng, D., Lee, K., Zhang, S., Greene, S., Nguyen, D.D., Kurylowicz, P., Hardin, C., Dixon, L., Janzer, L., Choo, K., Feng, Z., Zhang, B., Singhal, A., Du, D., McKinnon, D., Antropova, N., Bolukbasi, T., Keller, O., Reid, D., Finchelstein, D., Raad, M.A., Crocker, R., Hawkins, P., Dadashi, R., Gaffney, C., Franko, K., Bulanova, A., Leblond, R., Chung, S., Askham, H., Cobo, L.C., Xu, K., Fischer, F., Xu, J., Sorokin, C., Alberti, C., Lin, C.-C., Evans, C., Dimitriev, A., Forbes, H., Banarse, D., Tung, Z., Omernick, M., Bishop, C., Sterneck, R., Jain, R., Xia, J., Amid, E., Piccinno, F., Wang, X., Banzal, P., Mankowitz, D.J., Polozov, A., Krakovna, V., Brown, S., Bateni, M., Duan, D., Firoiu, V., Thotakuri, M., Natan, T., Geist, M., Girgin, S. tan, Li, H., Ye, J., Roval, O., Tojo, R., Kwong, M., Lee-Thorp, J., Yew, C., Sinopalnikov, D., Ramos, S., Mellor, J., Sharma, A., Wu, K., Miller, D., Sonnerat, N., Vnukov, D., Greig, R., Beattie, J., Caveness, E., Bai, L., Eisenschlos, J., Korchemniy, A., Tsai, T., Jasarevic, M., Kong, W., Dao, P., Zheng, Z., Liu, F., Yang, F., Zhu, R., Teh, T.H., Sanmiya, J., Gladchenko, E., Trdin, N., Toyama, D., Rosen, E., Tavakkol, S., Xue, L., Elkind, C., Woodman, O., Carpenter, J., Papamakarios, G., Kemp, R., Kafle, S., Grunina, T., Sinha, R., Talbert, A., Wu, D., Owusu-Afriyie, D., Du, C., Thornton, C., Pont-Tuset, J., Narayana, P., Li, J., Fatehi, S., Wieting, J., Ajmeri, O., Uria, B., Ko, Y., Knight, L., Héliou, A., Niu, N., Gu, S., Pang, C., Li, Y., Levine, N., Stolovich, A., Santamaria-Fernandez, R., Goenka, S., Yustalim, W., Strudel, R., Elqursh, A., Deck, C., Lee, H., Li, Z., Levin, K., Hoffmann, R., Holtmann-Rice, D., Bachem, O., Arora, S., Koh, C., Yeganeh, S.H., Põder, S., Tariq, M., Sun, Y., Ionita, L., Seyedhosseini, M., Tafti, P., Liu, Z., Gulati, A., Liu, J., Ye, X., Chrzaszcz, B., Wang, L., Sethi, N., Li, T., Brown, B., Singh, S., Fan, W., Parisi, A., Stanton, J., Koverkathu, V., Choquette-Choo, C.A., Li, Y., Lu, T.J., Ittycheriah, A., Shroff, P., Varadarajan, M., Bahargam, S., Willoughby, R., Gaddy, D., Desjardins, G., Cornero, M., Robenek, B., Mittal, B., Albrecht, B., Shenoy, A., Moiseev, F., Jacobsson, H., Ghaffarkhah, A., Rivière, M., Walton, A., Crepy, C., Parrish, A., Zhou, Z., Farabet, C., Radebaugh, C., Srinivasan, P., van der Salm, C., Fidjeland, A., Scellato, S., Latorre-Chimoto, E., Klimczak-Plucińska, H., Bridson, D., de Cesare, D., Hudson, T., Mendolicchio, P., Walker, L., Morris, A., Mauger, M., Guseynov, A., Reid, A., Odoom, S., Loher, L., Cotruta, V., Yenugula, M., Grewe, D., Petrushkina, A., Duerig, T., Sanchez, A., Yadlowsky, S., Shen, A., Globerson, A., Webb, L., Dua, S., Li, D., Bhupatiraju, S., Hurt, D., Qureshi, H., Agarwal, A., Shani, T., Eyal, M., Khare, A., Belle, S.R., Wang, L., Tekur, C., Kale, M.S., Wei, J., Sang, R., Saeta, B., Liechty, T., Sun, Y., Zhao, Y., Lee, S., Nayak, P., Fritz, D., Vuyyuru, M.R., Aslanides, J., Vyas, N., Wicke, M., Ma, X., Eltyshev, E., Martin, N., Cate, H., Manyika, J., Amiri, K., Kim, Y., Xiong, X., Kang, K., Luisier, F., Tripuraneni, N., Madras, D., Guo, M., Waters, A., Wang, O., Ainslie, J., Baldridge, J., Zhang, H., Pruthi, G., Bauer, J., Yang, F., Mansour, R., Gelman, J., Xu, Y., Polovets, G., Liu, J., Cai, H., Chen, W., Sheng, X., Xue, E., Ozair, S., Angermueller, C., Li, X., Sinha, A., Wang, W., Wiesinger, J., Koukoumidis, E., Tian, Y., Iyer, A., Gurumurthy, M., Goldenson, M., Shah, P., Blake, M.K., Yu, H., Urbanowicz, A., Palomaki, J., Fernando, C., Durden, K., Mehta, H., Momchev, N., Rahimtoroghi, E., Georgaki, M., Raul, A., Ruder, S., Redshaw, M., Lee, J., Zhou, D., Jalan, K., Li, D., Hechtman, B., Schuh, P., Nasr, M., Milan, K., Mikulik, V., Franco, J., Green, T., Nguyen, N., Kelley, J., Mahendru, A., Hu, A., Howland, J., Vargas, B., Hui, J., Bansal, K., Rao, V., Ghiya, R., Wang, E., Ye, K., Sarr, J.M., Preston, M.M., Elish, M., Li, S., Kaku, A.,

Gupta, J., Pasupat, I., Juan, D.-C., Someswar, M., M., T., Chen, X., Amini, A., Fabrikant, A., Chu, E., Dong, X., Muthal, A., Buthpitiya, S., Jauhari, S., Hua, N., Khandelwal, U., Hitron, A., Ren, J., Rinaldi, L., Drath, S., Dabush, A., Jiang, N.-J., Godhia, H., Sachs, U., Chen, A., Fan, Y., Taitelbaum, H., Noga, H., Dai, Z., Wang, J., Liang, C., Hamer, J., Ferng, C.-S., Elkind, C., Atias, A., Lee, P., Listík, V., Carlen, M., van de Kerkhof, J., Pikus, M., Zaher, K., Müller, P., Zykova, S., Stefanec, R., Gatsko, V., Hirnschall, C., Sethi, A., Xu, X.F., Ahuja, C., Tsai, B., Stefanoiu, A., Feng, B., Dhandhania, K., Katyal, M., Gupta, A., Parulekar, A., Pitta, D., Zhao, J., Bhatia, V., Bhavnani, Y., Alhadlaq, O., Li, X., Danenberg, P., Tu, D., Pine, A., Filippova, V., Ghosh, A., Limonchik, B., Urala, B., Lanka, C.K., Clive, D., Sun, Y., Li, E., Wu, H., Hongtongsak, K., Li, I., Thakkar, K., Omarov, K., Majmundar, K., Alverson, M., Kucharski, M., Patel, M., Jain, M., Zabelin, M., Pelagatti, P., Kohli, R., Kumar, S., Kim, J., Sankar, S., Shah, V., Ramachandruni, L., Zeng, X., Bariach, B., Weidinger, L., Vu, T., Andreev, A., He, A., Hui, K., Kashem, S., Subramanya, A., Hsiao, S., Hassabis, D., Kavukcuoglu, K., Sadovsky, A., Le, Q., Strohman, T., Wu, Y., Petrov, S., Dean, J. und Vinyals, O., 2024. *Gemini: A Family of Highly Capable Multimodal Models*. Verfügbar unter: <http://arxiv.org/abs/2312.11805> [Zugegriffen 29 Juni 2024].

AWS, 2024a. *Was ist Prompt Engineering? – AI Prompt Engineering erklärt – AWS*. [online] Amazon Web Services, Inc. Verfügbar unter: <https://aws.amazon.com/de/what-is/prompt-engineering/> [Zugegriffen 18 Februar 2024].

AWS, 2024b. *Was ist RAG? – Retrieval Augmented Generation erklärt – AWS*. [online] Amazon Web Services, Inc. Verfügbar unter: <https://aws.amazon.com/de/what-is/retrieval-augmented-generation/> [Zugegriffen 29 Juni 2024].

Beck, D., 2023. *Warum KI nicht frei von Vorurteilen ist*. [online] tagesschau.de. Verfügbar unter: <https://www.tagesschau.de/wissen/ki-vorurteile-101.html> [Zugegriffen 26 Oktober 2023].

Belkada, Y., Dettmers, T., Pagnoni, A., Gugger, S. und Mangrulkar, S., 2023. *Making LLMs even more accessible with bitsandbytes, 4-bit quantization and QLoRA*. [online] Verfügbar unter: <https://huggingface.co/blog/4bit-transformers-bitsandbytes> [Zugegriffen 22 Juni 2024].

botpress, 2024. *Die Zukunft der Spracherzeugung: Die 12 besten großen Sprachmodelle | Botpress Blog*. [online] Verfügbar unter: <https://botpress.com/de/blog/the-best-large-language-models-available-today> [Zugegriffen 29 Juni 2024].

Bourke, D., 2022. *PyTorch for Deep Learning & Machine Learning – Full Course*. [online] Verfügbar unter: <https://www.youtube.com/watch?v=V_xro1bcAuA> [Zugegriffen 30 Oktober 2023].

Browne, T., 2024. *Create T3 App*. [online] Create T3 App. Verfügbar unter: <https://create.t3.gg/> [Zugegriffen 21 Juni 2024].

Bundeszentrale für politische Bildung, 2024. *Europas Medien auf einen Blick*. [online] eurotopics.net. Verfügbar unter: <https://www.eurotopics.net/de/142186/medien> [Zugegriffen 12 Mai 2024].

Charles, S., 2021. *Overfitting and Underfitting*. [online] Data Science Portfolio. Verfügbar unter: <https://sourestdeeds.github.io/blog/overfitting-and-underfitting/> [Zugegriffen 2 Juni 2024].

Collins, I., Orbán, B., Domino, N., Agusti, L. und Huu Vu, T., 2024. *NextAuth.js*. [online] Verfügbar unter: <https://next-auth.js.org> [Zugegriffen 21 Juni 2024].

coolLabs Technologies Bt., 2024. *Coolify*. [online] Coolify. Verfügbar unter: <https://coolify.io> [Zugegriffen 21 Juni 2024].

CrashCourse, 2019. *Algorithmic Bias and Fairness: Crash Course AI #18*. [online] Verfügbar unter: <https://www.youtube.com/watch?v=gV0_raKR2UQ> [Zugegriffen 29 November 2023].

Dettmers, T., Pagnoni, A., Holtzman, A. und Zettlemoyer, L., 2023. *QLoRA: Efficient Finetuning of Quantized LLMs*. Verfügbar unter: <http://arxiv.org/abs/2305.14314> [Zugegriffen 24 Mai 2024].

Dontsov, A., 2021. *Neuronale Netze*. [online] nativDigital. Verfügbar unter: <https://nativdigital.com/neuronale-netze/> [Zugegriffen 8 Februar 2024].

Fawn Fitter und Steven T. Hunt, PhD, 2023. *How AI Can End Bias | SAP Insights*. [online] SAP. Verfügbar unter: <https://www.sap.com/insights/viewpoints/how-ai-can-end-bias.html> [Zugegriffen 30 Oktober 2023].

freeCodeCamp, 2023. *Create a Large Language Model from Scratch with Python – Tutorial*. [online] Verfügbar unter: <https://www.youtube.com/watch?v=UU1WVnMk4E8> [Zugegriffen 19 Januar 2024].

Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A. und Bengio, Y., 2015. *An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks*. Verfügbar unter: <http://arxiv.org/abs/1312.6211> [Zugegriffen 24 Mai 2024].

Google Career Certificates, 2024. *Discover Prompt Engineering | Google AI Essentials*. [online] Verfügbar unter: <https://www.youtube.com/watch?v=jNNatjruXx8> [Zugegriffen 27 Juli 2024].

Google for Developers, 2024. *Prompt Engineering für die generative KI | Machine Learning*. [online] Google for Developers. Verfügbar unter: <https://developers.google.com/machine-learning/resources/prompt-eng?hl=de> [Zugegriffen 27 Juli 2024].

Google Ireland Limited, 2024. *Gemini – chatten und inspirieren lassen*. [online] Gemini. Verfügbar unter: <https://gemini.google.com> [Zugegriffen 5 Juli 2024].

Google News, 2023. *Google News*. [online] Google News. Verfügbar unter: <https://news.google.com> [Zugegriffen 8 Oktober 2023].

Gow, G., 2024. *How To Use AI To Eliminate Bias*. [online] Forbes. Verfügbar unter: <https://www.forbes.com/sites/glenngow/2022/07/17/how-to-use-ai-to-eliminate-bias/> [Zugegriffen 15 Januar 2024].

Ground News, 2023. *Ground News*. [online] Ground News. Verfügbar unter: <https://ground.news/> [Zugegriffen 8 Oktober 2023].

Hetzner Online GmbH, 2024. *Dedicated Server, Cloud, Storage & Hosting*. [online] Verfügbar unter: <https://www.hetzner.com/> [Zugegriffen 21 Juni 2024].

Hölig, S., 2023. *Germany | Reuters Institute for the Study of Journalism*. [online] Verfügbar unter: <https://reutersinstitute.politics.ox.ac.uk/digital-news-report/2023/germany> [Zugegriffen 14 Januar 2024].

Hugging Face, 2023. *Hugging Face – The AI community building the future.* [online] Verfügbar unter: <https://huggingface.co/> [Zugegriffen 30 Oktober 2023].

Hugging Face, 2024. *Hugging Face – Leaderboard*. [online] Verfügbar unter: <https://huggingface.co/blog> [Zugegriffen 29 Juni 2024].

IBM, 2021. *What Is Underfitting?* [online] Verfügbar unter: <https://www.ibm.com/topics/underfitting> [Zugegriffen 29 Juni 2024].

IBM Data und AI Team, 2023. Shedding light on AI bias with real world examples. *IBM Blog*. Verfügbar unter: <https://www.ibm.com/blog/shedding-light-on-ai-bias-with-real-world-examples/> [Zugegriffen 30 Oktober 2023].

IBM Deutschland GmbH, 2023. *Was ist Prompt Engineering?* | *IBM*. [online] Verfügbar unter: <https://www.ibm.com/de-de/topics/prompt-engineering> [Zugegriffen 5 Juli 2024].

Karpathy, A., 2024. *Andrej Karpathy - YouTube*. [online] Verfügbar unter: <https://www.youtube.com/> [Zugegriffen 11 Januar 2024].

Klawonn, T., 2024. *Grenzen des „Web Scrapings“*. [online] Verfügbar unter: <https://www.forschung-und-lehre.de/recht/grenzen-des-web-scrapings-2421> [Zugegriffen 21 Juli 2024].

Loistl, C., 2024. *(KI) Zugriff auf Nachrichtenartikel für Bachelorarbeit CRM:0042326*.

Mangrulkar, S. und Paul, S., 2023. *PEFT: Parameter-Efficient Fine-Tuning of Billion-Scale Models on Low-Resource Hardware*. [online] Verfügbar unter: <https://huggingface.co/blog/peft> [Zugegriffen 24 Mai 2024].

Media Bias Group, 2024. *Media Bias Group - Home*. [online] Media Bias Group. Verfügbar unter: <https://media-bias-research.org/> [Zugegriffen 1 April 2024].

Mehta, A., 2024. Introduction to Prompt Engineering. *Medium*. Verfügbar unter: <https://medium.com/@anuj.mehta2015/introduction-to-prompt-engineering-bb440ddfb7f3> [Zugegriffen 27 Juli 2024].

Mistral AI, 2024. *Mistral AI | Frontier AI in your hands*. [online] Verfügbar unter: <https://mistral.ai/> [Zugegriffen 1 Mai 2024].

MongoDB, Inc., 2023. *MongoDB: Die Plattform Für Anwendungsdaten*. [online] MongoDB. Verfügbar unter: <https://www.mongodb.com/de-de> [Zugegriffen 13 November 2023].

Myra Security GmbH, 2023. *Web Scraping: Definition, Ursachen, Folgen, Schutz | Myra*. [online] Verfügbar unter: <https://www.myrasecurity.com/de/web-scraping/> [Zugegriffen 12 Januar 2024].

Nabwani, N., 2023. *Full Fine-Tuning, PEFT, Prompt Engineering, or RAG?* [online] Deci. Verfügbar unter: <https://deci.ai/blog/fine-tuning-peft-prompt-engineering-and-rag-which-one-is-right-for-you/> [Zugegriffen 6 Juni 2024].

Nachum, G., 2024. $ Cost of LLM continued pre-training. *Medium*. Verfügbar unter: <https://medium.com/@gilinachum/cost-of-llm-continued-pre-training-0c1998cb44ec> [Zugegriffen 29 Juni 2024].

Newman, N., Fletcher, R., Eddy, K., Robertson, C.T. und Nielsen, R.K., 2023. Reuters Institute Digital News Report 2023.

News API, 2023. *News API – Search News and Blog Articles on the Web*. [online] News API. Verfügbar unter: <https://newsapi.org> [Zugegriffen 8 Oktober 2023].

Nuxt, 2024. *Nuxt: The Intuitive Vue Framework*. [online] Nuxt. Verfügbar unter: <https://nuxt.com/> [Zugegriffen 2 Juni 2024].

OECD, 2020. *Künstliche Intelligenz in der Gesellschaft*. [online] Paris, FRANCE: OECD Publishing. Verfügbar unter: <http://ebookcentral.proquest.com/lib/saeuk/detail.action?docID=6474842> [Zugegriffen 23 Oktober 2023].

OpenAI, 2023. *OpenAI Platform*. [online] Verfügbar unter: <https://platform.openai.com> [Zugegriffen 5 Oktober 2023].

Park, Y., 2021. *IBM researchers investigate ways to help reduce bias in healthcare AI*. [online] IBM Research Blog. Verfügbar unter: <https://research.ibm.com/blog/ibm-reduce-bias-in-healthcare-ai> [Zugegriffen 30 Oktober 2023].

PI.EXCHANGE, 2024. *What is model overfitting?* [online] Verfügbar unter: <https://www.pi.exchange/knowledgehub/what-is-model-overfitting> [Zugegriffen 2 Juni 2024].

Pohl, R.F., 2022. *Cognitive Illusions: Intriguing Phenomena in Thinking, Judgment, and Memory*. 3. Aufl. [online] London: Routledge. https://doi.org/10.4324/9781003154730.

polisphere GmbH, 2024. *polisphere – passion for politics*. [online] polisphere. Verfügbar unter: <https://www.polisphere.eu/de/> [Zugegriffen 3 August 2024].

Poole, E.O., Richard, 2024. *What is the cost of training large language models?* [online] CUDO Compute. Verfügbar unter: <https://www.cudocompute.com/blog/what-is-the-cost-of-training-large-language-models> [Zugegriffen 29 Juni 2024].

Prete, A.M.D., Tzanou, M., Spiekermann, S., Taddeo, M. und Veale, M., 2022. Bias in algorithms – Artificial intelligence and discrimination. *Publications Office of the European Union*, S.106. https://doi.org/10.2811/25847.

Prisma Data, Inc., 2023. *Prisma | Next-generation ORM for Node.js & TypeScript*. [online] Prisma. Verfügbar unter: <https://www.prisma.io> [Zugegriffen 9 November 2023].

Raschka, S., 2023. *Understanding Encoder And Decoder LLMs*. [online] Verfügbar unter: <https://magazine.sebastianraschka.com/p/understanding-encoder-and-decoder> [Zugegriffen 5 Juli 2024].

shadcn/ui, 2024. *shadcn/ui*. [online] Verfügbar unter: <https://ui.shadcn.com/> [Zugegriffen 21 Juni 2024].

Similarweb, 2024a. *ground.news Traffic-Analysen, Ranking-Statistiken und Tech Stack*. [online] Similarweb. Verfügbar unter: <https://www.similarweb.com/de/website/ground.news/> [Zugegriffen 12 Januar 2024].

Similarweb, 2024b. *newsbreak.com Traffic-Analysen, Ranking-Statistiken und Tech Stack*. [online] Similarweb. Verfügbar unter: <https://www.similarweb.com/de/website/newsbreak.com/> [Zugegriffen 12 Januar 2024].

Spinde, T., Hamborg, F. und Gipp, B., 2020. Media Bias in German News Articles: A Combined Approach. In: I. Koprinska, M. Kamp, A. Appice, C. Loglisci, L. Antonie, A. Zimmermann, R. Guidotti, Ö. Özgöbek, R.P. Ribeiro, R. Gavaldà, J. Gama, L. Adilova, Y. Krishnamurthy, P.M. Ferreira, D. Malerba, I. Medeiros, M. Ceci, G. Manco, E. Masciari, Z.W. Ras, P. Christen, E. Ntoutsi, E. Schubert, A. Zimek, A. Monreale, P. Biecek, S. Rinzivillo, B. Kille, A. Lommatzsch und J.A. Gulla, Hrsg. *ECML PKDD 2020 Workshops*, Communications in Computer and Information Science. Cham: Springer International Publishing. S.581–590. https://doi.org/10.1007/978-3-030-65965-3_41.

Spinde, T., Mandl, C. und Hilmer, D., 2024. *Frage zu Ihrem Paper: Media Bias in German News Articles: A Combined Approach*.

Sun, C., Qiu, X., Xu, Y. und Huang, X., 2020. *How to Fine-Tune BERT for Text Classification?* Verfügbar unter: <http://arxiv.org/abs/1905.05583> [Zugegriffen 17 Juni 2024].

Tailwind Labs Inc., 2024. *Tailwind CSS - Rapidly build modern websites without ever leaving your HTML.* [online] Verfügbar unter: <https://tailwindcss.com/> [Zugegriffen 21 Juni 2024].

TED und Luccioni, S., 2023. *AI Is Dangerous, but Not for the Reasons You Think*. [online] Verfügbar unter: <https://www.youtube.com/watch?v=eXdVDhOGqoE> [Zugegriffen 29 November 2023].

TED und Sharma, K., 2019. *How to keep human bias out of AI*. [online] Verfügbar unter: <https://www.youtube.com/watch?v=BRRNeBKwvNM> [Zugegriffen 29 November 2023].

The Factual, 2023. *The Factual - Unbiased news, Trending topics*. [online] The Factual. Verfügbar unter: <https://www.thefactual.com/news> [Zugegriffen 8 Oktober 2023].

uni-hamburg, 2020. *20200130-handreichung-web-scraping.pdf*. [online] Verfügbar unter: <https://www.wiso.uni-hamburg.de/forschung/forschungslabor/downloads/20200130-handreichung-web-scraping.pdf> [Zugegriffen 15 März 2024].

Varsha, P.S., 2023. How can we manage biases in artificial intelligence systems – A systematic literature review. *International Journal of Information Management Data Insights*, 3(1), S.9. https://doi.org/10.1016/j.jjimei.2023.100165.

VAST.AI, 2024. *Rent GPUs | Vast.ai*. [online] Verfügbar unter: <https://vast.ai/> [Zugegriffen 11 Januar 2024].

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. und Polosukhin, I., 2017. *Attention Is All You Need*. Verfügbar unter: <http://arxiv.org/abs/1706.03762> [Zugegriffen 21 Januar 2024].

Vercel, Inc., 2023. *Next.js by Vercel - The React Framework*. [online] Verfügbar unter: <https://nextjs.org/> [Zugegriffen 9 November 2023].

Weights and Biases, Inc., 2024. *Weights & Biases: The AI Developer Platform*. [online] Weights & Biases. Verfügbar unter: <https://wandb.ai/site> [Zugegriffen 1 Juni 2024].

Xu, L., Xie, H., Qin, S.-Z.J., Tao, X. und Wang, F.L., 2023. *Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment*. Verfügbar unter: <http://arxiv.org/abs/2312.12148> [Zugegriffen 23 Mai 2024].

Ying, X., 2019. An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*, 1168, S.022022. https://doi.org/10.1088/1742-6596/1168/2/022022.

Anhang

Anhang 1 Elektronischer Anhang

Anhang 1.1 ASP

Das im ASP entstandene Ergebnis als E-Learning. Grundlagen von Transformer Modellen.

Anhang 1.2 NewsAI (Python Backend)

Anhang 1.3 NextJS Web-App

Anhang 1.4 Datensätze

Datensätze, die während des Projekts erstellt wurden und für das Training verwendet wurden:

- train-test_6700.json (KI gelabelter Datensatz mit 6.700 Sampels pro Kategorie)
- train-test-manu_6700.json (Manuell gelabelter Datensatz mit 6.700 Sampels pro Kategorie)
- train-test-manu_12000.json (Manuell gelabelter Datensatz mit 12.000 Sampels pro Kategorie)

Anhang 1.5 Model Weights

Trainierte Modelle:

- distilbert-base-uncased-finetuned-news-bias-de (Modell, welches mit KI-gelabelten Daten trainiert wurde)
- distilbert-base-uncased-finetuned-news-bias-de-manu (Modell, welches mit Manuell-gelabelten Daten trainiert wurde)

Anhang 1.6 Keys und Passwörter

Keys und Passwörter, die benötigt werden, damit die NextJS Web-App und das Python-Backend funktionieren. (Bitte vertraulich behandeln.)

Anhang 1.7 Hugingface Spaces

- unbiased-news (Huggingface Space mit dem KI-Modell, welches mit KI-Gelabelten Daten trainiert wurde)
- unbiased-news-manu (Huggingface Space mit dem KI-Modell, welches mit Manuell-Gelabelten Daten trainiert wurde)

Anhang 2 Zeitplan

Zeitplan

Name	Blockiert	Blockiert von	Datum
Datenbeschaffung	Labeling, Feature: Zusammenfassung der News		@11. März 2024 → 24. März 2024
Research: Datenbank	Skript: automatisierte Speicherung		@14. März 2024
Skript: automatisierte Speicherung	Automatisierung: Speicherung der Daten	Research: Datenbank	@14. März 2024 → 15. März 2024
Automatisierung: Speicherung der Daten	Labeling	Skript: automatisierte Speicherung	@16. März 2024 → 17. März 2024
Schreiben des Major Projekts (Puffer)			@21. März 2024 → 24. März 2024
Labeling	Training & Finetuning	Datenbeschaffung, Automatisierung: Speicherung der Daten	@28. März 2024 → 14. April 2024
Skript: automatisiertes Labeling	Automatisierung: Labeling der News, Labeling: Fake News		@28. März 2024 → 4. April 2024

Zeitplan 1

Aa Name	↗ Blockiert	↗ Blockiert von	Datum
Labeling: Fake News		Skript: automatisiertes Labeling	@4. April 2024 → 6. April 2024
Automatisierung: Labeling der News		Skript: automatisiertes Labeling	@6. April 2024 → 7. April 2024
Schreiben des Major Projekts (Puffer)			@11. April 2024 → 14. April 2024
🔍 Research: KI-Model	🏋 Training & Finetuning		@18. April 2024 → 28. April 2024
Vergleiche der Modelle			@18. April 2024 → 21. April 2024
Schreiben des Major Projekts (Puffer)			@25. April 2024 → 28. April 2024
Reasearch: Finetuning	Skript: Finetuning		@2. Mai 2024 → 11. Mai 2024
🏋 Training & Finetuning	🏁 Finalisierung	🔍 Research: KI-Model, 💉 Labeling	@2. Mai 2024 → 15. Juni 2024
Research: Hosting & Hosting einrichten			@11. Mai 2024 → 16. Mai 2024
Skript: Finetuning	Automatisierung: Finetuning	Reasearch: Finetuning	@17. Mai 2024 → 1. Juni 2024
Automatisierung: Finetuning		Skript: Finetuning	@2. Juni 2024 → 8. Juni 2024
Schreiben des Major Projekts (Puffer)			@9. Juni 2024 → 15. Juni 2024
Auth	Veröffentlichung		@16. Juni 2024 → 20. Juni 2024
👷 Entwicklung der Application	🏁 Finalisierung, 🇨🇭 Feature: Zusammenfassung der News		@16. Juni 2024 → 14. Juli 2024
Darstellung der News und Metadaten	Tests, Optimierung primär auf mobile Geräte		@20. Juni 2024 → 22. Juni 2024
Veröffentlichung	Tests	Auth	@21. Juni 2024 → 23. Juni 2024
Tests		Darstellung der News und	@27. Juni 2024 → 4. Juli 2024

Zeitplan 2

Aa Name	↗ Blockiert	↗ Blockiert von	Datum
		Metadaten, Veröffentlichung	
Optimierung primär auf mobile Geräte	Feature: Leichte und intuitive Bedienung nach Vorbild von Google News	Darstellung der News und Metadaten	@4. Juli 2024 → 5. Juli 2024
Feature: Leichte und intuitive Bedienung nach Vorbild von Google News		Optimierung primär auf mobile Geräte	@6. Juli 2024 → 7. Juli 2024
Schreiben des Major Projekts			@11. Juli 2024 → 14. Juli 2024
🇬🇧 Feature: Zusammenfassung der News		💾 Datenbeschaffung, 👷 Entwicklung der Application	@18. Juli 2024 → 26. Juli 2024
Zusammenfassung einrichten			@18. Juli 2024 → 19. Juli 2024
Schreiben des Major Projekts			@20. Juli 2024 → 26. Juli 2024
Zusammenführen der Systeme			@27. Juli 2024 → 28. Juli 2024
🏁 Finalisierung		🏋️ Training & Finetuning, 👷 Entwicklung der Application	@27. Juli 2024 → 4. August 2024
Schreiben des Major Projekts			@1. August 2024 → 4. August 2024
🚫 Sonstige Zeit (Blocker)			@8. August 2024 → 23. August 2024
Feedback			@8. August 2024 → 9. August 2024
Unterrichte (15h)			@10. August 2024 → 11. August 2024
Krankheit			@15. August 2024 → 18. August 2024
Urlaub			@22. August 2024 → 23. August 2024

Zeitplan 3

Anhang 3 Datenanfrage-E-Mail

Sehr geehrte Damen und Herren,

mein Name ist Daniel Hilmer und ich bin Student im Bachelorstudiengang Webdesign & Web-Development an der SAE in München. Im Rahmen meiner Bachelorarbeit beschäftige ich mich mit dem Training einer Künstlichen Intelligenz (KI), die deutsche Nachrichtenartikel anhand der verwendeten Sprache auf die politischen Einrichtungen analysiert und die Ergebnisse auf einer Webanwendung bereitstellt. Aktuell nutze ich für die Datensammlung Ausschnitte aus der News-API (https://newsapi.org/). Um die Analyse jedoch zu vertiefen und die KI genauer trainieren zu können, benötige ich Zugriff auf den vollständigen Inhalt der Nachrichtenartikel.

Aus diesem Grund möchte ich folgende Anfragen an Sie stellen:
- Zugang zu Ihrer API-Schnittstelle: Dies würde mir ermöglichen, die Nachrichtenartikel automatisiert und effizient abzurufen und zu analysieren.
- Erlaubnis zum Herunterladen der Artikel mithilfe eines Programms (Scraping): Falls ein API-Zugriff nicht möglich ist, wäre ich auch dankbar, wenn Sie mir die Erlaubnis erteilen würden, die Artikel mithilfe eines Programms automatisiert herunterzuladen.

Ich versichere Ihnen, dass die KI und die Webanwendung nicht veröffentlicht werden. Die Artikel werden ausschließlich im Rahmen meines Bachelorprojekts verwendet.

Ich bedanke mich im Voraus für Ihre Zeit und Mühe. Für Rückfragen stehe ich Ihnen jederzeit gerne zur Verfügung.

Sollten Sie nicht die richtige Ansprechperson sein, leiten Sie diese E-Mail gerne entsprechend weiter.

Mit freundlichen Grüßen
Daniel Hilmer

Anhang 4 Antwort des BRs zur Scraping Anfrage

(KI) Zugriff auf Nachrichtenartikel für Bachelorarbeit CRM:0042326

Techinfo, Service <Service.Techinfo@br.de>
Do, 02.05.2024 13:30
An:Daniel Hilmer <mail@danielhilmer.de>

2 Anlagen (21 KB)
image001.jpg; image002.png;

Sehr geehrter Herr Hilmer,

vielen Dank für Ihre Anfrage und Ihr Interesse am Bayerischen Rundfunk. Es hat bei unserer hausinternen Recherche etwas gedauert, dafür bitten wir um Verständnis.
Leider können wir Ihnen aktuell keine API Schnittstelle zur Verfügung stellen.

Es ist ein sehr interessantes Thema, das Sie sich für Ihre Bachelor-Arbeit ausgesucht haben. Leider ist es aber auch so, dass viele rechtliche und strategische Fragen rund um das Thema KI noch offen oder nicht abschließend geklärt. Ihre Anfrage enthält viele Aspekte, was die Gestaltung der Rahmenbedingungen für ein externes KI-Training mit unserem Material betrifft. Hier bedarf es der Einbeziehung verschiedener Entscheidungsträger (Geschäftsleitung, Juristen, Technik). In Folge dessen müssten auch die technischen Wege z.B. Mehrbelastungen für die Schnittstellen erst getestet werden. Sie sind deshalb mit Ihrer Anfrage bei uns einfach (noch) zu früh dran.

Die Einschätzung unseres zuständigen Juristen zu einer vergleichbaren Anfrage lautete, dass öffentlich zugängliche Inhalte via Text und Data Mining genutzt werden können und dies nach wie vor gilt. Archivmaterial kann auf diese Weise nicht genutzt werden. Eine API- Schnittstelle kann der BR dafür – wie oben erläutert – jedenfalls derzeit nicht zur Verfügung stellen. Wenn es nur um die aktuellen Inhalte geht, gibt es auch die Möglichkeit, die RSS-Feeds der BR-Angebote zu nutzen.

Wir hoffen auf Ihr Verständnis und wünschen Ihnen viel Erfolg bei Ihrer Abschlussarbeit.

Mit freundlichen Grüßen
Christine Loistl

Technische Information

Bayerischer Rundfunk
Produktions- und Technikdirektion
Rundfunkplatz 1 | 80335 München
Telefon: +49 800 5900 - 789 (kostenfrei)
techinfo@br.de
BR.de/technik

[br.de/klaresprache]

---------- Ursprüngliche Nachricht ----------
Von: Daniel Hilmer <mail@danielhilmer.de>
Empfangen: Donnerstag, 21. März 2024 16:30:46
An: BRonline InfoService <bronline.infoservice@br.de>;
BRonline.InfoService@br.de <bronline.infoservice@br.de>
Betreff: Zugriff auf Nachrichtenartikel für Bachelorarbeit

Sehr geehrte Damen und Herren,

mein Name ist Daniel Hilmer und ich bin Student im Bachelorstudiengang Webdesign & Web-Development an der SAE in München. Im Rahmen meiner Bachelorarbeit beschäftige ich mich mit dem Training einer Künstlichen Intelligenz (KI), die deutsche Nachrichtenartikel anhand der verwendeten Sprache auf die politischen Einrichtungen analysiert und die Ergebnisse auf einer Webanwendung bereitstellt. Aktuell nutze ich für die Datensammlung Ausschnitte aus der News-API (https://newsapi.org/). Um die Analyse jedoch zu vertiefen und die KI genauer trainieren zu können, benötige ich Zugriff auf den vollständigen Inhalt der Nachrichtenartikel.

Aus diesem Grund möchte ich folgende Anfragen an Sie stellen:
- Zugang zu Ihrer API-Schnittstelle: Dies würde mir ermöglichen, die Nachrichtenartikel automatisiert und effizient abzurufen und zu analysieren.
- Erlaubnis zum Herunterladen der Artikel mithilfe eines Programms (Scraping): Falls ein API-Zugriff nicht möglich ist, wäre ich auch dankbar, wenn Sie mir die Erlaubnis erteilen würden, die Artikel mithilfe eines Programms automatisiert herunterzuladen.

Ich versichere Ihnen, dass die KI und die Webanwendung nicht veröffentlicht werden. Die Artikel werden ausschließlich im Rahmen meines Bachelorprojekts verwendet.
Ich bedanke mich im Voraus für Ihre Zeit und Mühe.
Für Rückfragen stehe ich Ihnen jederzeit gerne zur Verfügung.

Sollten Sie nicht die richtige Ansprechperson sein, leiten Sie diese E-Mail gerne entsprechend weiter.

Mit freundlichen Grüßen
Daniel Hilmer

Anhang 5 Media Bias Group - Annomatic Anfrage

Re: Frage zu Ihrem Paper: Media Bias in German News Articles: A Combined Approach

Christoph Mandl <C.Mandl@media-bias-research.org>
Do, 14.03.2024 12:40
An:Daniel Hilmer <d.hilmer@sae.edu>
Cc:Timo Spinde <t.spinde@media-bias-research.org>

Hallo Herr Hilmer,

danke für den Githubnamen ich werde sie hinzugfuegen.

Bzgl. des LLMs:

Leider sind noch nicht viele 7B LLMs fuer die deutsche Sprache verfügbar. Die meisten open-source LLMs die deutsch können sind leider gross (e.g. Mixtral 8x7B).

Das einzige (mir bekannte) 7B LLM waere https://huggingface.co/occiglot/occiglot-7b-de-en-instruct welches leider nicht auf Colab laeuft (wegen CPU RAM limitations beim laden).
Somit denke ich das es, mit den gegebenen Mitteln, schwierig ist nur auf open-source LLMs zurück zu greifen.

Ich kann verstehen, dass sie gerne auf OpenAI verzichten wuerden. Deshalb bin ich mir nicht ganz sicher was hier die beste vorgehensweise ist damit sie ihre Daten am annotiert kostengünstigsten bekommen.

Sieh koennen mir auch gerne auf Discord schreiben (username ist mandlchr).

Dort das schreiben vielleicht angenehmer und schneller.

Viele Grüße,

Christoph

On 3/13/24 10:21, Daniel Hilmer wrote:

> Hallo Herr Mandl,
>
> vielen Dank für die Rückmeldung.
> Meinen GitHub Account finden Sie unter: dackJaniel oder per E-Mail d.hilmer@sae.edu bzw. daniel.hilmer@outlook.de.
>
> Für das LLM könnte ich mir folgende Lösungen vorstellen:
>
> 1. Ein großes Cluster habe ich nicht. Die SAE hat allerdings einige leistungsfähige Computer (selbstverständlich inkl. GPUs), die ich für das Projekt verwenden kann. Ggf. z.B. mit LM Studio (https://lmstudio.ai/) wenn möglich.
> 2. Einen OpenAI Key habe ich. Am liebsten würde ich auf diesen allerdings verzichten, wenn möglich, um die Kosten für das Projekt so klein wie möglich zu halten.
> 3. Auch Google Colab habe ich bereits verwendet. Dies wäre deshalb eine alternative.
>
> Welches der Lösungen halten Sie am geeignetsten?
>
> Vielen Dank für Ihre Unterstützung.
> Viele Grüße
> Daniel Hilmer
>
> Am Di., 12. März 2024 um 18:54 Uhr schrieb Christoph Mandl <C.Mandl@media-bias-research.org>:
>
> > Hallo Herr Hilmer,
> >
> > Ich entschuldige mich für die späte Antwort.
> >
> > Das Framework ist ein Pythonpackage. Dieses ist noch nicht in PyPI hinterlegt wordurch das direkte klonen notwendig ist. Hierfuer muesste ich Sie zu dem Github Projekt hinzufügen.

Haben sie einen Github username dafür?

Desweiteren brauchen Sie die Möglichkeit LLMs laufen bzw. abfragen zu können.

D.h. entweder:
1) einen Cluster auf dem das LLM laufen kann
2) einen OpenAI API-Key für die interaction mit OpenAI Models.
3) für kleine Models(<= 7B) können in Google Colab/Kaggle Notebooks laufen. Jedoch nur in einer quantifiziert Variante.

Haben sie eine dieser Möglichkeiten?

Sollten Sie noch weitere Fragen haben, können sie sich gerne bei mir melden.

Herzliche Grüße
Christoph Mandl

On 3/11/24 11:00, Timo Spinde wrote:

> Hallo Herr Hilmer,
>
> das passt, wir können bereits Zugang geben! Hier in CC ist der Haupt-Entwickler des Annomatic-Frameworks, Christoph Mandl, der weitere Informationen bereitstellen kann. Ich sage ihm Bescheid!
>
> Herzliche Grüße,
>
> Timo Spinde
>
> Am 11.03.2024 um 10:31 schrieb Daniel Hilmer:
>
> > Hallo Herr Spinde,
> >
> > gerne wollte ich mich informieren, ob Sie mir bereits Zugänge zu Ihrem System geben können.
> > Ich befinde mich nun im Major Projekt (dem Bachelor Projekt) und möchte laut Zeitplan spätestens am **28.03.2024** mit dem Labeling meiner Daten beginnen. Im Optimalfall könnte ich das System zuvor testen.
> > Ist das für Sie realistisch? Benötigen Sie weiteren Informationen von mir?
> >
> > Vielen Dank für Ihre Unterstützung
> > Viele Grüße
> > Daniel Hilmer
> >
> > Am Sa., 13. Jan. 2024 um 12:55 Uhr schrieb Timo Spinde <t.spinde@media-bias-research.org>:
> >
> > > Hallo Herr Hilmer,
> > >
> > > alles klar. Tatsächlich haben wir nicht direkt neue Daten, aber wir entwickeln seit geraumer Zeit in Framework, mit dem Daten automatisch gelabeled werden können, mit Hilfe zahlreicher LLMs. Die Qualität ist vergleichbar mit menschlichen Annotationen. Ich denke, so in ca. 4- 6 Wochen können wir Ihnen Zugang zu einer ersten öffentlichen Version geben.
> > >
> > > Ich meinen Kollegen Herrn Mandl hier mit in CC, der den Zugang dann teilen wird. Das sollte einen großen Benefit für ihr Projekt bringen, und gibt uns zeitgleich die Möglichkeit, unser Framework mit ersten Anwendern zu testen.
> > >
> > > Herzliche Grüße,

Timo Spinde

Am 13.01.2024 um 10:00 schrieb Daniel Hilmer:

> Hallo Herr Spinde,
>
> vielen Dank für Ihre schnelle Rückmeldung sowie Ihre Offenheit und die Informationen.
> Für meine Arbeit plane ich, ein Pretrained Machiene Learning Model zu fine tunen. Ich werde vorerst 5 Output-Labels haben (links, leicht link, mitte, leicht rechts, rechts) in die ich deutsche Nachrichten automatisiert einordnen möchte.
> Mein Plan ist bis dato, mit einem Datensatz von ~5.000 - 50.000 Daten zu arbeiten (hier habe ich leider noch keine genauen Referenzwerte).
> Evtl. wissen Sie nun schon, ob Ihre Daten für mich relevant sein könnten oder nicht. Falls ja, freue ich mich sehr.
> Bezüglich meines Zeitplans: Aktuell befinde ich mich in der Vorbereitung auf mein Bachelorprojekt. Diese endet am 16.02.24. Daraufhin erfolgt die Umsetzung bis in den August, 24.
> Gerne würde ich Ihr Angebot der gelabelten Daten annehmen. Auch wenn dies bis über das Ende der Vorbereitungszeit hinaus benötigt.
> Sollten Sie noch weitere Fragen oder Anregungen haben, schreiben Sie mir gerne.
>
> Mit freundlichen Grüßen
> Daniel Hilmer
>
> ---
>
> **Von:** Timo Spinde <Timo.Spinde@uni-konstanz.de>
> **Gesendet:** Freitag, 12. Januar 2024 17:57
> **An:** Daniel Hilmer <d.hilmer@sae.edu>; felix.hamborg@uni-konstanz.de <felix.hamborg@uni-konstanz.de>; bela.gipp@uni-konstanz.de <bela.gipp@uni-konstanz.de>
> **Cc:** spinde@uni-wuppertal.de <spinde@uni-wuppertal.de>; gipp@uni-wuppertal.de <gipp@uni-wuppertal.de>
> **Betreff:** Re: Frage zu Ihrem Paper: Media Bias in German News Articles: A Combined Approach
>
> Sehr geehrter Herr Hilmer,
>
> vielen Dank für die Nachricht! Es gibt tatsächlich nicht viele Arbeiten in dem Bereich in Deutsch, ich arbeite allerdings aktuell an einer mehrsprachigen Variante.
>
> Die Daten von dem angesprochenen Paper (das ja doch schon einige Jahre alt ist) sind leider auf einer defekten externen Festplatte, die schon seit langem repariert gehört; sobald ich das erledigt habe, kann ich sie Ihnen schicken. Allerdings sind dort nur wenige Artikel gelabeled worden, sodass es Ihnen vermutlich keinen besonders großen Vorteil bringt. Sollten Sie selbst Daten annotieren wollen; Wir haben das bis zuletzt mit unserer

Plattform TASSY gemacht (in diesem Paper dokumentiert: TASSY), für die Implementierung gibt es hier ausführliche Details: https://github.com/Media-Bias-Group/Teaching-Platform.

Wie ist denn ihr Zeitplan? Da wir aktuell Daten labeln, kann ich ihnen in einiger Zeit (4 - 8 Wochen) ggfs. passende Daten zukommen lassen. Um hier nicht zuviel Nachrichten hin und herzuschreiben antworten sie ruhig nur mir, als Erstautor kenne ich die Arbeit von damals am besten.

Mit freundlichen Grüßen,

Timo Spinde

Am 12.01.2024 um 16:07 schrieb Daniel Hilmer:

> Sehr geehrte Damen und Herren,
>
> mit Freuden habe ich Ihr Paper „Media Bias in German News Articles: A Combined Approach" gelesen. Eines der wenigen, das sich mit deutschen Nachrichten auseinandergesetzt hat.
> Derzeit befinde ich mich in der Vorbereitung meines Bachelorprojekts mit dem Thema: „Entwicklung einer deutschsprachigen Unbiased News SaaS-Applikation zur automatischen News-Analyse und politischen Einordnung mit AI-Unterstützung", an der SAE in München. Aus diesem Grund wende ich mich an Sie, mit der Frage, ob Sie mir weitere Informationen bzw. die gelabelten Datensätze im Rahmen meiner Bachelorarbeit zur Verfügung stellen könnten. Dies würde meine Labeling-Arbeit der Newsartikel voraussichtlich erheblich vereinfachen.
> Sollten Sie noch weitere Informationen zu mir oder meiner Arbeit benötigen, um die Entscheidung zu treffen, wenden Sie sich gerne an mich (daniel.hilmer@outlook.de).
>
> Vielen Dank für Ihre Unterstützung. Ich freue mich auf eine kurze Rückmeldung.
>
> Mit freundlichen Grüßen
> Daniel Hilmer

--

Media Bias Group Logo

Anhang 6 Raw Daten des Modellvergleichs

```
 1. Model: albert-base-v2, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16666666666666666, 'Execution Time': 66.36662077903748, 'Precision score': 0.7777777777777777,
'Recall score': 0.3333333333333333}
 2. Model: albert-large-v2, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16666666666666666, 'Execution Time': 184.79143381118774, 'Precision score': 0.7777777777777777,
'Recall score': 0.3333333333333333}
 3. Model: bert-base-uncased, Metrics: {'Accuracy': 0.3343333333333333, 'F1 Score':
0.1705813638346204, 'Execution Time': 60.551806926727295, 'Precision score': 0.6263369729411884,
'Recall score': 0.3343333333333333}
 4. Model: bert-large-uncased, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16754628466113763, 'Execution Time': 179.1169421672821, 'Precision score': 0.44522770504643616,
'Recall score': 0.3333333333333333}
 5. Model: dbmdz/bert-base-german-cased, Metrics: {'Accuracy': 0.326, 'F1 Score':
0.25319175081227635, 'Execution Time': 57.885921478271484, 'Precision score': 0.5411804397246754,
'Recall score': 0.326}
 6. Model: deepset/gelectra-base, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16666666666666666, 'Execution Time': 56.61951923370361, 'Precision score': 0.7777777777777777,
'Recall score': 0.3333333333333333}
 7. Model: deepset/gelectra-large, Metrics: {'Accuracy': 0.33, 'F1 Score': 0.20928954768340394,
'Execution Time': 155.5948007106781, 'Precision score': 0.5371958220156254, 'Recall score': 0.33}
 8. Model: distilbert-base-multilingual-cased, Metrics: {'Accuracy': 0.3333333333333333, 'F1
Score': 0.16666666666666666, 'Execution Time': 35.639017820358276, 'Precision score':
0.7777777777777777, 'Recall score': 0.3333333333333333}
 9. Model: distilbert-base-uncased, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16666666666666666, 'Execution Time': 36.38976788520813, 'Precision score': 0.7777777777777777,
'Recall score': 0.3333333333333333}
10. Model: google/electra-base-discriminator, Metrics: {'Accuracy': 0.3466666666666667, 'F1
Score': 0.27730638902552335, 'Execution Time': 61.06939435005188, 'Precision score':
0.5645973473188158, 'Recall score': 0.3466666666666667}
11. Model: google/electra-small-discriminator, Metrics: {'Accuracy': 0.3333333333333333, 'F1
Score': 0.16666666666666666, 'Execution Time': 42.78672766685486, 'Precision score':
0.7777777777777777, 'Recall score': 0.3333333333333333}
12. Model: google-bert/bert-base-german-cased, Metrics: {'Accuracy': 0.3433333333333333, 'F1
Score': 0.25703209149322964, 'Execution Time': 56.724793910980225, 'Precision score':
0.5918828676228659, 'Recall score': 0.3433333333333333}
13. Model: intfloat/multilingual-e5-large, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16737173539893357, 'Execution Time': 162.09458374977112, 'Precision score': 0.5111853088480801,
'Recall score': 0.3333333333333333}
14. Model: intfloat/multilingual-e5-large-instruct, Metrics: {'Accuracy': 0.33666666666666667,
'F1 Score': 0.19756323361467196, 'Execution Time': 156.8266360759735, 'Precision score':
0.49828794065011844, 'Recall score': 0.33666666666666667}
15. Model: microsoft/deberta-base, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16666666666666666, 'Execution Time': 113.31463932991028, 'Precision score': 0.7777777777777777,
'Recall score': 0.3333333333333333}
16. Model: microsoft/deberta-large, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16666666666666666, 'Execution Time': 304.0361235141754, 'Precision score': 0.7777777777777777,
'Recall score': 0.3333333333333333}
17. Model: roberta-base, Metrics: {'Accuracy': 0.343, 'F1 Score': 0.2633101851851852, 'Execution
Time': 65.05475234985352, 'Precision score': 0.5693181818181818, 'Recall score':
0.34299999999999997}
18. Model: roberta-large, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score':
0.16666666666666666, 'Execution Time': 182.886292219162, 'Precision score': 0.7777777777777777,
'Recall score': 0.3333333333333333}
19. Model: t5-base, Metrics: {'Accuracy': 0.32866666666666666, 'F1 Score': 0.25307373375865727,
'Execution Time': 161.87602353096008, 'Precision score': 0.5401493398757733, 'Recall score':
0.32866666666666666}
20. Model: t5-large, Metrics: {'Accuracy': 0.3343333333333333, 'F1 Score': 0.16947916998017196,
'Execution Time': 451.8825409412384, 'Precision score': 0.6112967914438503, 'Recall score':
0.3343333333333333}
21. Model: t5-small, Metrics: {'Accuracy': 0.3333333333333333, 'F1 Score': 0.16666666666666666,
'Execution Time': 71.99578809738159, 'Precision score': 0.7777777777777777, 'Recall score':
0.3333333333333333}
22. Model: xlnet-base-cased, Metrics: {'Accuracy': 0.3373333333333333, 'F1 Score':
0.26829615346439467, 'Execution Time': 124.45333051681519, 'Precision score': 0.5805171632641176,
'Recall score': 0.3373333333333333}
```

```
23. Model: xlnet-large-cased, Metrics: {'Accuracy': 0.34933333333333333, 'F1 Score':
0.26457945948978795, 'Execution Time': 332.0750961303711, 'Precision score': 0.5519309481573632,
'Recall score': 0.34933333333333333}
```

Anhang 7 Hyperparameter Tuning (sweeps)

```
sweep_config = {
    "method": "random",
    'parameters': {
        'learning_rate': {
            'min': 0.0001,
            'max': 0.1
        },
        'batch_size': {
            'values': [8, 16, 32, 64]
        },
        'epochs': {
            "value": 10,
        },
        'dropout': {
            'min': 0.1,
            'max': 0.5
        },
        'num_layers': {
            'values': [2, 3, 4, 5, 6]
        },
        "weight_decay": {
            "min": 0.0,
            "max": 0.5
        },
        "warmup_steps": {
            "min": 0,
            "max": 1000
        },
        "peft_alpha": {
            "min": 1,
            "max": 32
        },
        "peft_dropout": {
            "min": 0.1,
            "max": 0.5
        },
        "peft_r": {
            "min": 16,
            "max": 128
        },
    },
    "metric": {
        'name': 'eval/loss',
        'goal': 'minimize'
    },
    'early_terminate': {
        'type': 'hyperband',
        'min_iter': 5
    },
}
```

Anhang 8 Optimierte Hyperparameter

```
lora_alpha=50
lora_dropout=0.6
r=50
bias="none"
target_modules=["q_lin", "v_lin"]
task_type=TaskType.SEQ_CLS
num_train_epochs=1000
learning_rate=0.0003
per_device_train_batch_size=64
per_device_eval_batch_size=64
warmup_steps=300
weight_decay=0.25
fp16=True
load_best_model_at_end=True
metric_for_best_model="eval_loss"
greater_is_better=False
```

Anhang 9 Anleitung zur Verwendung der NewsAI (Python Backend)

README.md 2024-08-16

NewsAI

1 / 10

README.md 2024-08-16

Prototype

Voraussetzungen

- Docker installieren & starten

Installieren & Starten

SSH

Wenn man sich mit SSH in den Container verbinden möchte, muss ein SSH Key mit dem Name `id_rsa.pub` in das Hauptverzeichnis des Projekts gelegt werden und den aktuellen Key ersetzen. Der private Key muss auf dem System vorhanden sein.

Build

```
docker composer build
```

Über den Build Befehl wird das `Dockerfile` ausgeführt und das Image erstellt. Bei einem Docker Build wird nur das Image für die Produktivumgebung erstellt. Für die Dev-Umgebung kann im `docker-compose.yml` das volume `./:/usr/src/app` hinzugefügt werden.

Starten

```
docker compose up
```

Über den Start Befehl wird die `docker-compose.yml` Datei ausgeführt.

Verbindung zum Container

```
# Verbindung zum Container
docker exec -it newsai bash
```

Alternativ kann eine Verbindung über

- SSH hergestellt werden (wenn ein SHH Key vorhanden ist)
- oder über die VSCode Remote Extension hergestellt werden.

Poetry

Poetry wird für das Dependency Management verwendet. Folgende Befehle können in der Shell ausgeführt werden:

```
```

2 / 10

```
# Shell starten (wenn nicht bereits in der Shell)
# wird vor dem ausführen der Befehle benötigt
poetry shell
```

```
# Dependencies installieren
# wird benötigt wenn Dependencies nicht gefunden werden
poetry install
```

.env

Wichtig: Ohne das Volume in der `docker-compose.yml` Datei, muss die `.env` Datei manuell in der Docker Umgebung erstellt werden.

Benenne die `.env.example` Datei in `.env` um und fülle die Variablen aus. `DB_NAME_TEST` wird nur für Tests benötigt. Lösche sie, wenn du sie nicht verwendest. Alle anderen Variablen sind notwendig.

Testen

Für die Tests wird `pytest` verwendet.

Wichtig:

- Tests sind nur verfügbar, wenn das Volume in der `docker-compose.yml` Datei aktiviert ist
- setzen der `DB_NAME_TEST=newsai_test` Variable in der `.env` Datei

Die Unit Tests können in der `tests` Ordner gefunden werden. Die Tests können über folgenden Befehl ausgeführt werden:

```
# Ausführen der Unit Tests
pytest tests
```

Die Integration Tests können in dem `tests` Ordner gefunden werden. Die Tests können über folgenden Befehl ausgeführt werden:

```
# Ausführen der Integration Tests
pytest tests --integration
```

Die Integration Tests benötigen lange, da sie Live Daten verwenden.

Cronjob

Der Cronjob wird in der `Dockerfile` Datei ausgeführt. Der Cronjob wird alle 12 Stunden ausgeführt und die NewsAPI wird abgefragt. Er kann mittels folgendem Befehl überprüft werden:

```
crontab -l
```

Um den Cronjob nicht auszuführen, können die Zeilen in der `Dockerfile` auskommentiert werden (Kommentar im `Dockerfile` dient als Hinweis).

Alternativ kann der Cronjob auch manuell gelöscht werden:

```
crontab -r
```

Ausführen

Alle notwendigen Dateien sind im `src` Ordner. Die Produktivumgebung kann über die `run.py` Datei gestartet werden.

```
python3 src/run.py
```

In der `run.py` werden aktuell über die Funktion `get_newsapi_prod()` 10 Nachrichtenartikel abgerufen. Diese kann in der Funktion (Datei: `src/scraping/news.py`) angepasst werden. Dafür muss die `page_size=` Variable in der Funktion geändert werden. Folgende Einstellungen wurden getroffen:

```
news_res = newsapi.get_everything(
            language="de",
            sort_by="publishedAt",
            page_size=10,
            page=1,

domains="zeit.de,spiegel.de,faz.net,tagesschau.de,tagesspiegel.de,handelsblatt.com
,welt.de,sueddeutsche.de,bild.de,morgenpost.de,freitag.de,fr.de,hallespektrum.de,j
ungefreiheit.de,jungewelt.de,n-
tv.de,stern.de,mdr.de,ndr.de,hessenschau.de,report24.news,blauenarzisse.de,sezessi
on.de,t-online.de,fr.de,tz.de,rtl.de,zdf.de,deutschlandfunk.de,br.de",
            exclude_domains="news.google.com",
        )
```

Diese Einstellungen können in der `src/scraping/news.py` Datei geändert werden. Die Anleitung ist hier: https://newsapi.org/docs/endpoints/everything

Die Testumgebung kann über die `run_dev.py` Datei gestartet werden. (Dafür muss das Volume in der `docker-compose.yml` Datei aktiviert sein.)

Je nachdem was ausgeführt werden soll, können Funktionen und Methoden in die `run_dev.py` Datei importiert werden. Dies wurde für die Entwicklung verwendet. Aus diesem Grund kann es sein, dass einige Funktionen in der aktuellen Konfiguration nicht funktionieren. Die `run_dev.py` Datei ist nur für die Entwicklung gedacht. Alle garantiert funktionierenden Funktionen sind in der `run.py` Datei.

```
# Shell starten (wenn nicht bereits in der Shell)
poetry shell
# Installieren der Dependencies (alle, auch dev)
poetry install
```

```
python3 src/run_dev.py
```

Wenn PyTorch benötigt wird, muss es manuell installiert werden:

```
# Shell starten (wenn nicht bereits in der Shell)
poetry shell
# Torch installieren
pip3 install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu121
```

FYI: Die CLI funktioniert in den meisten Fällen nicht! Sie sollte nicht verwendet werden.

Hinweise

- Bei dieser App handelt es sich um einen Prototypen. Zukünftig müsste der Code Refactored werden und einige Abläufe optimiert werden.
- Die Keys und Datenbanken für die den Prototypen sind in der Test und Produktivumgebung gleich. Änderungen in der Datenbank werden auch auf die Live Web-App übertragen.
- Bitte den Container bzw. den Cronjob nicht zu oft/ lange laufen lassen, da die Laufzeit der Inference Kosten verursacht.

Notizen zur Entwicklung

CUDA

Install

https://docs.nvidia.com/cuda/cuda-installation-guide-microsoft-windows/index.html

Wenn Error in Sample Files

https://stackoverflow.com/questions/12136808/cuda-4-2-props-and-target-file-is-missing-visual-studio-2010

WSL2 - Windows Subsystem for Linux

Install

https://learn.microsoft.com/de-de/windows/wsl/install

```
wsl --install
```

Check Version

```
wsl --list --verbose
```

Docker

Install

Docker Desktop for Windows: https://hub.docker.com/editions/community/docker-ce-desktop-windows/

Build

```
docker compose build
```

Run

```
docker compose up
```

Info zu MONGO IN DOCKER

Um auf MONGO zuzugreifen kann man außerhalb von Docker den conn-String `mongodb://admin:admin@localhost:27017/` verwenden. In Docker muss `mongodb://admin:admin@mongo:27017/` verwendet werden. Hier braucht man den Namen des Containers/ Service.

News API

URL

https://newsapi.org/

Install

```
python3 -m pip install newsapi-python
```

Google Cloud Engine

Um eine GPU für das Training und die Annotation mit mehr Leistung zu haben. kann die Google Cloud Engine verwendet werden. Hierbei wird ein Docker Image auf Google Artifact Registry gepusht und auf der GCE

ausgeführt.

Google Cloud CLI: https://cloud.google.com/sdk/docs/install?hl=de **YT-Video:** https://www.youtube.com/watch?v=cLOld8de5ZI

Lokales Image auf Artifact Registry pushen

Anleitung: https://cloud.google.com/artifact-registry/docs/docker/pushing-and-pulling?authuser=1&hl=de

Taggen

```
europe-west3-docker.pkg.dev/newsai-419410/newsai/newsai
```

```
docker tag newsai europe-west3-docker.pkg.dev/newsai-419410/newsai/newsai
```

Pushen

```
docker push europe-west3-docker.pkg.dev/newsai-419410/newsai/newsai
```

In der Artifact Registry ist das Image nun zu finden. Jetzt kann ein Compute Engine erstellt werden und das Image darauf ausgeführt werden. Am einfachsten ist in das Repository zu gehen und auf `Bereitstellen` > `Bereitstellen in GCE` zu klicken.

Google Cloud Engine

Anleitung: https://cloud.google.com/compute/docs/instances/create-start-instance?hl=de

Wichtig: Container muss mit GPU eingerichtet werden. Speicherplatz ab 50 GB.

Conn

```
gcloud compute config-ssh
```

Verbindung zu Google Cloud

Anleitung: https://cloud.google.com/compute/docs/instances/connecting-advanced?hl=de

Am einfachsten kann auf `SSH` > `Cloud Befehl anzeigen` auf dem laufenden Container geklickt werden. Man erhält einen ähnlichen string wie:

```
gcloud compute ssh --project=newsai-419410 --zone=us-central1-a newsai
```

Diesen kann man in die Shell kopieren. Jetzt öffnet sich eine SSH-Verbindung zu der GCE (Putty).

GPU auf GCE aktivieren

Anleitung:

- https://cloud.google.com/container-optimized-os/docs/how-to/run-gpus?hl=de
- https://cloud.google.com/container-optimized-os/docs/how-to/run-gpus?hl=de#e2e

Auch im Startscript in GCE möglich

```
sudo cos-extensions install gpu
```

```
# Make the driver installation path executable by re-mounting it.
sudo mount --bind /var/lib/nvidia /var/lib/nvidia
sudo mount -o remount,exec /var/lib/nvidia
/var/lib/nvidia/bin/nvidia-smi
```

Ggf. auch ohne sudo ausführen, wenn man sich bereits als root eingeloggt hat.

Docker Container mit GPU starten

Anleitung:

- https://cloud.google.com/container-optimized-os/docs/how-to/run-gpus?hl=de#identify-driver
- https://cloud.google.com/compute/docs/gpus/install-drivers-gpu?hl=de#minimum-driver
- https://developer.nvidia.com/cuda-11-4-0-download-archive?target_os=Linux&target_arch=x86_64&Distribution=Ubuntu&target_version=20.04&target_type=deb_network

Ggf. erst Docker Container stoppen:

```
docker ps
docker stop <container_name>
```

Docker Container mit GPU starten:

```
docker run --volume /var/lib/nvidia/lib64:/usr/local/nvidia/lib64 --volume
/var/lib/nvidia/bin:/usr/local/nvidia/bin --device /dev/nvidia0:/dev/nvidia0 --
device /dev/nvidia0:/dev/nvidia1 --device /dev/nvidia-uvm:/dev/nvidia-uvm --device
/dev/nvidiactl:/dev/nvidiactl us-central1-docker.pkg.dev/newsai-
419410/newsai/newsai
```

README.md 2024-08-16

Eintreten:

```
docker exec -it <container_name> bash
```

GPU Checken:

```
nvidia-smi
```

Text File erstellen:

```
apt-get install nano
```

```
nano test.py
```

```
import torch
print(torch.__version__)
print(torch.cuda.is_available())
```

Falls das nicht geht ist ggf. die falsche Torch Version installiert.

Torch deinstallieren

```
pip uninstall torch torchvision
```

Je nach CUDA Version muss die passende Torch Version installiert werden.: https://pytorch.org/get-started/locally/

Stable (2.2.2) > Linux > Pip > Cuda 11.8

```
pip3 install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu118
```

Jetzt sollte die GPU erkannt werden.

Python Script starten (SSH kann abgebrochen werden)

- https://unix.stackexchange.com/questions/362115/how-to-keep-a-python-script-running-when-i-close-putty

9 / 10

```
apt-get install tmux
```

```
tmux
python3 label_dataset.py
```

```
tmux attach
```

venv

Init

```
C:\Python310\python -m venv .venv
.\.venv\Scripts\activate
```

Unix:

```
source .venv/bin/activate
```

Install

```
pip3 install -r requirements.txt
pip3 install torch==2.2.0 torchvision==0.17.0 torchaudio==2.2.0 --index-url
https://download.pytorch.org/whl/cu121
```

Poetry

```
pip3 install poetry
poetry init
```

Probleme bie Poetry: PyTorch wird nicht in der richtigen Version installiert, deshalb wird es manuell in die venv installiert.

```
pip3 install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu121
```

Anhang 10 Anleitung zur Verwendung der NextJS Web-App

README.md 2024-08-16

Web App: Unbiased News

Voraussetzungen

- Node.js

Installation

NPM-Pakete installieren:

```
npm install
```

Starten

```
npm run dev
```

Testen

Dafür muss der Server auf `http://localhost:3000/` laufen.

Unit Tests (Vitest):

```
# Starten des Servers wenn er noch nicht läuft
npm run test
```

Mit UI:

```
npm run test:ui
```

End-to-End Tests (Playwright):

```
# Starten des Servers wenn er noch nicht läuft
npm run dev
```

```
# Starten der Tests
npx playwright test
```

1 / 2

README.md 2024-08-16

Mit UI:

```
npx playwright test --ui
```

Bauen

```
npm run build
```

Login

Die Webseite ist nicht öffentlich zugänglich. Eigentlich sollten nur oAuth-Provider wie Google oder GitHub verwendet werden. Da ich aber keine Zugangsdaten für einen solchen Provider zur Verfügung stellen kann, habe ich einen Dummy-Provider eingerichtet. Die Zugangsdaten sind:

- E-Mail: `test@sae.de`
- Passwort: `Pluck-Osmosis8-Unwarlike-Dean-Geology`

Des Weiteren wird im dev Modus kein Login benötigt. Dafür muss in der `.env` Datei die Variable `IS_DEV` auf `true` gesetzt werden.

Verwendete Technologien:

- Next.js
- NextAuth.js
- Prisma
- Tailwind CSS
- tRPC
- Playwright
- Vitest

2 / 2