Duale Hochschule Baden-Württemberg Mannheim

Projektabschlussdokumentation

Entwicklung eines NLP-Tools zur Textklassifikation und -erkennung

Studiengang Wirtschaftsinformatik

Studienrichtung Data Science

Verfasser:	Aymane Bouguern, Lukas Bonn, Jan Rüdt, Amina Uicker-Darwish
Matrikelnummern:	1552312, 5856761, 1737304, 8278962
Kurs:	WWI20DSA
Eingereicht:	27.07.2023

Inhaltsverzeichnis

Abbildungsverzeichnis iii

Abkürzungsverzeichnis iv

1 Projektauftrag und Anforderungen 1

2 Datengrundlage 3
2.1 Presidential Speeches . . . 3
2.2 CNN News . . . 4
2.3 Jurisdictions . . . 4
2.4 Literature . . . 5
2.5 Blogs . . . 6

3 Implementierung 7
3.1 Textzusammenfassung . . . 7
3.1.1 Referenzzusammenfassungen erstellen . . . 7
3.1.2 Zusammenfassungsfunktion erstellen . . . 7
3.1.3 Vorverarbeitungsschritte . . . 8
3.1.4 TextRank, LSA . . . 8
3.1.5 Evaluierung der Textzusammenfassung . . . 9
3.1.6 Wahl des ROUGE-1-Scores . . . 9
3.1.7 Ergebnisse . . . 10
3.2 Textklassifikation . . . 10
3.2.1 Datenvorverarbeitungsschritte . . . 10
3.2.2 Numerische Repräsentation der Klassen . . . 11
3.2.3 Weitere Vorverarbeitungsschritte . . . 11
3.2.4 Modellerstellung . . . 12
3.2.5 Modellevaluierung . . . 12
3.2.6 Erstellen der Klassifikationsfunktion und des Schwellwerts . . . 13
3.3 Webanwendung . . . 14
3.3.1 Architektur und Erstellen des Frontends . . . 14
3.3.2 Einbinden der Klassifikations- und Zusammenfassungsfunktion . . . 15
3.3.3 Evaluierung des Frontends . . . 16

4 Projekt-Recap 18
4.1 Erfüllung der funktionalen und nicht-funktionalen Anforderungen . . . 18
4.2 Allgemeine Herausforderungen . . . 20
4.3 Bewertung der Aufwandsschätzung nach GANTT-Chart und Meilensteinen 23
4.4 Lessons Learned . . . 24
4.5 Zusammenfassung . . . 25

Anhang **25**

A Anhang **26**
A.1 GANTT-Chart 27
A.2 Webanwendung 29
A.3 Umfrage Webanwendung 32

Literaturverzeichnis **35**

Abbildungsverzeichnis

Abbildung 1.1 Erstellter Projektstrukturplan 1

Abbildung 3.1 Vergleich der Texte pro Klasse vor dem Data Balancing *(Links)* und danach *(Rechts)* 11
Abbildung 3.2 Vergleich der ML-Modelle in Bezug auf die *Accuracy* auf den Testdaten 13
Abbildung 3.3 Architektur der Webanwendung 14

Abbildung A.1 GANTT-Chart (1/2) 27
Abbildung A.2 GANTT-Chart (2/2) 28
Abbildung A.3 StartPage der Webanwendung 29
Abbildung A.4 ResultPage der Webanwendung 30
Abbildung A.5 Wave-Tool vor den Anpassungen 30
Abbildung A.6 Fehler im Wave-Tool 31
Abbildung A.7 Wave-Tool nach den Anpassungen 31
Abbildung A.8 Umfrage (1/6) 32
Abbildung A.9 Umfrage (2/6) 32
Abbildung A.10 Umfrage (3/6) 33
Abbildung A.11 Umfrage (4/6) 33
Abbildung A.12 Umfrage (5/6) 33
Abbildung A.13 Umfrage (6/6) 34

Abkürzungsverzeichnis

ML	Machine Learning
LSA	Latent Semantic Indexing
SVM	Support Vector Machines
kNN	K-Nearest-Neighbor
SVD	Singular Value Decomposition
NLP	Natural Language Processing

1 Projektauftrag und Anforderungen

Ziel des dieser Dokumentation zugrundeliegenden Projektes war es, ein NLP-Tool zur Textklassifikation und -zusammenfassung zu entwickeln, um dadurch Nutzern die Möglichkeit zu geben, eingegebene Texte als eine von fünf Klassen zu identifizieren und diese in zusammengefasster Version unter Berücksichtigung einer wählbaren Kompressionsrate angezeigt zu bekommen. Ein kurzer Recap des Projektauftrags ist in Tabelle 1.1 dargestellt.

Projektname	Entwicklung eines NLP-Tools für die Klassifikation und das Zusammenfassen von Texten.
Auftraggeber	Die Auftraggeber waren die Dozenten Enzo Hilzinger und Michael Lang
Projektart	Forschungs- und Bildungsprojekt
Projektziel	Das Ziel des Projektes war es, ein NLP-Tool zur Textklassifikation und -zusammenfassung zu entwickeln, welches über ein Frontend erreichbar sein sollte.
Beteiligte Personen	Jan Rüdt, Amina Uicker-Darwish, Aymane Bouguern, Lukas Bonn

Tabelle 1.1: Recap Projektauftrag

Zur Gliederung des Projekts wurde ein Projektstrukturplan erstellt, welcher bereits in der Zwischenstandsdokumentation detailliert vorgestellt wurde (siehe Abbildung 1.1). Dieser besteht aus drei Bestandteilen, welche das Grundgerüst des Projekts bildeten.

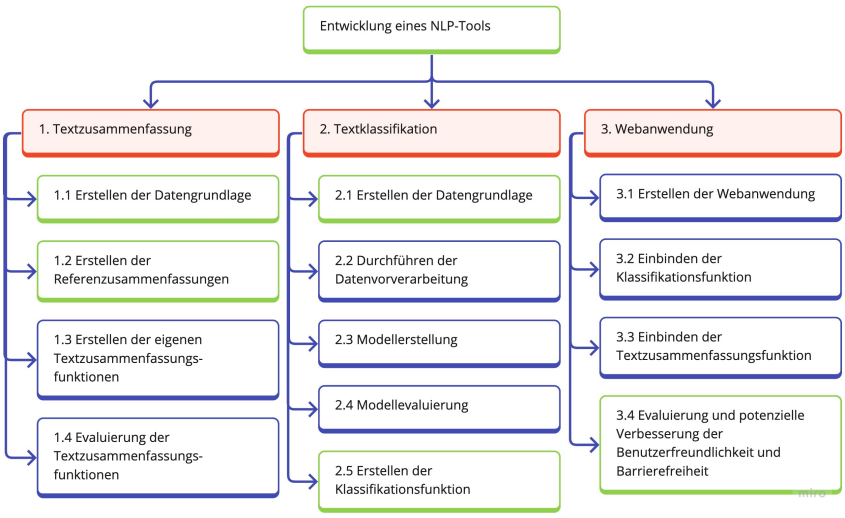


Abbildung 1.1: Erstellter Projektstrukturplan

Auf Basis der darin enthaltenen Arbeitspakete wurde zudem ein GANTT-Chart (siehe Abbildung A.1 und A.2) zur zeitlichen Planung des Projekts erstellt. Auch im Rahmen dieser Abschlussdokumentation wird sich an den Bestandteilen des Projektstrukturplans orientiert und die technische Implementierung dieser Bestandteile erläutert.

2 Datengrundlage

Zur Erfüllung der aufgestellten Anforderungen wurden fünf Oberkategorien von Texten als Datengrundlage herangezogen. In diesem Kapitel werden die gewählten Datensätze vorgestellt und die durchgeführten Textbereinigungs- und Vorverarbeitungsschritte näher erläutert. Folgende fünf Kategorien werden dabei im Rahmen des NLP-Tools unterschieden:

2.1 Presidential Speeches

Textinhalt	Politische Reden amerikanischer Präsidenten
Anzahl der in die Datengrundlage aufgenommenen Texte	801
Durchschnittliche Länge der Texte vor der Vorverarbeitung	4916 Wörter
Durchschnittliche Länge der Texte nach der Vorverarbeitung	651 Wörter

Tabelle 2.1: Information über den Datensatz der Klasse *Presidential Speeches*

In die Datengrundlage vor der Vorverarbeitung wurden nur politische Reden aufgenommen, welche eine Länge von mindestens 1100 Wörtern hatten, sodass aus diesen einheitliche Textsequenzen zwischen 500 und 800 Wörtern zum Ergänzen der Datengrundlage genutzt werden konnten. Die Position dieser Textsequenzen im Ursprungstext wurde dabei zufällig bestimmt. Zudem wurden Zeilenumbrüche in den Texten entfernt, sodass diese über eine zueinander ähnliche Struktur verfügen (Datenquelle: (vgl. University of Virginia, 2022)).

2.2 CNN News

Textinhalt	Nachrichtenartikel veröffentlicht von CNN
Anzahl der in die Datengrundlage aufgenommenen Texte	1002
Durchschnittliche Länge der Texte vor der Vorverarbeitung	595 Wörter
Durchschnittliche Länge der Texte nach der Vorverarbeitung	628 Wörter

Tabelle 2.2: Information über den Datensatz der Klasse *CNN News*

Von ursprünglich 1100 geladenen Texten wurden nur die Nachrichtenartikel in die Datengrundlage aufgenommen, welche über eine Länge von mindestens 250 Wörtern verfügten. Zudem wurde der Beginn der Nachrichtenartikel (z.B. „BAGHDAD, Iraq (CNN) –“) entfernt, um den eigentlichen Text und die darin enthaltenen Informationen in den Fokus zu stellen (Datenquelle: (vgl. Wolf et al., 2022)).

2.3 Jurisdictions

Textinhalt	Unterlagen zu juristischen Fällen des Obersten Gerichtshof des Vereinigten Königreichs
Anzahl der in die Datengrundlage aufgenommenen Texte	780
Durchschnittliche Länge der Texte vor der Vorverarbeitung	14464 Wörter
Durchschnittliche Länge der Texte nach der Vorverarbeitung	650 Wörter

Tabelle 2.3: Information über den Datensatz der Klasse *Jurisdictions*

Es wurden nur juristische Texte der Datengrundlage ergänzt, welche eine ursprüngliche Länge von mindestens 3000 Worten hatten, wobei aus diesen Textsequenzen mit einer Länge zwischen 500 und 800 Worten extrahiert wurden (Datenquelle: (vgl. Shukla et al., 2022)).

2.4 Literature

Textinhalt	Klassische Literatur in englischer Sprache
Anzahl der in die Datengrundlage aufgenommenen Texte	952
Durchschnittliche Länge der Texte vor der Vorverarbeitung	105257 Wörter
Durchschnittliche Länge der Texte nach der Vorverarbeitung	650 Wörter

Tabelle 2.4: Information über den Datensatz der Klasse *Literature*

Bevor die Texte des gewählten Gutenberg-Datensatzes genutzt werden konnten, wurden zunächst nur die Texte geladen, welche in englischer Sprache geschrieben wurden und über eine ursprüngliche Länge von mindestens 30000 Wörtern verfügten. Aus den Texten wurden Zeilenumbrüche und Mehrfachleerzeichen entfernt. Da in den Texten vereinzelt Seitenzahlen oder weitere für den Inhalt irrelevante Dinge in geschweiften Klammern und z.B. Informationen zu Abbildungen in eckigen Klammern vorhanden waren, wurden diese Elemente aus den Texten entfernt, wodurch eine Bereinigung der Texte stattfand.

Zudem wurde die Länge der Texte, wie auch innerhalb weiterer Kategorien, auf 500 bis 800 Worte gekürzt, um die Datengrundlage zu vereinheitlichen. Die angepassten Texte entsprechen dabei zufälligen Textsequenzen der ursprünglichen Texte, wobei bestimmt wurde, dass die ersten 5000 und die letzten 20000 Wörter der ursprünglichen Texte kein Teil der modifizierten Texte sein durften. Dies liegt daran, dass zu Beginn der ursprünglichen literarischen Texte häufig Elemente, wie Informationen über den Autor oder ein Inhaltsverzeichnis, vorhanden waren und am Ende der Texte teilweise Details zu Lizenzrechten, das Projekt Gutenberg oder andere Aspekte, welche keinen literarischen Inhalt haben, standen. Um nur den literarischen Inhalt der Texte in die Datengrundlage aufzunehmen, wurde diese Vorverarbeitung durchgeführt (Datenquelle: (vgl. Gerlach & Font-Clos, 2022)).

2.5 Blogs

Textinhalt	Blogartikel von Personen im Alter zwischen 13 und 47
Anzahl der in die Datengrundlage aufgenommenen Texte	1000
Durchschnittliche Länge der Texte vor der Vorverarbeitung	1185 Wörter
Durchschnittliche Länge der Texte nach der Vorverarbeitung	650 Wörter

Tabelle 2.5: Information über den Datensatz der Klasse *Blogs*

Es wurden Blogartikel ausgewählt, welche eine ursprüngliche Länge zwischen 1000 und 1500 Wörter hatten, sodass aus diesen Textsequenzen von rund 650 Wörtern extrahiert werden konnten (Datenquelle: (vgl. Tatman, 2017)).

3 Implementierung

In diesem Kapitel wird unter Berücksichtigung der Aspekte des in Kapitel 1 genannten Projektstrukturplans die Implementierung des NLP-Tools beschrieben. Dabei wird zunächst auf die Zusammenfassungsfunktion, anschließend auf die Klassifikationfunktion und abschließend auf die Webanwendung eingegangen.

3.1 Textzusammenfassung

In diesem Abschnitt liegt der Fokus auf der Erstellung und Evaluierung der Textzusammenfassungsfunktion.

3.1.1 Referenzzusammenfassungen erstellen

Um eine objektive Bewertung der eigenen Zusammenfassungsfunktion zu ermöglichen, wurden Referenzzusammenfassungen erstellt. Dies dient dazu, eine unabhängige Grundlage für den Vergleich der eigenen Zusammenfassungen zu haben. Dabei wurden verschiedene Ansätze wie TextRank, LSA und LexRank *(sumy-Bibliothek)* verwendet, um die Referenzzusammenfassungen zu generieren. Durch die Verwendung mehrerer Ansätze wurde versucht, eventuelle Bias-Effekte zu reduzieren und ein breiteres Spektrum an Zusammenfassungen zu erhalten. Dabei wurden die verschiedenen Ansätze prozentual gleich viel verwendet, um die Referenzzusammenfassungen zu erstellen.

3.1.2 Zusammenfassungsfunktion erstellen

Die Aufgabe bestand darin, eine Zusammenfassungsfunktion zu entwickeln, die einen gegebenen Text komprimieren kann. Um die Verwendung einer variablen Kompressionsrate zu ermöglichen, wurde ein extraktiver Ansatz gewählt, der relevante Sätze aus dem Text extrahiert, anstatt neue zu generieren (vgl. Mithbavkar & Chauhan, 2021).

3.1.3 Vorverarbeitungsschritte

Bevor die Zusammenfassungsfunktion angewendet werden konnte, mussten einige Vorverarbeitungsschritte durchgeführt werden. Dazu gehörten im Rahmen des Projekts die Entfernung von Stop-Words, das Lemmatisieren der Wörter, das Kleinschreiben der Texte und das Entfernen der Punctuations. Diese Schritte dienten dazu, die Texte für die anschließende Zusammenfassung nutzbar zu machen.

3.1.4 TextRank, LSA

Im Rahmen dieses Projektes wurden die beiden Algorithmen TextRank und LSA zur Umsetzung der Textzusammenfassung gewählt. Beide Ansätze wurden zunächst implementiert und auf dem Datensatz angewendet.

TextRank:

Textrank ist ein Algorithmus, der auf der Idee des PageRank-Algorithmus basiert, welcher von Google entwickelt wurde, um die Relevanz von Webseiten zu bewerten. Der Textrank-Algorithmus wird jedoch auf Texte angewendet, um die wichtigsten Sätze eines Textes zu identifizieren und eine Zusammenfassung zu erstellen (vgl. Joshi, 2023).

Dabei wird ein Graph erstellt, wobei jeder Satz als Knoten repräsentiert wird. Die Knoten sind miteinander durch Kanten verbunden, die die Ähnlichkeit zwischen den Texteinheiten darstellen. Diese Ähnlichkeit wird normalerweise anhand der Kosinus-Ähnlichkeit der Vektorrepräsentationen der Sätze berechnet (vgl. Mihalcea, 2004).

Die Wichtigkeit jedes Knotens im Graphen wird mithilfe des PageRank-Algorithmus oder ähnlicher Ranking-Verfahren bestimmt. PageRank bewertet die Bedeutung einer Seite basierend auf der Anzahl und Qualität der Verbindungen zu anderen Seiten im Web. Ähnlich bewertet Textrank die Wichtigkeit eines Satzes anhand seiner Verbindungen zu anderen Sätzen im Text (vgl. Jain, 2020).

LSA:

LSA ist eine Technik aus dem Bereich des maschinellen Lernens und des NLP (Natural Language Processing), welche verwendet wird, um semantische Beziehungen zwischen

Wörtern und Texten zu ermitteln. Zunächst wird eine Term-Dokument-Matrix erstellt, bei der die Texte im Dokument als Spalten und die Wörter als Zeilen repräsentiert werden. Jeder Eintrag in der Matrix gibt an, wie oft ein bestimmtes Wort in einem bestimmten Text vorkommt (vgl. Ozsoy et al., 2011).

Um die Dimensionalität der Term-Dokument-Matrix zu reduzieren und die zugrunde liegenden semantischen Strukturen zu erfassen, wird eine mathematische Technik namens Singulärwertzerlegung (Singular Value Decomposition, SVD) angewendet. SVD hilft dabei, die relevantesten Eigenschaften der Daten zu extrahieren (vgl. Chakravarthy, 2021).

Die SVD ermöglicht es, die Ähnlichkeiten zwischen Wörtern und Texten zu berechnen. Wörter und Texte, die ähnliche semantische Bedeutungen haben, werden im reduzierten Raum näher beieinander liegen (vgl. Ozsoy et al., 2011).

Um eine Zusammenfassung basierend darauf zu erstellen, werden die Sätze ausgewählt, die die meisten wichtigen semantischen Informationen des Originaltexts enthalten. Dies wird erreicht, indem man die Ähnlichkeiten der Sätze zu einem „Repräsentations"-Satz berechnet, der aus den wichtigsten semantischen Eigenschaften des gesamten Textes besteht (vgl. Ozsoy et al., 2011).

Sowohl Textrank als auch LSA ermöglichen eine effiziente Text-Zusammenfassung, wobei Textrank auf graphenbasierten Rankings basiert und LSA auf der mathematischen Dimensionsreduktion. Ihre Implementierung und der Vergleich helfen dabei, die bestmögliche Zusammenfassungsfunktion für das Projekt zu wählen.

3.1.5 Evaluierung der Textzusammenfassung

3.1.6 Wahl des ROUGE-1-Scores

Um die Qualität der erstellten Zusammenfassungen zu bewerten, wurde der Rouge-1 F1-Score verwendet. Der Rouge-1 F1-Score ist eine Metrik, die verwendet wird, um die Qualität einer Textzusammenfassung zu bewerten, indem sie die Übereinstimmung der generierten Zusammenfassung mit einer Referenzzusammenfassung auf Wortebene misst. Der Rouge-1 F1-Score kombiniert die beiden Maße Precision und Recall in einem einzigen Wert, der die Leistung der Zusammenfassung darstellt. Er ermöglicht es, Zusammenfassungen unterschiedlicher Längen miteinander zu vergleichen (vgl. „Rouge-score", 2022; Chiusano, 2022).

3.1.7 Ergebnisse

Im Rahmen der Evaluierung wurden die beiden Ansätze LSA und TextRank angewendet und ihre Leistung anhand des Rouge-1 F1-Scores bewertet. Basierend auf dem Ergebnis wurde der LSA-Algorithmus für die Zusammenfassungsfunktion ausgewählt.

Der LSA-Algorithmus erzielte einen F1-Score von 55%. Hingegen erreichte TextRank einen F1-Score von 49%. Mit einem etwas niedrigeren F1-Score von 49% zeigt der TextRank ebenfalls eine solide Leistung, jedoch leicht unter der von LSA.

3.2 Textklassifikation

In diesem Kapitel wird mit der Klassifikation auf die weitere wesentliche Funktionalität des NLP-Tools eingegangen. Dazu werden zunächst die durchgeführten Vorverarbeitungsschritte beschrieben, um anschließend die Modellerstellung- und evaluierung zu erläutern. Abgeschlossen wird mit der erstellten Klassifikationsfunktion, welche die Einbindung des gewählten Modells in die Webanwendung vorbereitet.

3.2.1 Datenvorverarbeitungsschritte

Im Folgenden werden die Datenvorverarbeitungsschritte erklärt, welche durchgeführt wurden, um das Trainieren der Modelle zu ermöglichen.

Data Balancing:

Bevor die Texte innerhalb der Datengrundlage zur Umsetzung der Klassifizierung genutzt werden konnten, mussten diese zunächst in Bezug auf die Anzahl der Texte pro Klasse gleichverteilt werden. Dies wurde über ein Downsampling durchgeführt, wobei die Kategorie „Jurisdictions“ mit der geringsten Anzahl von 780 Texten als Referenz genommen wurde. Aus den restlichen Kategorien wurden jeweils zufällig 780 Texte ausgewählt, um dadurch eine bestmögliche Klassifikation zu ermöglichen (siehe Abbildung 3.1).

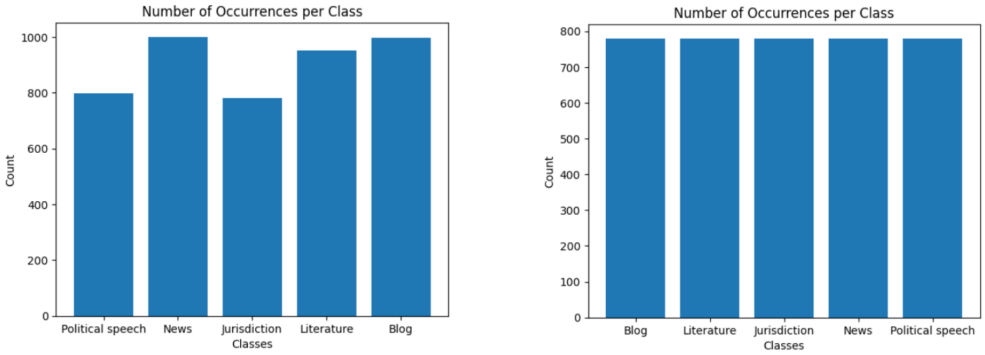


Abbildung 3.1: Vergleich der Texte pro Klasse vor dem Data Balancing *(Links)* und danach *(Rechts)*

3.2.2 Numerische Repräsentation der Klassen

Als weiteren Schritt der Vorverarbeitung wurden die fünf im zweiten Kapitel erläuterten Klassen numerisch repräsentiert, sodass eine Verarbeitung durch die genutzten Machine Learning-Modelle ermöglicht werden konnte. Dazu wurde ein Dictionary erstellt, welches für jede Klasse eine Zahl von null bis vier beinhaltet und dazu genutzt wurde, die Klassen in Textform innerhalb des zu nutzenden Datensatzes durch numerische Repräsentationen zu ersetzen.

3.2.3 Weitere Vorverarbeitungsschritte

Im Rahmen der weitergehenden Vorverarbeitung für die Nutzung der Daten in den Machine Learning-Modellen, ist zwischen den Schritten für die „klassischen“ Machine Learning-Modelle und denen für das BERT-Modell, als Ansatz des Transfer Learning, zu unterschieden. Als „klassische“ Machine Learning-Modelle wurden im Rahmen des Projekts SVM, kNN und Naive Bayes verwendet.

„Klassische“Machine Learning-Modelle - Vorverarbeitung:

Zur Nutzung in den „klassischen“ Machine Learning-Modellen wurden verschiedene Vorverarbeitungsschritte durchgeführt. Dabei erfolgte zunächst eine Kleinschreibung auf den Texten und eine Tokenisierung der Texte auf Wortebene, bevor die einzelnen Wörter lemmatisiert wurden. Um dies umzusetzen, war es notwendig, zunächst die einzelnen Wörter den zugehörigen Wortarten zuzuordnen *(POS-Tagging)*. Auch wurden in diesem Schritt Stop-Words sowie Zeichen, welche weder eine Zahl noch ein Buchstabe sind *(Non-alpha*

text), aus den Texten entfernt, um den Fokus auf den Inhalt der Texte zu legen. Darüber hinaus wurden die Texte anschließend mithilfe von TF-IDF in Form von Wortvektoren numerisch repräsentiert, um den Einsatz in den Modellen zu ermöglichen (vgl. Bedi, 2018).

BERT Modell - Vorverarbeitung:

Im Rahmen der Vorverarbeitung der Daten für die Nutzung des BERT-Modells wurden zunächst die Trainings- bzw. Validierungsdaten von den Testdaten zur späteren Durchführung der Evaluierung getrennt, wonach der BERT-Tokenizer definiert wurde, um die Texte in die gewünschte Form umzuwandeln. Durch diesen werden unter anderem Start- und Endtoken hinzugefügt ([CLS], [SEP]) und die Texte auf eine fixe Länge gebracht. Auch werden „Attention masks" zu den Texten ergänzt, welche dem BERT-Modell die Information geben, welche Token in den Texten es zu berücksichtigen gilt. Darüber hinaus wurden die Trainings- und die Validierungsdaten in einem Verhältnis 8 zu 2 aufgeteilt, wonach die Datensätze erstellt und ein Dataloader vorbereitet wurden (vgl. Albanese, 2022). Dieser ermöglicht es in Form von Batches über die Datensätze zu iterieren (vgl. Linux Foundation, n. d.), was für das Training des BERT-Modells genutzt wurde (vgl. Albanese, 2022).

3.2.4 Modellerstellung

Im weiteren Verlauf wurden die Modelle erstellt, bzw. bestehende Modelle geladen, um diese auf den vorverarbeiteten Daten weitergehend zu trainieren.

Für das SVM- und das kNN-Modell wurde GridSearch verwendet, um diese auf verschiedenen Hyperparameterkombinationen zu trainieren, um so die optimalen Parameter zu ermitteln. Im Zuge des Trainings des BERT-Modells wurden verschiedene Parameter angepasst, um dieses in Bezug auf die Klassifikationsaufgabe zu verbessern. Dies beinhaltete die Anzahl der gewählten Epochen, die Lernrate und die Größe der Batches. Im Zuge der einzelnen Trainingsepochen wurden zudem die im Vorhinein erstellten Validierungsdaten verwendet, um den Zwischenstand zu messen und einen ersten Eindruck der Modellleistung zu erhalten.

3.2.5 Modellevaluierung

Zum Evaluieren der Klassifikationsmodelle wurde sich für die *Accuracy* entschieden, um dadurch den Anteil der korrekt klassifizierten Texte im Vergleich zu allen klassifizierten

Texten zu ermitteln (vgl. Grandini et al., 2020, S. 3). Wie bereits beschrieben, wurde ein Testdatensatz erstellt, welcher in Folge des Trainings der Modelle zum Evaluieren dieser genutzt wurde.

Die Evaluierung der Klassifikationsmodelle kam dabei zu dem Ergebnis, dass das SVM-Modell mit einer Accuracy von **96,79%** am besten abschneidet. Das BERT-Modell lag mit 96,67% knapp dahinter (vgl. Abbildung 3.2). Aufgrund dieser Ergebnisse wurde sich für das **SVM-Modell** zur Nutzung in dem zu erstellenden NLP-Tool entschieden, um die Klassifikationsanforderung zu erfüllen. Das Modell wurde hierzu als Datei exportiert.

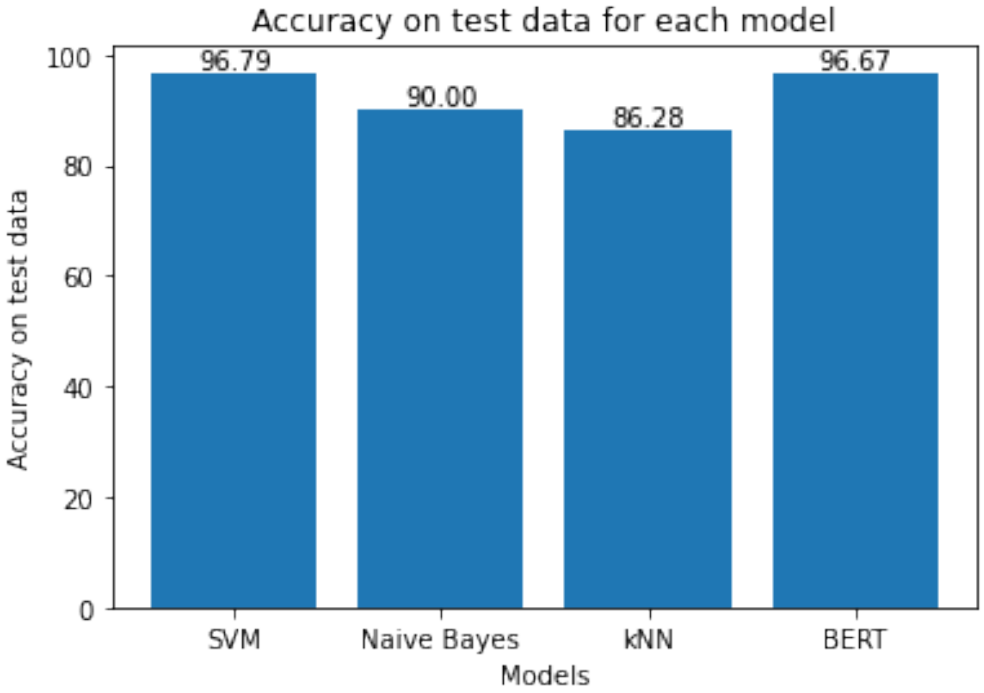


Abbildung 3.2: Vergleich der ML-Modelle in Bezug auf die *Accuracy* auf den Testdaten

3.2.6 Erstellen der Klassifikationsfunktion und des Schwellwerts

Zur Nutzung des im Rahmen der Evaluierung gewählten SVM-Modells innerhalb des NLP-Tools, wurde eine Klassifikationsfunktion erstellt. Diese nimmt die Benutzereingabe entgegen und führt gewisse Vorverarbeitungsschritte auf dem übergebenen Text aus, welche auch in der Vorbereitung des Modelltrainings vollzogen wurden. Anschließend wird in der

Funktion das SVM-Modell geladen und eine Klassifikationsvorhersage für den betrachteten Text getroffen (vgl. Bedi, 2018). Hierbei werden Wahrscheinlichkeiten zwischen 0 und 1 zurückgegeben. Als zusätzliche Komponente wurde in dem Tool ein Schwellwert festgelegt. Sollte die Wahrscheinlichkeit der vorhergesagten Klasse unterhalb dieses Schwellwerts liegen, wird zurückgegeben, dass eine Klassifikation auf dem Text nicht möglich war. Der Schwellwert wurde dabei auf Basis verschiedener durchgeführter Tests auf Probetexten auf 0,65 gesetzt.

3.3 Webanwendung

3.3.1 Architektur und Erstellen des Frontends

In diesem Projekt wurde für die Webanwendung das Flask-Framework verwendet. Dabei wird auf verschiedene HTML-Seiten mit CSS- und Java-Script-Inhalten sowie auf die Klassifikations- und Zusammenfassungsfunktion, welche in den folgenden Schritten eingebunden werden, zugegriffen (siehe Abbildung 3.3). Die Entscheidung für Flask als Framework wurde aufgrund seiner Fähigkeit getroffen, alle Anforderungen zu erfüllen und eine schnelle Implementierung zu ermöglichen. Zudem wurden in früheren Projekten positive Erfahrungen mit diesem Framework gemacht.

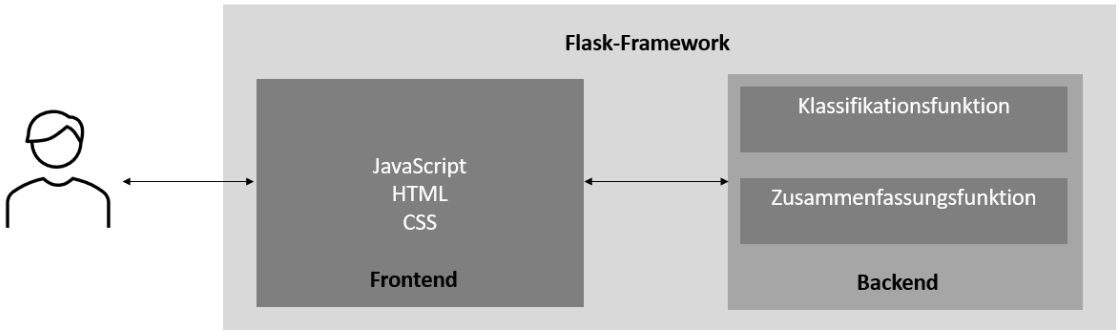


Abbildung 3.3: Architektur der Webanwendung

Die erste Website der Webanwendung ist die Start-Page (siehe Abbildung A.3). Hier kann der Benutzer Texte für die Klassifikation und Zusammenfassung in eine Textbox eingeben oder alternativ Dateien im ODT, TXT oder DOCX-Format hochladen. Die Integration der beiden Python-Packages DOCX und ODF ermöglicht das Lesen dieser Dateiformate. Eine

weitere Option besteht darin, Texte per Mikrofon einzusprechen. Ein entsprechender Button auf der Website startet die Aufnahme, und durch erneutes Drücken wird sie gestoppt. Die Aufnahme wird dann in Text umgewandelt, in einer TXT-Datei gespeichert und zum Download bereitgestellt. Der Benutzer kann anschließend die Datei über die Dateiuploadfunktion hochladen. Für die Aufnahme und Textumwandlung wird die Web Speech API verwendet, die in den meisten modernen Browsern wie Chrome, Firefox, Safari und Edge integriert ist.

Anschließend kann der Benutzer auswählen, ob er eine Klassifikation und/oder eine Zusammenfassung angezeigt bekommen möchte. Bei Bedarf kann er mit einem Schieberegler eine Kompressionsrate zwischen 20% und 80% festlegen.

Um die Barrierefreiheit zu verbessern, wurden auf der Landing-Page zwei Funktionen integriert. Ein Superzoom-Button ermöglicht es dem Benutzer, den Kontrast auf 300% einzustellen. Zusätzlich kann der Benutzer den Kontrast ändern, wobei der Hintergrund schwarz und die Schrift grün dargestellt werden, ähnlich dem hohen Kontrastmodus von Windows 10 (vgl. Microsoft, n. d.).

Nachdem der Benutzer den Text eingegeben und die Optionen ausgewählt hat, kann er die Eingabe durch Klicken des Submit-Buttons abschicken und gelangt zur Result-Page (siehe Abbildung A.4). Die Submit-Funktion wird in der app.py Datei aufgerufen, wo die Benutzereingaben verarbeitet und die entsprechenden Funktionen für die Klassifikation und Zusammenfassung aufgerufen werden. Auf der Result-Page wird der eingegebene Text in einem Fenster dargestellt. Anschließend werden, je nach getroffener Auswahl, die gewünschte und die tatsächliche Kompressionsrate auf Wortebene, die Zusammenfassung und/oder die vorhergesagte Klasse angezeigt. Auch hier sind die Superzoom- und Toggle-Kontrastfunktionen für ein barrierefreies Frontend integriert.

3.3.2 Einbinden der Klassifikations- und Zusammenfassungsfunktion

Die Klassifikations- und Zusammenfassungsfunktionen wurden in das Frontend der Webanwendung mithilfe von Flask integriert. Wenn der Benutzer die Eingabe abschickt, wird die Funktion submit() in der app.py Datei aufgerufen, die die Nutzereingaben verarbeitet. Zunächst wird überprüft, ob der Benutzer einen Text in das Textfeld eingegeben oder

eine Datei hochgeladen hat. Entsprechend wird der Textinhalt aus dem Textfeld extrahiert oder die hochgeladene Datei mithilfe der read_docx_file() bzw. der read_odt_file() Funktionen eingelesen. Zusätzlich kann eine txt-Datei eingelesen werden. Sollte sowohl ein Text eingegeben als auch eine Datei hochgeladen worden sein, gilt der Text innerhalb der Datei für die weitere Verarbeitung. Anschließend wird überprüft, welche Optionen der Benutzer ausgewählt hat (Klassifikation und/oder Zusammenfassung). Je nach Auswahl werden die entsprechenden Funktionen text_classification() und text_summary() aufgerufen, um die gewünschten Ergebnisse zu erzielen. Falls eine Klassifikation gewünscht ist, wird die vorhergesagte Klasse zurückgegeben und gegebenenfalls angezeigt. Wenn eine Zusammenfassung gewünscht ist, wird die Funktion text_summary() mit der entsprechenden Kompressionsrate aufgerufen und es werden die zusammengefasste Version des Textes sowie die tatsächliche Kompressionsrate ausgegeben und angezeigt. Die Ergebnisse werden anschließend in der Result-Page der Webapp präsentiert, wo der eingegebene Text, die Klassifikationsergebnisse und/oder die Zusammenfassung angezeigt werden.

3.3.3 Evaluierung des Frontends

Nach der Verordnung zur Schaffung barrierefreier Informationstechnik nach dem Behindertengleichstellungsgesetz: BITV 2.0 sind für ein barrierefreies Frontend bestimmte Anforderungen festgelegt (vgl. Maurice-Demourioux, 2021). Diese Anforderungen umfassen verschiedene Aspekte:

Wahrnehmbar:

- Es muss ausreichend Kontrast zwischen Text und Hintergrund gewährleistet sein, um die Lesbarkeit für alle Benutzer zu verbessern.
- Die Schriftgröße sollte anpassbar sein, um die Bedürfnisse verschiedener Benutzergruppen zu berücksichtigen, und die Seite muss responsiv gestaltet sein, um eine optimale Darstellung auf verschiedenen Geräten zu ermöglichen.
- Textalternativen für Bilder müssen vorhanden sein, um auch Benutzern mit Sehbehinderungen den Inhalt zugänglich zu machen.

Bedienbar:

- Alle Inhalte und Funktionalitäten müssen sowohl per Maus als auch per Tastatur bedienbar sein

Verständlich:

- Informationen und Bedienungselemente müssen klar und verständlich gestaltet sein, damit alle Benutzer die Funktionen und Inhalte der Website leicht erfassen können.
- Der Textinhalt muss gut lesbar und verständlich sein, damit Benutzer die Informationen leicht erfassen können.

Robust:

- Die Inhalte müssen von allen Benutzeragenten und assistiven Technologien wie Screenreadern interpretierbar sein, um die Zugänglichkeit für Menschen mit unterschiedlichen Hilfsmitteln zu gewährleisten.

Im Rahmen des Projekts wurden alle testbaren Anforderungen auf ihre Umsetzung geprüft und entsprechend implementiert. Zusätzlich erfolgte eine Evaluierung der Barrierefreiheit des Frontends mithilfe des Browser-Plugins „WAVE Evaluation“. Das Tool, welches von dem *Institute for Disability Research, Policy & Practice* entwickelt wurde, bewertet die Barrierefreiheit von Webseiten und gibt visuelles Feedback durch Symbole und Indikatoren auf den Seiten (vgl. Institute for Disability Research, Policy Practice, n. d.). Dabei wurden anfangs vier Fehler durch das WAVE-Tool aufgezeigt, die auf fehlende „form label“ und die nicht angegebene Seitensprache im HTML-Code zurückzuführen waren (siehe Abbildung A.5 und A.6).

Diese Mängel wurden im Anschluss behoben, sodass das Tool keine weiteren Fehler mehr angezeigt hat (siehe Abbildung A.7).

Zusätzlich wurde eine Umfrage zur Bewertung der Benutzerfreundlichkeit, Bedienbarkeit, Barrierefreiheit und des Verbesserungspotenzials der Webanwendung durchgeführt. Die Ergebnisse zeigten sehr positive Rückmeldungen in den meisten Bereichen. Allerdings wurde das Design der Website von einigen Befragten als durchschnittlich bis etwas ansprechend bewertet (siehe Abbildung A.8 - A.13). Daraufhin wurden Anpassungen am Layout und Design der Website vorgenommen, um die Seite optisch ansprechender zu gestalten und die Benutzererfahrung weiter zu verbessern.

4 Projekt-Recap

4.1 Erfüllung der funktionalen und nicht-funktionalen Anforderungen

Die Anforderungen an das Hochschulprojekt wurden in funktionale und nicht-funktionale Anforderungen untergliedert. Die detaillierte Beschreibung der einzelnen Anforderungen im Kontext des Projekts wurden bereits in der Dokumentation „Entwicklung eines NLP-Tools zur Textzusammenfassung und -klassifizierung“ in Kapitel 2 „Lasten- und Pflichtenheft“ niedergeschrieben. Zusätzlich zu den zu diesem Zeitpunkt definierten Anforderungen sind weitere Anforderungen auf Wunsch der Stakeholder hinzugekommen. Diese sind in Tabelle 4.1 farblich (gelb) markiert. Die folgenden Tabellen (Tabelle 4.1 und Tabelle 4.2) dienen dazu, eine Übersicht über die gesamten Anforderungen des Projekts zu geben und die Erfüllung bzw. nicht-Erfüllung abschließend zu bewerten. Zur Übersichtlichkeit wurden die funktionalen Anforderungen jeweils entsprechend einer Kategorie aus dem Projektstrukturplan zugeordnet oder, wenn sie allgemein für das Projekt gelten, der Kategorie „Allgemein“ zugeordnet. Die nicht-funktionalen Anforderungen beziehen sich auf das gesamte Endergebnis und sind keiner spezifischen Kategorie zugeordnet. Die Spalte „Dokumentation der Anforderung“ in beiden Tabellen beinhaltet jeweils den Abschnitt in diesem Dokument, in dem detailliert beschrieben wird, wie die entsprechenden Anforderungen umgesetzt wurden.

Funktionale Anforderungen			
Kategorie	Anforderung	Erfüllung	Dokumentation der Anforderung (Kapitel)
Zusammenfassung	Zusammenfassungsfunktion	ja	3.1.2
	Kompressionsrate 20%-80%	ja	3.1.2
	Englische Sprache	ja	3.1.2
	Zusammenfassung unterschiedlicher Textarten	ja	2
Klassifikation	mind. 5 Klassen als Oberkategorien	ja	2
	Auswahl geeigneter Modelle und Algorithmen	ja	3.2.4
	Klassifikation englischer Texte	ja	2
Webanwendung	Eingabe unterschiedlicher Textformate	ja	3.3.1
	Auswahl der gewünschten Kompressionsrate	ja	3.3.1
	Anzeigen der Resultate	ja	3.3.1
	Barrierefreiheit	ja	3.3.1
Generell	Mehrstufiger Ansatz	ja	3.1.2, 3.2.4, 3.3.1
	Eigenständige Auswahl der Toolchain	ja	3.1.2, 3.3.1, 3.2.4

Tabelle 4.1: Erfüllung der funktionalen Anforderungen

Nicht Funktionale Anforderungen		
Anforderungen	Erfüllung	Dokumentation der Anforderung
Leistung	Ja	3.1.5, 3.2.5
Benutzerfreundlichkeit	Ja	3.3.3
Nachvollziehbarkeit	Ja	Gesamte Dokumentation

Tabelle 4.2: Erfüllung nicht funktionaler Anforderungen

Zusammenfassend kann festgestellt werden, dass alle definierten Anforderungen im Rahmen des Hochschulprojekts umgesetzt werden konnten.

Darüber hinaus konnten weitere von den Stakeholdern geäußerte Wünsche erfüllt werden, die zuvor nicht als Anforderung definiert wurden und als zusätzliche Funktionalitäten angesehen werden können. Ein Beispiel hierfür ist eine Funktionalität in der Webanwendung, die es dem Benutzer ermöglicht Text über eine Spracheingabefunktion einzulesen, wie bereits in Kapitel 3.3.1 erläutert wurde.

4.2 Allgemeine Herausforderungen

Mit dem Ziel, die in Tabelle 4.1 und 4.2 aufgelisteten Anforderungen zu erfüllen, ist das Projektteam einigen Herausforderungen und Problemen begegnet. In diesem Abschnitt werden die wichtigsten Herausforderungen beschrieben und darauf verwiesen, welcher Lösungsansätze vom Projekteam gewählt wurde.

1. Auswahl eines Ansatzes für die Textzusammenfassung
Die Herausforderung bestand darin, einen passenden Ansatz für die Textzusammenfassung zu wählen, der den Anforderungen des Projekts gerecht wird, wie beispielsweise eine Zusammenfassung auf Basis einer gegebenen Kompressionsrate. Die Entscheidung zwischen einem extraktiven oder abstraktiven Ansatz war an dieser Stelle von großer Bedeutung. Beides sind Möglichkeiten zur Textzusammenfassung, aber grundlegend unterschiedliche Konzepte. Nach den ersten Recherchen und Untersuchungen stellten wir fest, dass sich die Umsetzung der Kompressionsrate mit einem extraktiven Ansatz möglicherweise einfacher gestalten könnte. Bei diesem Ansatz besteht die Zusammenfassung aus ausgewählten Sätzen oder Abschnitten des Originaltextes (vgl. Mithbavkar & Chauhan, 2021). Dadurch ist es leichter, die Länge und Kompression der Zusammenfassung zu kontrollieren, da die Sätze bereits in der ursprünglichen Form vorliegen und nicht neu generiert werden müssen, wie es bei einem abstraktiven Ansatz der Fall gewesen wäre (vgl. Mithbavkar & Chauhan, 2021). Ein weiteres Kriterium bei der Entscheidung zwischen einem extraktiven oder abstraktiven Ansatz war es, eine geeignete Evaluierungsmethode für die Funktionalität der Textzusammenfassung zu finden. Denn um die Qualität und Genauigkeit der Zusammenfassungen zu bewerten, waren Referenzzusammenfassungen erforderlich, die als Vergleichsgrundlage dienen. Ausführlicher wurde dieser Umstand bereits in Kapitel 3.1.2 und 3.1.1 beschrieben.

2. Datenerstellung

Es existierten einige Anforderungen an die Datengrundlage, wie beispielsweise im vorherigen Abschnitt angesprochen wurde. Dazu gehörte das Vorhandensein von passenden Referenzzusammenfassungen für jeden Text, um eine zuverlässige Evaluierung der Textzusammenfassungen zu ermöglichen. Des Weiteren mussten die Texte eine angemessene Mindestlänge haben, um sinnvolle Zusammenfassungen zu generieren. Gleichzeitig durften die Texte jedoch nicht zu lang sein, da die verfügbare Hardwarekapazität ihre Verarbeitungsgrenzen hatte. Darüber hinaus war es notwendig, dass die Texte aus mindestens fünf verschiedenen Oberkategorien stammten. Die Suche nach einer bereits existierenden Datengrundlage, die all diese Anforderungen erfüllte und öffentlich zugänglich war, erwies sich als erfolglos. Daher musste eine eigene Datengrundlage erstellt werden, was eine enorm anspruchsvolle Aufgabe darstellte. Es waren mehrere Schritte erforderlich, um dies zu erreichen. Zunächst mussten fünf Datensätze mit geeigneten Texten und passender Länge gefunden werden. Zusätzlich mussten geeignete Tools gefunden werden, um die Referenzzusammenfassungen zu generieren. Schließlich war es von entscheidender Bedeutung, sicherzustellen, dass die erstellte Datengrundlage keinen Bias aufwies. Dafür mussten die Texte unterschiedliche Längen aufweisen und die Referenzzusammenfassungen basierend auf verschiedenen Kompressionsraten erstellt werden. All diese Schritte erforderten viel Zeit, Mühe und Sorgfalt, um eine qualitativ hochwertige Datengrundlage zu schaffen, die den spezifischen Anforderungen gerecht wurde. In Kapitel 2 wurde genauer erläutert wie die Datengrundlage schließlich generiert wurde.

3. Umgang mit Evaluierungsergebnissen bei der Textzusammenfassung

Bereits während der Konzeptionsphase des Projekts war absehbar, dass die Evaluierungsergebnisse möglicherweise nicht den gewünschten Anforderungen entsprechen würden. Diese Unsicherheit resultierte schätzungsweise aus der teilweisen Verzerrung (Bias) der Datengrundlage durch die verwendeten Tools zur Erstellung der Referenzzusammenfassungen und der gewählten Kompressionsrate. Bei der Erstellung der Datengrundlage wurde versucht, diesem Bias entgegenzuwirken, indem unterschiedliche Tools und Kompressionsraten zufällig angewendet wurden. Trotz dieser Maßnahmen konnte eine vollständige Beseitigung des Bias nicht erreicht werden.

Die Herausforderung bestand darin, diesen Hintergrund zu erkennen und die Evaluierungsergebnisse nicht vorschnell als negativ zu bewerten, sondern sie im Zusammenhang mit diesem Bias zu interpretieren. Es war wichtig, die potenzielle Verzerrung als Teil des Gesamtbildes zu betrachten und die Ergebnisse entsprechend zu kontextualisieren.

4. Barrierefreies Frontend

Die kurzfristige Anforderung seitens der Stakeholder nach einem barrierefreien Frontend stellte eine bedeutende Herausforderung dar. Das Projektteam musste flexibel auf diese neue Anforderung reagieren, da das Design und die Funktionalitäten der Webanwendung bereits vorhanden waren und nun zusätzliche barrierefreie Funktionen integriert werden mussten. Es bestand jedoch Unklarheit darüber, wie genau ein barrierefreies Frontend definiert war und wie es innerhalb eines Hochschulprojekts umgesetzt werden konnte. Diese Aufgabe war für das Projektteam völlig neu, und zusätzlich musste die Barrierefreiheit noch evaluiert werden.

Um dieser Herausforderung gerecht zu werden, war eine umfangreiche Recherche erforderlich. Zudem war ein detaillierter Austausch mit den Stakeholdern von großer Bedeutung, um deren genaue Vorstellungen dieser Anforderungen zu verstehen. Durch diese Maßnahmen konnte das Projektteam die notwendigen Informationen sammeln und einen klaren Rahmen für die Implementierung des barrierefreien Frontends schaffen. Kapitel 3.3.1 behandelt diese Thematik im Detail.

5. Einbindung der Textzusammenfassung

Die Herausforderung bestand darin, eine beständige Fehlermeldung zu beheben, die bei der Einbindung der ausgewählten Textzusammenfassungsfunktion (LSA-basiert) auftrat. Diese Funktion konnte lokal auf dem Rechner erfolgreich genutzt werden, um Textzusammenfassungen zu erzeugen. Doch bei ihrer Integration in die Webanwendung stieß das Team auf eine bis dato unbekannte Fehlermeldung. Bei der Fehlersuche, die den Quellcode Schritt für Schritt ausführte und unterbrach, um den Ursprung des Fehlers zu identifizieren, stellte man fest, dass der Fehler auf eine Diskrepanz zwischen der Webanwendung und der Funktion zurückzuführen war: Die Kompressionsrate wurde in beiden Teilen in unterschiedlichen Formaten angegeben. Diese Inkonsistenz zwischen der Webanwendung und der lokalen Funktion erwies sich als erhebliche Herausforderung. Nach mehreren Versuchen gelang es schließlich, diese Unstimmigkeit zu erkennen und entsprechend zu korrigieren.

4.3 Bewertung der Aufwandsschätzung nach GANTT-Chart und Meilensteinen

Ein Teil des Hochschulprojekts bestand darin, die Erarbeitung der einzelnen Arbeitspakete, welche bereits im detaillierten Projektstrukturplan beschrieben wurden, im Rahmen eines GANTT-Charts zeitlich einzuschätzen und den Projektmitgliedern zuzuordnen. In diesem Abschnitt der Abschlussdokumentation wird nun rückblickend bewertet, ob der Zeit- und Arbeitsplan nach dem GANTT-Chart (Abb. A.1 in der Umsetzung tatsächlich eingehalten werden konnte und welche Abweichungen es gegeben hat.

Insgesamt wurde der vorgegebene Zeitrahmen nicht überschritten, und die Anforderungen konnten innerhalb der gegebenen Zeit erfüllt werden. Dennoch wurde festgestellt, dass das GANTT-Diagramm nur begrenzte Möglichkeiten zur Parallelisierung der Aufgaben bot. Wenn mehr Spielraum dagewesen wäre, hätte die Arbeitsweise noch effizienter gestaltet werden können. Die mangelnde Parallelisierung der Aufgaben resultierte hauptsächlich aus dem Umstand, dass ein erheblicher Zeitaufwand für die Beschaffung der Datengrundlage erforderlich war. Diese Herausforderung wurde bereits in Abschnitt 4.2 näher erläutert. Nachdem bereits eine Datengrundlage erstellt wurde, ergab sich eine Anforderung seitens der Stakeholder in Bezug auf die Kategorien, was dazu führte, dass diese neu erstellt werden musste. Da die Datengrundlage nahezu für alle Arbeitspakete eine Voraussetzung bildete, konnte während dieser Phase nur wenig parallelisiert werden.

Im Anschluss an die Phase der Datengrundlagen konnten die Arbeitspakete sehr gut unter den Projektteilnehmern aufgeteilt werden, was eine effektive Parallelisierung der Aufgaben ermöglichte. Dies führte zu einer Steigerung der Arbeitsgeschwindigkeit und half dabei, den Zeitplan einzuhalten. Besonders ab dem Zeitpunkt im GANTT-Diagramm nach dem 26. Mai traten nur wenige Probleme auf, und die Probleme, die auftraten, waren nicht von anderen Arbeitspaketen abhängig.

Eine der wenigen größeren Schwierigkeiten trat nach diesem Zeitpunkt im Arbeitspaket „Einbinden der Textzusammenfassungsfunktion“ auf. Diese resultierten hauptsächlich aus einer wiederholten Fehlermeldung, die bei der Ausführung der Funktion innerhalb der Webanwendung generiert wurde. Details dazu finden sich im Abschnitt 4.2. Trotz dieser Verzögerung war die fristgerechte Fertigstellung des Gesamtprojekts nicht gefährdet, jedoch konnte das betreffende Arbeitspaket nicht im vorgesehenen Zeitfenster abgeschlossen werden.

Schließlich konnte das Team schnell und effizient arbeiten, um sowohl im Zeitrahmen zu bleiben als auch die Zeit, die durch die vorherige Verzögerung bei der Datengrundlage verloren gegangen war, wieder aufzuholen.

4.4 Lessons Learned

Basierend auf der umfangreichen Dokumentation des Projektabschlusses können wichtige „Lessons Learned" aus dem Projekt extrahiert werden. Diese Lessons Learned stellen wertvolle Erkenntnisse und Erfahrungen dar, die während des Projekts gesammelt wurden und helfen können, zukünftige Projekte dieser Art zu verbessern. Durch die Analyse der Projektunterlagen, einschließlich des Gantt-Charts, können Erkenntnisse gewonnen werden, die zur Vermeidung ähnlicher Fehler oder Engpässe in zukünftigen Projekten dienen.

Eine wichtige „Lesson Learned" aus dem Projekt ist die Bedeutung einer detaillierten und frühzeitigen Abstimmung mit den Stakeholdern die in diesem Projekt im Rahmen der Erstellung der Datengrundlage auftauchte. Da diese Grundlage eine Voraussetzung für das gesamte Projekt darstellte, führten Probleme in diesem Bereich zu Zeitverzögerungen. Für zukünftige Projekte sollten daher von Anfang an intensive Gespräche mit den Stakeholdern geführt werden, um ihre Anforderungen und Erwartungen in Bezug auf die Datengrundlage oder anderweitige grundlegende Anforderungen korrekt zu verstehen.

Ein weitere Lessons-Learned aus diesem Projekt ist die Notwendigkeit, Absprachen unter den Teammitgliedern schriftlich festzuhalten. In einem konkreten Fall kam es aufgrund von Kommunikationsschwierigkeiten dazu, dass zwei Teammitglieder, welche dem gleichen Arbeitspaket zugewiesen wurden, gleichzeitig an derselben Aufgabe arbeiteten. Dies führte in einem Fall zu Doppelarbeit und ineffizienter Ressourcennutzung. Diese Erfahrung verdeutlicht die Bedeutung der schriftlichen Dokumentation von Absprachen, um Missverständnisse zu vermeiden und eine klare Aufgabenverteilung sicherzustellen.

Zusätzlich zu den beiden genannten Lessons-Learned ist zu betonen, dass die Zusammenarbeit innerhalb des Teams sehr gut funktioniert hat. Dieser positive Aspekt sollte ebenfalls hervorgehoben werden.

Ein Beispiel hierfür ist die Etablierung von Wochenzielen innerhalb der Arbeitspakete, die bereits einen zeitlichen Rahmen hatten. Es war äußerst vorteilhaft, wöchentliche Treffen abzuhalten, um den aktuellen Status und den Fortschritt gemeinsam zu besprechen und zu berichten, ob das Wochenziel des jeweiligen Teammitglieds erreicht wurde. Die klare

Festlegung von Wochenzielen und die regelmäßigen Treffen haben ermöglicht, einen besseren Überblick über den Fortschritt des Projekts zu behalten und rechtzeitig Anpassungen vorzunehmen, wenn nötig. Diese Herangehensweise hat nicht nur geholfen, die Effizienz zu steigern, sondern auch die Zusammenarbeit im Team zu stärken.

4.5 Zusammenfassung

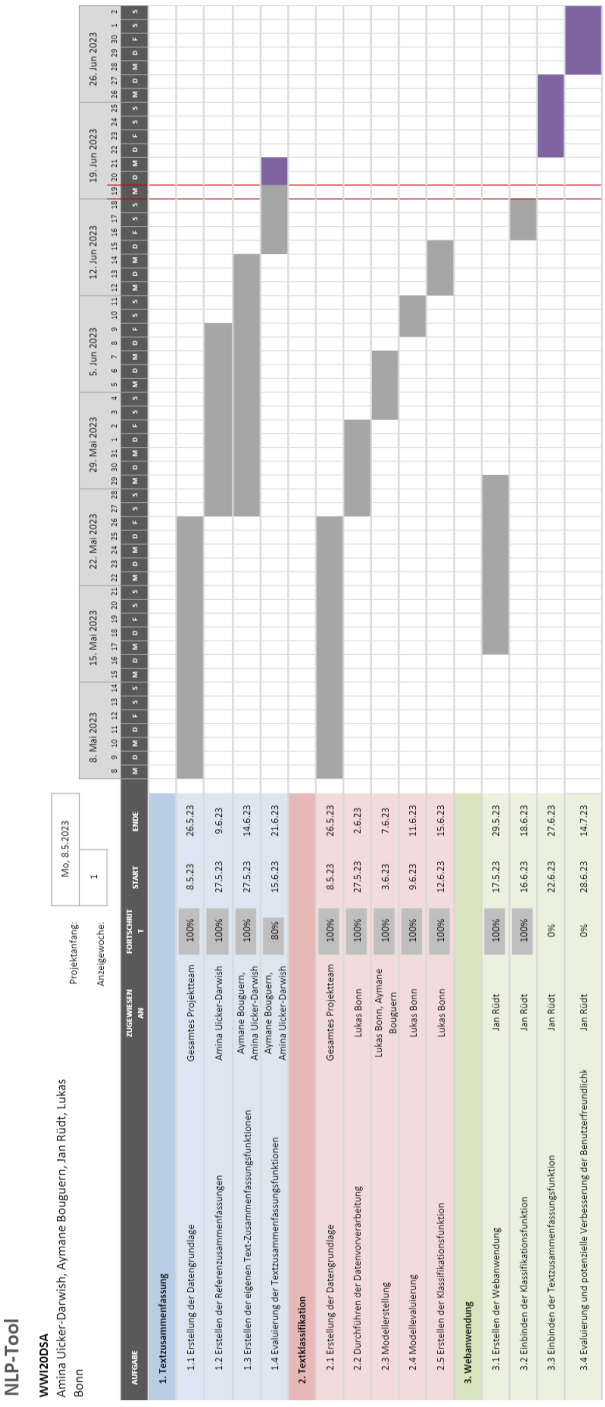
Die erfolgreiche Realisierung des Projekts zur Textzusammenfassung und -klassifikation von fünf Textkategorien stellt einen bedeutsamen Erfolg dar. Dazu wurde eine Support Vector Machine (SVM) und eine auf Latent Semantic Analysis (LSA) basierende Funktion, effizient eingesetzt.

Die SVM zur Klassifizierung wies letztlich eine Accuracy von 96,79% auf. Parallel dazu wurde die LSA-basierte Funktion zur Textzusammenfassung genutzt, die einen soliden Rouge F1-Score von 55% erreichte.

Die Implementierung dieser Modelle resultierte in einer barrierefreien Anwendung, die in den Bereichen Textzusammenfassung und Textklassifikation eine angemessene Leistungen erbringt. Dieser Erfolg bietet eine solide Grundlage für zukünftige Innovationen.

A Anhang

A.1 GANTT-Chart



AUFGABE	ZUGEWIESEN AN	FORTSCHRITT	START	ENDE
1. Textzusammenfassung				
1.1 Erstellung der Datengrundlage	Gesamtes Projektteam	100%	8.5.23	26.5.23
1.2 Erstellen der Referenzzusammenfassungen	Amina Ucker-Darwish	100%	27.5.23	9.6.23
1.3 Erstellen der eigenen Text-Zusammenfassungsfunktionen	Aymane Bouguern, Amina Ucker-Darwish	100%	27.5.23	14.6.23
1.4 Evaluierung der Textzusammenfassungsfunktionen	Aymane Bouguern, Amina Ucker-Darwish	80%	15.6.23	21.6.23
2. Textklassifikation				
2.1 Erstellung der Datengrundlage	Gesamtes Projektteam	100%	8.5.23	26.5.23
2.2 Durchführen der Datenvorverarbeitung	Lukas Bonn	100%	27.5.23	2.6.23
2.3 Modellerstellung	Lukas Bonn, Aymane Bouguern	100%	3.6.23	7.6.23
2.4 Modellevaluierung	Lukas Bonn	100%	9.6.23	11.6.23
2.5 Erstellen der Klassifikationsfunktion	Lukas Bonn	100%	12.6.23	15.6.23
3. Webanwendung				
3.1 Erstellen der Webanwendung	Jan Rüdt	100%	17.5.23	29.5.23
3.2 Einbinden der Klassifikationsfunktion	Jan Rüdt	100%	16.6.23	18.6.23
3.3 Einbinden der Textzusammenfassungsfunktion	Jan Rüdt	0%	22.6.23	27.6.23
3.4 Evaluierung und potenzielle Verbesserung der Benutzerfreundlichk	Jan Rüdt	0%	28.6.23	14.7.23

Abbildung A.1: GANTT-Chart (1/2)

AUFGABE	ZUGEWIESEN AN	FORTSCHRITT	START	ENDE
1. Textzusammenfassung				
1.1 Erstellung der Datengrundlage	Gesamtes Projektteam	100%	8.5.23	26.5.23
1.2 Erstellen der Referenzzusammenfassungen	Amina Ucker-Darwish	100%	27.5.23	9.6.23
1.3 Erstellen der eigenen Text-Zusammenfassungsfunktionen	Aymane Bouguern, Amina Ucker-Darwish	100%	27.5.23	14.6.23
1.4 Evaluierung der Textzusammenfassungsfunktionen	Aymane Bouguern, Amina Ucker-Darwish	80%	15.6.23	21.6.23
2. Textklassifikation				
2.1 Erstellung der Datengrundlage	Gesamtes Projektteam	100%	8.5.23	26.5.23
2.2 Durchführen der Datenvorverarbeitung	Lukas Bonn	100%	27.5.23	2.6.23
2.3 Modellerstellung	Lukas Bonn, Aymane Bouguern	100%	3.6.23	7.6.23
2.4 Modellevaluierung	Lukas Bonn	100%	9.6.23	11.6.23
2.5 Erstellen der Klassifikationsfunktion	Lukas Bonn	100%	12.6.23	15.6.23
3. Webanwendung				
3.1 Erstellen der Webanwendung	Jan Rüdt	100%	17.5.23	29.5.23
3.2 Einbinden der Klassifikationsfunktion	Jan Rüdt	100%	16.6.23	18.6.23
3.3 Einbinden der Textzusammenfassungsfunktion	Jan Rüdt	0%	22.6.23	27.6.23
3.4 Evaluierung und potenzielle Verbesserung der Benutzerfreundlichk	Jan Rüdt	0%	28.6.23	14.7.23

Abbildung A.2: GANTT-Chart (2/2)

A.2 Webanwendung

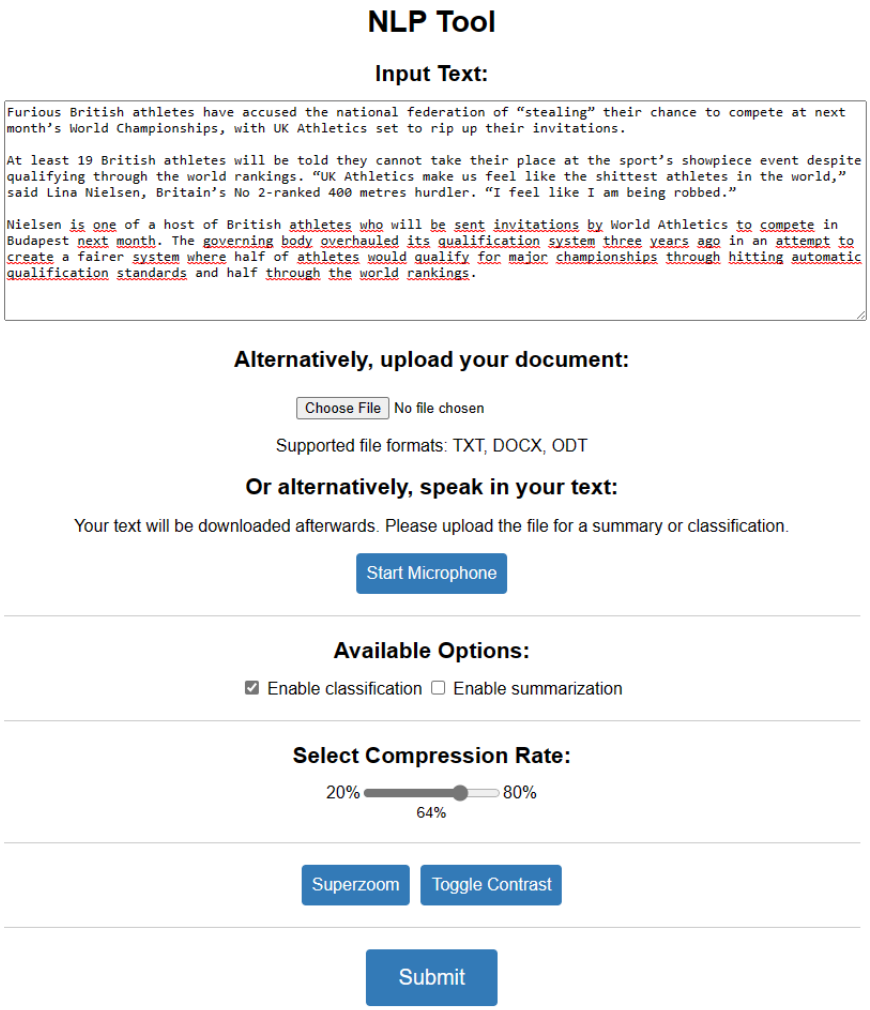


Abbildung A.3: StartPage der Webanwendung

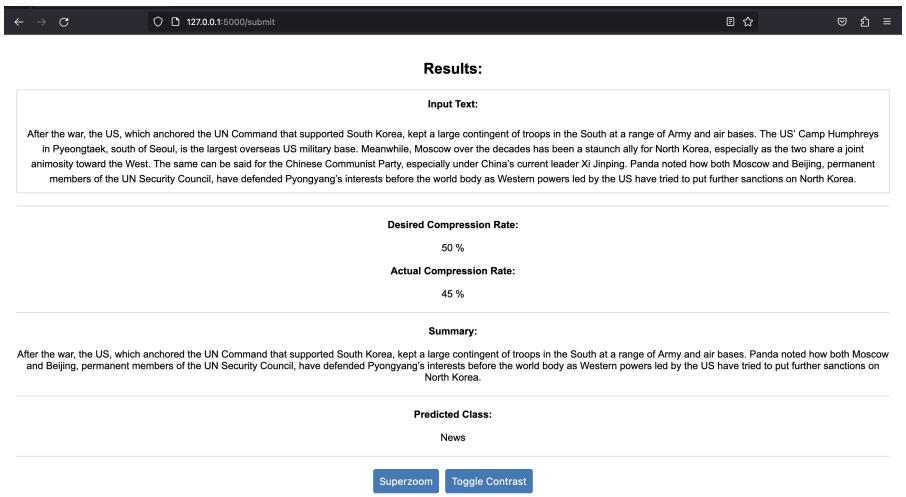


Abbildung A.4: ResultPage der Webanwendung

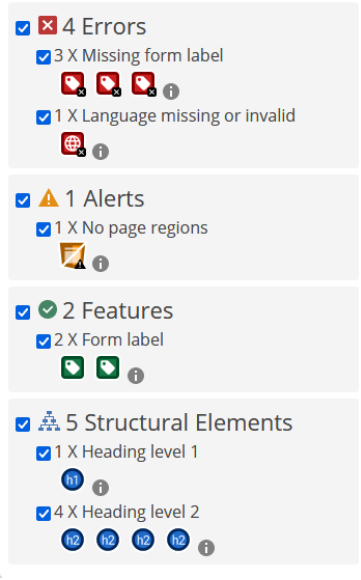


Abbildung A.5: Wave-Tool vor den Anpassungen

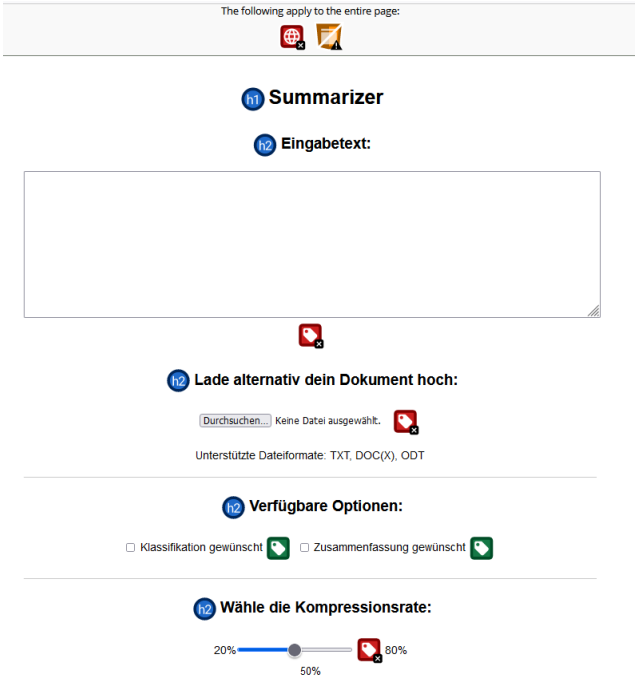


Abbildung A.6: Fehler im Wave-Tool

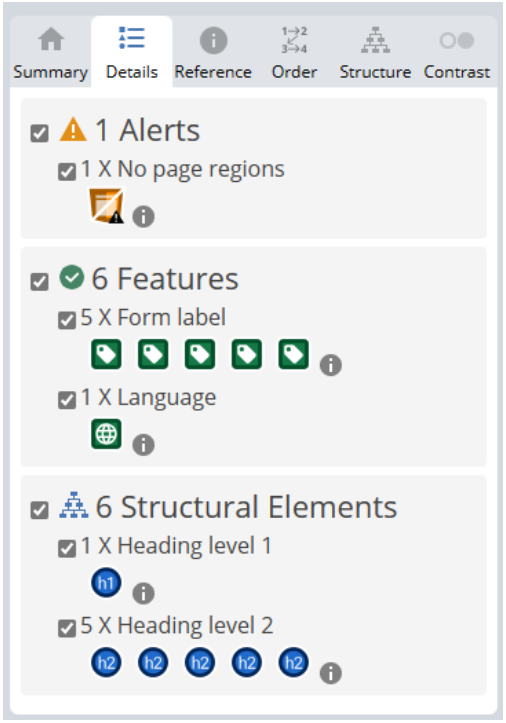


Abbildung A.7: Wave-Tool nach den Anpassungen

A.3 Umfrage Webanwendung

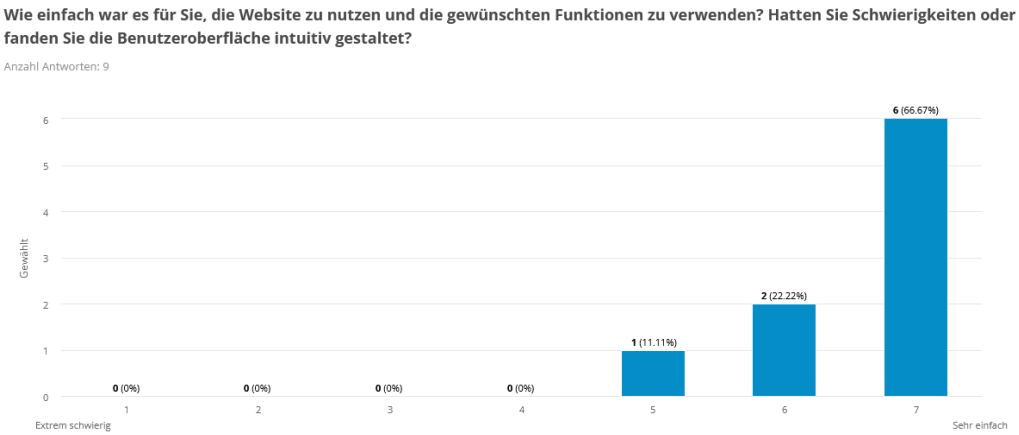


Abbildung A.8: Umfrage (1/6)

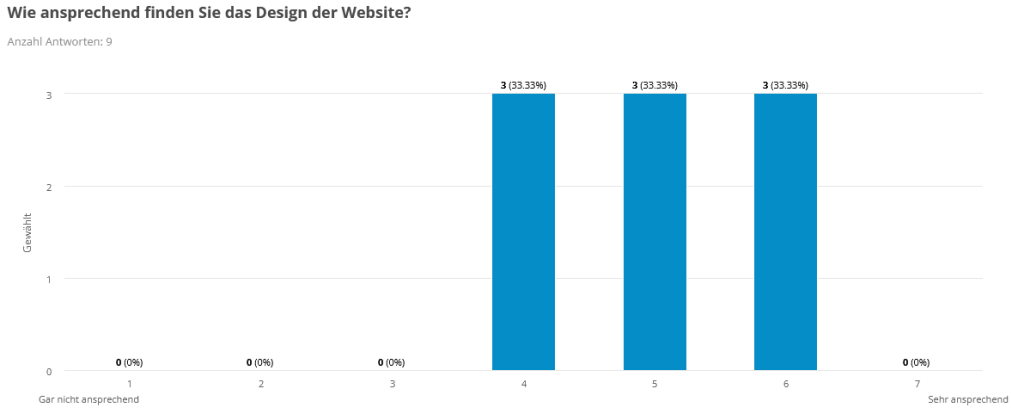


Abbildung A.9: Umfrage (2/6)

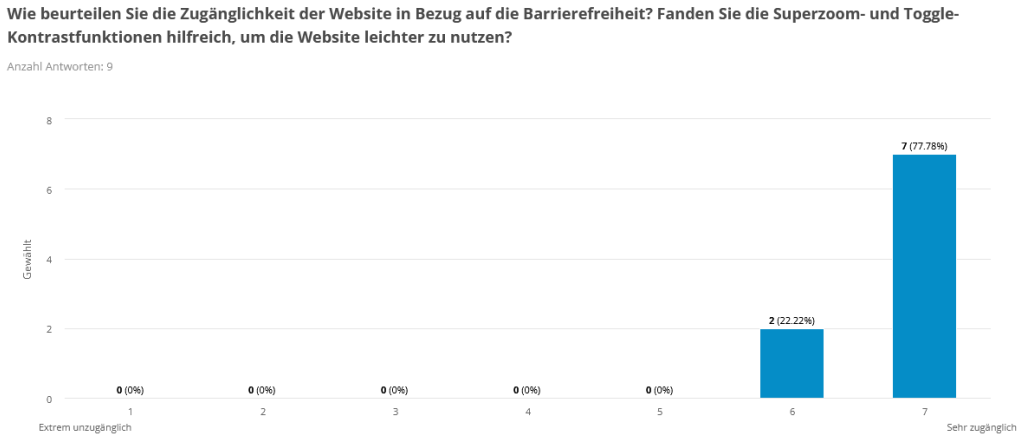


Abbildung A.10: Umfrage (3/6)

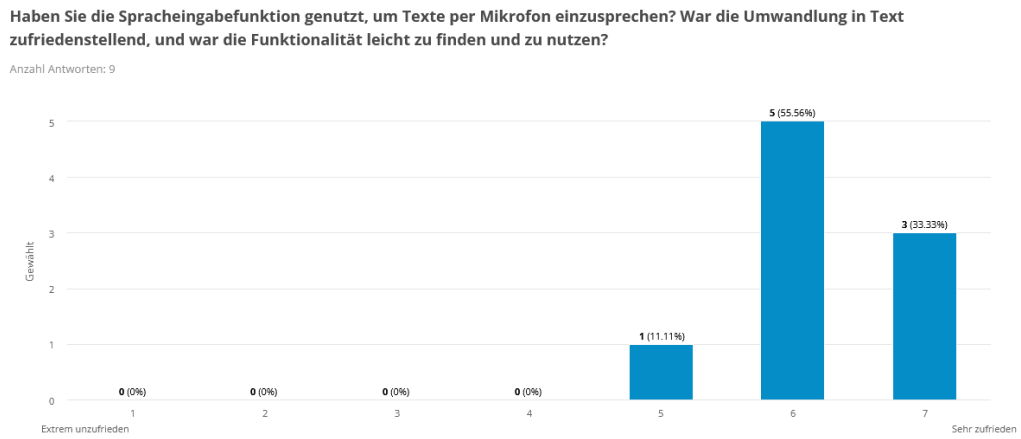


Abbildung A.11: Umfrage (4/6)

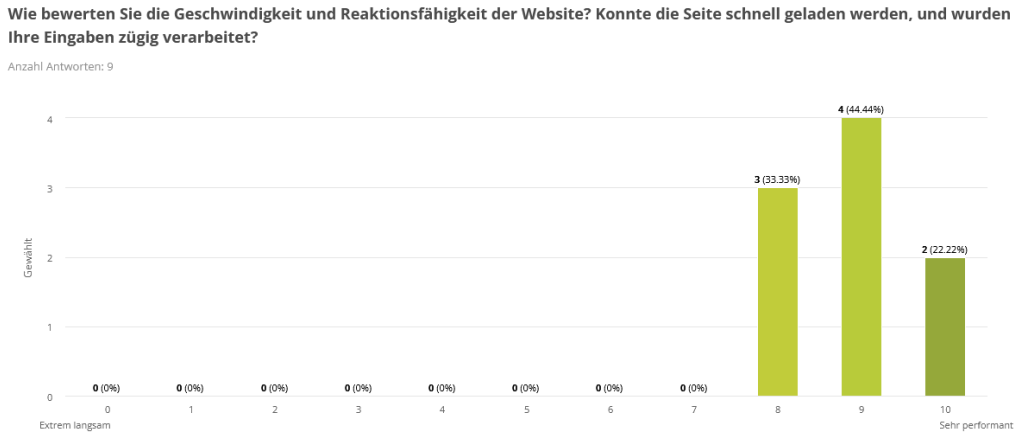
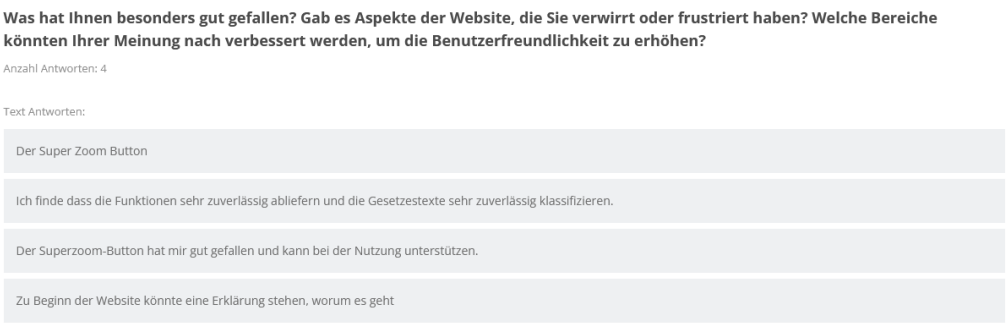


Abbildung A.12: Umfrage (5/6)



Was hat Ihnen besonders gut gefallen? Gab es Aspekte der Website, die Sie verwirrt oder frustriert haben? Welche Bereiche könnten Ihrer Meinung nach verbessert werden, um die Benutzerfreundlichkeit zu erhöhen?

Anzahl Antworten: 4

Text Antworten:

Der Super Zoom Button

Ich finde dass die Funktionen sehr zuverlässig abliefern und die Gesetzestexte sehr zuverlässig klassifizieren.

Der Superzoom-Button hat mir gut gefallen und kann bei der Nutzung unterstützen.

Zu Beginn der Website könnte eine Erklärung stehen, worum es geht

Abbildung A.13: Umfrage (6/6)

Literaturverzeichnis

Albanese, N. C. (2022). A guide to Text Classification(NLP) using SVM and Naive Bayes with Python. *Towards Data Science*. Verfügbar 26. Juli 2023 unter https://towardsdatascience.com/fine-tuning-bert-for-text-classification-54e7df642894

Bedi, G. (2018). A guide to Text Classification(NLP) using SVM and Naive Bayes with Python. Verfügbar 26. Juli 2023 unter https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34

Chakravarthy, S. (2021). Document summarization using latent semantic indexing. https://towardsdatascience.com/document-summarization-using-latent-semantic-indexing-b747ef2d2af6

Chiusano, F. (2022). Two minutes NLP — Learn the ROUGE metric by examples. https://medium.com/nlplanet/two-minutes-nlp-learn-the-rouge-metric-by-examples-f179cc285499

Gerlach, M., & Font-Clos, F. (2022). *Blog Authorship Corpus*. Verfügbar 25. Juli 2023 unter https://github.com/pgcorpus/gutenberg

Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*.

Institute for Disability Research, Policy Practice. (n. d.). *WAVE Browser Extensions*. Verfügbar 25. Juli 2023 unter https://wave.webaim.org/extension/

Jain, C. (2020). TextRank for text summarization. *OpenGenus IQ: Computing Expertise Legacy*. https://iq.opengenus.org/textrank-for-text-summarization/

Joshi, P. (2023). An introduction to text summarization using the TextRank Algorithm (with Python implementation). *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/

Linux Foundation. (n. d.). *DATASETS DATALOADERS*. Verfügbar 25. Juli 2023 unter https://pytorch.org/tutorials/beginner/basics/data_tutorial.html#preparing-your-data-for-training-with-dataloaders

Maurice-Demourioux, Y. (2021). *Barrierefreie Websites: Rechtliche Grundlagen und BITV 2.0*. Verfügbar 23. Juli 2023 unter https://www.webit.de/blog/2021/03/11/barrierefreie-websites-rechtliche-grundlagen-und-checkliste-fuer-redakteure

Microsoft. (n. d.). *Ändern des Farbkontrasts in Windows*. Verfügbar 25. Juli 2023 unter https://support.microsoft.com/de-de/windows/%C3%A4ndern-des-farbkontrasts-in-windows-fedc744c-90ac-69df-aed5-c8a90125e696#WindowsVersion=Windows_10

Mihalcea, R. (2004, 1. Juli). *TextRank: Bringing order into text*. https://aclanthology.org/W04-3252/

Mithbavkar, O., & Chauhan, A. (2021). Techniques for Extractive and Abstractive Text Summarization. Verfügbar 23. Mai 2023 unter https://medium.com/@devilchauhan0/techniques-for-extractive-and-abstractive-text-summarization-6ed44a5465f6

Ozsoy, M. G., Alpaslan, F. N., & Cicekli, I. (2011). Text summarization using latent semantic analysis. *Journal of Information Science*, *37*(4), 405–417. https://doi.org/10.1177/0165551511408848

Rouge-score. (2022, 22. Juli). https://pypi.org/project/rouge-score/

Shukla, A., Bhattacharya, P., Poddar, S., Mukherjee, R., Ghosh, K., Goyal, P., & Ghosh, S. (2022). *Legal Case Document Summarization: Extractive and Abstractive Methods and their Evaluation*. Verfügbar 25. Juli 2023 unter https://zenodo.org/record/7151679

Tatman, R. (2017). *Blog Authorship Corpus*. Verfügbar 25. Juli 2023 unter https://www.kaggle.com/datasets/rtatman/blog-authorship-corpus

University of Virginia. (2022). *PRESIDENTIAL SPEECHES: DOWNLOADABLE DATA*. Verfügbar 25. Juli 2023 unter https://millercenter.org/presidential-speeches-downloadable-data

Wolf, Platen, V., & McMillan. (2022). *cnn$_d$ailymail*. Verfügbar 25. Juli 2023 unter https://huggingface.co/datasets/cnn_dailymail