Natural Language Processing (NLP) mit PyTorch am Beispiel eines Hochschul-Chatbots

Nicolas Lang (M.Sc.)

Technische Hochschule Mittelhessen
Fachbereich Mathematik, Naturwissenschaften und Datenverarbeitung (MND)
Wilhelm-Leuschner-Str. 13, 61169 Friedberg
nicolas.lang@mnd.thm.de

Prof. Dr. Harald Ritz

Technische Hochschule Mittelhessen
Fachbereich Mathematik, Naturwissenschaften und Informatik (MNI)
Wiesenstr. 14, 35390 Gießen
harald.ritz@mni.thm.de

Prof. Dr. Frank Kammer

Technische Hochschule Mittelhessen
Fachbereich Mathematik, Naturwissenschaften und Informatik (MNI)
Wiesenstr. 14, 35390 Gießen
frank.kammer@mni.thm.de

Jonas Wölfer (M.Sc.)

Technische Hochschule Mittelhessen
Fachbereich Mathematik, Naturwissenschaften und Informatik (MNI)
Wiesenstr. 14, 35390 Gießen
jonas.woelfer@mni.thm.de

ABSTRACT

Der vorliegende Artikel beschreibt die Weiterentwicklung des KI-gestützten FAQ-Chatbots der Technischen Hochschule Mittelhessen (THM) zur Verbesserung der Antwortgenauigkeit durch die Entwicklung eines domänenspezifischen Transformers für das Extractive Question Answering (EQA).

Im Jahr 2021 wurde der Chatbot „Winfy“ im Rahmen eines Masterprojekts ins Leben gerufen. Dieses Projekt wurde vom Prüfungsausschussvorsitzenden B.Sc. und M.Sc. Wirtschaftsinformatik angestoßen, um den Studierenden eine Möglichkeit zu bieten, Antworten auf häufig gestellte Fragen (FAQ) rund um das Thema Prüfungsangelegenheiten in den beiden Wirtschaftsinformatik-Studiengängen zu erhalten.

Der Chatbot basiert auf einem Fragen-Antwort-Katalog, in dem eine einzelne Antwort mehreren Fragen zugeordnet ist. Wenn ein Studierender eine Frage stellt, liefert das System den Antwortblock, der mit der semantisch am besten passenden Frage verknüpft ist. Dies kann jedoch die Suche nach der exakt gewünschten Information erschweren. Um das Antwortverhalten zu verbessern, wurde ein Extractive Question Answering (EQA)-Modell entwickelt. Dieses sollte gezielt die Antwort aus dem Antwortblock extrahieren.

Hierfür wurde ein Transformer-Encoder mit ELECTRA-Architektur entwickelt und in einem zweistufigen Verfahren trainiert: Zunächst erfolgte ein Pre-Training auf domänenspezifischen Texten, gefolgt von einer Feinabstimmung auf einem speziell erstellten und annotierten EQA-Datensatz für den Hochschulbereich. Abschließend wurden verschiedene deutsche Sprachmodelle auf diesem und anderen etablierten EQA-Datensätzen verglichen. Das Modell „Winfy-ELECTRA-Base“ erzielte dabei die besten Ergebnisse und stellt eine signifikante Verbesserung der Antwortqualität dar.

SCHLÜSSELWÖRTER

Chatbot, Machine Learning (ML), Natural Language Processing (NLP), Transformers, Huggingface, Neuronale Netze, Frequently Asked Questions (FAQ), PyTorch, ELECTRA, Extractive Question Answering (EQA), Künstliche Intelligenz (KI)

VERWANDTE ARBEITEN

Verwandte Arbeiten verfolgen ebenfalls den Ansatz, domänenspezifisches Pre-Training einzusetzen, um die Leistungsfähigkeit der Modelle zu steigern, und zeigen dabei, dass dies oft bessere Ergebnisse erzielt als die Feinabstimmung allgemein vortrainierter Modelle. Zu bekannten Beispielen zählen „SciBERT“ oder „PubMedBERT“, die auf wissenschaftlichen Texten trainiert wurden, mit einem kontextspezifischen Vokabular arbeiten und somit deutlich besser abschneiden [1] [2].

AUSGANGSSITUATION

Um die Mitarbeitenden des Fachbereichssekretariats MND der Technischen Hochschule Mittelhessen (THM) zu entlasten, wurde 2021 das Projekt „Winfy“ vom Prüfungsausschussvorsitzenden gestartet. Dieser KI-basierte FAQ-Chatbot (https://feedback.mni.thm.de/winfy/) wurde entwickelt, um häufig gestellte Fragen zu Prüfungsangelegenheiten im Studiengang Wirtschaftsinformatik zu beantworten. Der Chatbot arbeitet mit einer Datenbank häufig gestellter Fragen, wobei mehrere

Fragen mit einer Antwort verknüpft sind. Durch eine Ähnlichkeitsberechnung, die vom Sentence-Transformer „German-Semantic“ [3] durchgeführt wird, wird die eingehende Nutzerfrage mit dem vorhandenen Fragenkatalog abgeglichen. Die Antwort auf die Frage mit dem höchsten Ähnlichkeitswert wird ausgegeben [4].

Abbildung 1: Ähnlichkeitsberechnung Sentence-Transformer

Das „German-Semantic“-Modell erzielt bereits sehr gute Ergebnisse. In den Antwortblöcken sind die Antworten auf mehrere Fragen zusammengefasst. Dies spart Zeit bei der Pflege der Fragen und kann den Studierenden zusätzliche Informationen im selben Kontext geben, erschwert allerdings die Antwortfindung, da die Länge der Antworten sehr umfangreich sein kann.

AUFGABENSTELLUNG UND ZIELSETZUNG

Das Ziel war es, die Antwortgenauigkeit des Chatbots zu verbessern. Dafür sollte ein eigenes neuronales Netz auf Basis der Transformer-Architektur für den Hochschuleinsatz gebaut werden. Als Einschränkung galt, dass das System keinen Text generieren durfte, um zu vermeiden, dass Falschinformationen verbreitet werden. Somit wurde der Ansatz des Extractive Question Answering verfolgt [17]. Anstelle eines bereits vorhandenen Modells sollte ein eigenes trainiert werden, um dieses auf den domänenspezifischen Kontext anzupassen. Dafür mussten entsprechende Daten für das Pre-Training gesammelt, gefiltert und aufbereitet werden.

Da kein EQA-Datensatz für die Hochschuldomäne existiert, um abschließend die Performance des Modells zu bewerten, musste selbst einer erstellt und annotiert werden. Als Basis wurde der bestehende Fragen-Antwort-Katalog des Chatbots genutzt.

Abschließend sollten die bereits vorhandenen Modelle mit dem selbst trainierten Modell verglichen werden – zum einen anhand anerkannter EQA-Benchmarks, aber auch auf dem domänenspezifischen Datensatz.

DATENSAMMLUNG

Für das Pre-Training musste eine große Menge an Daten gesammelt werden. Es wurde auf bekannte Quellen wie Wikipedia [5] und Wikivoyage [6] zurückgegriffen. Diese enthalten hochwertige Texte mit langen Sequenzen und eignen sich sehr gut für das Training von neuronalen Netzen.

Um an domänenspezifische Hochschultexte zu gelangen, wurden Modulhandbücher und Prüfungsordnungen gesammelt und in einem Datensatz namens „Academic Crawl“ zusammengefasst.

Da dies nur zu einer sehr geringen Datenmenge führte, wurde außerdem der GC4-Datensatz [7] verwendet. Dabei handelt es sich um den deutschen Auszug des CommonCrawl [8], der Texte von Internetseiten, Nachrichtenseiten sowie Foren und sozialen Medien enthält. Der GC4-Datensatz ist in drei Qualitäts-kategorien unterteilt: Head, Middle und Tail [7]. Es wurden nur die Artikel der Head-Kategorie verwendet, um den Qualitätsgrad hochzuhalten. Weiterhin wurden anschließend lediglich die Artikel verarbeitet, die eine Qualität von 0,99 oder höher aufwiesen. Dies ergab eine Datenmenge von 450 GB.

Da auch domänenfremde Texte enthalten waren, mussten diese gefiltert werden. Dies wurde aufgrund der enormen Datenmenge mithilfe einer Methode des Information Retrieval, dem TF-IDF (Term Frequency-Inverse Document Frequency), durchgeführt [9]. Dabei wurden exemplarische Hochschultexte genommen und mit den Artikeln des Datensatzes verglichen. Als Maßgabe sollten die Artikel eine Ähnlichkeit von 40% zu den Beispielen aufweisen. Nach der Filterung hatte der Datensatz eine Größe von 16 GB. Schlussendlich wurde das Vorab-Training mit folgenden Datensätzen durch-geführt:

Datensatz	**Größe**
Wikipedia 2024	6,2 GB
Wikivoyage	55 MB
Crawled Academic Dataset	6,1 MB
GC4 Corpus Filtered	16 GB
Total	**22,25 GB**

Tabelle 1: Gesammelte Pre-Training-Datensätze

Um die insgesamt 22,25 GB an Daten für das Pre-Training zu nutzen, mussten diese noch aufbereitet werden.

DATENAUFBEREITUNG

Die Daten wurden in drei Schritten aufbereitet: der Satztrennung, Normalisierung und Prüfung auf unerlaubte Zeichen. Transformer-Encoder arbeiten mit einer fixen Eingabelänge. Dieses Modell sollte mit einer Tokenlänge von 512 Tokens trainiert werden. Dafür war es wichtig, dass die Trainingsbeispiele diese Länge nicht überschreiten. Um dies zu ermöglichen, mussten die Eingabesequenzen der Datensätze zunächst in einzelne Sätze unterteilt werden, damit diese später auf die gewünschte Länge konkateniert werden können.

Die Satztrennung wurde mithilfe des „SoMaJo“-Tokenizers durchgeführt. Dieser eignet sich besonders gut für die Verarbeitung deutscher Webtexte [10].

Um das Rauschen in den Daten zu verringern und das Vokabular des Tokenizers so effizient wie möglich zu gestalten, wurden die Sätze normalisiert. Dafür wurden die Skripte des CCNet-Projekts [11] verwendet. Dabei wurden Zeichen, die die gleiche Bedeutung haben, auf eine einheitliche Form gebracht, z. B. verschiedene

Formen von Anführungszeichen («, “, '). Weiterhin wurden Unicode-Fehler korrigiert und Emojis entfernt, die häufig in Webtexten auftreten. Somit konnte die Datenqualität angehoben werden.

Weiterhin wurden Zeichen entfernt, die im späteren Einsatz des Sprachmodells nicht vorgesehen waren und somit nicht zum Lernerfolg des Modells beitrugen. Vorgesehene Zeichen waren ASCII-Zeichen sowie ausgewählte Symbole und Umlaute der deutschen Sprache. Durch das Normalisieren und das Entfernen unerlaubter Zeichen sollte sichergestellt werden, dass keine Vokabularplätze des Tokenizers für Tokens verschwendet werden, die nur selten in den Datensätzen vorkommen bzw. im späteren Einsatz überhaupt nicht mehr benötigt werden.

Anschließend wurde der Tokenizer mithilfe der vorliegenden Texte trainiert, damit dieser optimal auf die Trainingsdaten abgestimmt ist.

```
1.  "vocab": {
2.    "[UNK]": 0,
3.    "[PAD]": 1,
4.    "[CLS]": 2,
5.    "[SEP]": 3,
6.    "[MASK]": 4,
7.    "!": 5,
8.    "\"": 6,
9.    "'": 7,
10.   "(": 8,
11.   ")": 9,
12.   ",": 10,
13.   "-": 11,
14.   ".": 12,
15.   "0": 13,
16.   "1": 14,
17.   "2": 15,
18.   ...
19.   "fachober": 29996,
20.   "tschern": 29997,
21.   "##dekan": 29998,
22.   "schleich": 29999
23. }
```

Listing 1: Tokenizer Vokabularauszug

Somit konnten die einzelnen Sätze zur gewünschten Tokenlänge von 512 Tokens zusammengefügt werden. Dabei wurden nur Texte zusammengefügt, die aus dem gleichen Artikel stammten.

EQA-DATENSÄTZE

Um den Modellen das EQA beibringen zu können und deren Performance darauf zu bewerten, mussten Datensätze gesammelt werden. Dabei wurde die maschinelle Übersetzung des SQuAD (Stanford Question Answering Dataset) (hier als SQuAD-De aufgeführt) verwendet. Dieser wurde als Anhang im MLQA-Datensatz [12] von Facebook veröffentlicht. Der Vorteil ist, dass dieser mit seinen 89.996 Einträgen sehr umfangreich ist. Durch die maschinelle Übersetzung leidet allerdings die Datenqualität, da diese nicht die Nuancen der deutschen Sprache enthält [13].

Als weiterer Datensatz wurde der GermanQuAD [13] (German Question Answering Dataset) verwendet. Dieser beinhaltet deutlich weniger Einträge (13.722), weist allerdings eine sehr hohe Qualität auf. Bei den annotierten Texten handelt es sich um originale deutsche Artikel, die von geschultem Personal für diese Aufgabe nach definierten Qualitätskriterien gelabelt wurden [13].

Um die Performance der Modelle für das EQA auf der Hochschuldomäne bewerten zu können, musste noch ein Datensatz erstellt werden, da ein solcher bis zu diesem Zeitpunkt nicht existierte. Als Datenbasis wurde der Fragenkatalog des Chatbots „Winfy“ genutzt.

Dabei wurden die Fragen zuerst bereinigt. Fragen, die sich zu ähnlich waren, wurden automatisiert entfernt (semantische Ähnlichkeit von 85%). Die verbliebenen 2.863 Fragen wurden mithilfe der Qualitätskriterien des GermanQuAD annotiert. Für das Annotieren wurde die Software Label Studio [14] verwendet.

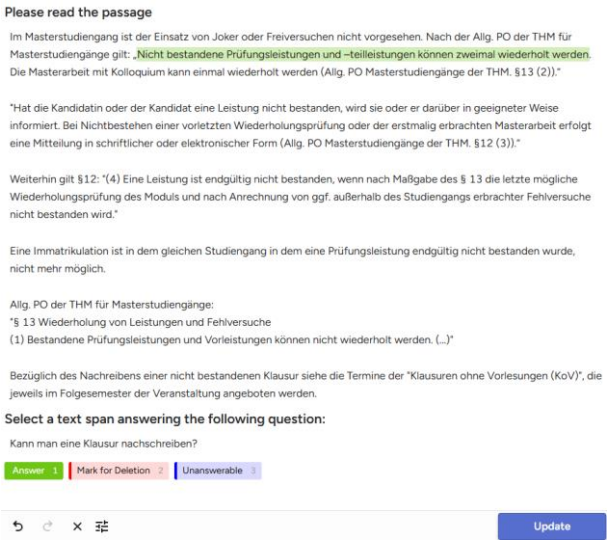


Abbildung 2: Label Studio Annotationsprozess

Dabei wurde die Antwort zur gestellten Frage im Kontext markiert. Fragen, die nicht den Qualitätskriterien entsprachen oder sich nicht beantworten ließen, wurden entsprechend markiert und entfernt.

```
1.  {
2.   "paragraphs": [
3.      {
4.        "context": "Bei Abholung der Chipkarte im InfoCenter wird Ihnen...",
5.        "document_id": 33135,
6.        "qas": [
7.         {
8.          "question": "Wo finde ich die Aktualisierungstermials …"
9.          "id": 33135,
10.         "answers": [
11.                {
12.           "answer_id": 11195,
13.           "text": "im InfoCenter und in den Gebäuden Gebäuden A2, A4 und A7",
14.           "answer_start": 248
15.           }
16.           ],
17.         "is_impossible": false
18.     }
19.    ]
20.   }
21.  ]
22. }
```

Listing 2: WinfyQuAD

Anschließend wurden die Daten mithilfe eines Skripts in eine für das EQA trainierbare Form gebracht (siehe Abbildung 2 oben). Es verblieben 1.746 Fragen. Der entstandene Datensatz wurde aufgrund der Datenbasis „WinfyQuAD“ (Winfy Question Answering Dataset) genannt.

MODELLARCHITEKTUR

Als Modellarchitektur wurde der ELECTRA-Ansatz (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) gewählt. Dabei werden zwei Modelle gleichzeitig trainiert: ein Generator- und ein Diskriminator-Modell [15].

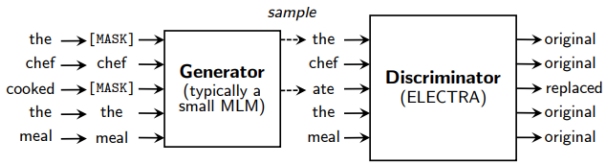


Abbildung 3: ELECTRA-Training (Quelle: [15])

Die beiden Modelle werden mithilfe von Self-Supervised-Learning trainiert. Zuerst werden 15% der Tokens einer Eingabesequenz maskiert. Die maskierte Sequenz wird an den Generator übergeben. Dieser hat die Aufgabe des Masked Language Modeling (MLM) und muss erraten, welche Tokens ursprünglich maskiert wurden. Die Vorhersage des Generators durchläuft einen „Sample"-Prozess, sodass nicht automatisch die wahrscheinlichste Vorhersage des Generators genommen wird. Der Diskriminator erhält die „beschädigte" Eingabesequenz und muss erraten, welche der Tokens echt oder vertauscht wurden. Dadurch muss der Diskriminator bei 100 % der Tokens eine Vorhersage treffen, während dies beim MLM nur bei 15 % geschieht. Dies führt dazu, dass ELECTRA ein dateneffizienterer Trainingsansatz ist als z. B. BERT [15].

Weiterhin besitzt der Generator nur ca. ein Viertel bis ein Drittel der Parametergröße des Diskriminators, damit dieser nicht zu gute Vorhersagen beim MLM ausgibt und das Training aus dem Gleichgewicht gerät [15]. Als Modellgröße wurden folgende Hyperparameter gewählt:

Hyperparameter	Generator	Discriminator
Hidden Layers	12	12
Sequence length	512	512
Hidden Size	256	768
FFN inner hidden size	1024	3072
Attention heads	4	12
Attention head size	64	64
Embedding size	768	768
Learning rate decay	Linear	Linear
Warmup steps	10K	10K
Learning rate	2e-4	2e-4
Attention dropout	0.1	0.1
Dropout	0.1	0.1
Weight decay	0.01	0.01
Batch size	256	256
Train steps	766K	766K
Vocab size	30K	30K
Total parameter	33M	110M

Tabelle 2: Hyperparameter Winfy-ELECTRA-Base

Dabei wurde sich am originalen ELECTRA orientiert.

Um die Architektur umzusetzen, wurde PyTorch in Kombination mit der Transformers-Bibliothek von Huggingface [16] verwendet. Diese Bibliothek stellt bereits die Architektur für den Generator und den Diskriminator bereit. Die Interaktion zwischen diesen beiden Modellen musste allerdings selbst entwickelt werden.

PRE-TRAINING

Das Pre-Training wurde auf einer NVIDIA GeForce RTX 4090 durchgeführt. Die 766.000 Trainingsschritte dauerten insgesamt 560 Stunden.

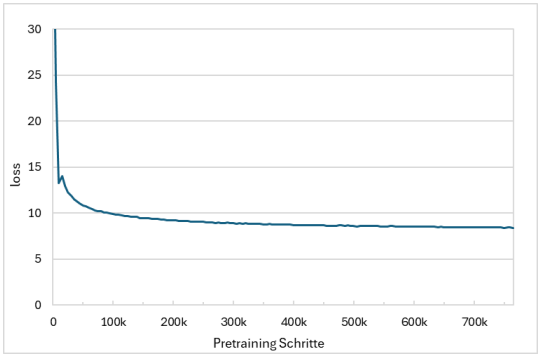


Abbildung 4: Verlustkurve Winfy-ELECTRA-Base

In der obigen Grafik ist der vereinte Validationsverlust der Modelle zu sehen. Dabei wurde der Verlust des Diskriminators mit dem Faktor 50 gewichtet, damit dieser bei der Anpassung der Gewichte bevorzugt wird. Es zeigte sich, dass der Verlust bis zum Ende stetig sank und kein Anzeichen einer Überanpassung auf die Trainingsdaten (Overfitting) zeigte. Somit könnte das Pre-Training auch weiter fortgeführt werden.

Das resultierende Modell wurde nach dem Chatbot benannt, in welchem es verwendet werden soll: „Winfy-ELECTRA-Base". Nun musste dem Modell noch die Aufgabe des EQA angelernt werden.

EXTRACTIVE QUESTION ANSWERING

EQA gehört zu den wichtigsten Aufgaben des Natural Language Processing (NLP). Dabei wird zwischen „Single-Span" und „Multi-Span" differenziert. Beim „Single-Span" wird versucht, die Antwort aus genau einer Textspanne zu extrahieren. Beim „Multi-Span" wird die Antwort aus mehreren Stellen im Text extrahiert.[17]. In diesem Projekt wurde aufgrund mangelnder Verfügbarkeit entsprechender Datensätze nur das „Single-Span"-EQA behandelt.

Um einem Modell das EQA beizubringen, muss die Architektur nicht geändert werden. Es muss lediglich die Ausgangsschicht für den neuen Einsatzbereich angepasst werden. Für die Aufgabe des EQA werden zwei lineare Schichten benötigt. Die erste Schicht soll den Anfang der Antwort im Kontext vorhersagen, die zweite Schicht soll das Ende der Antwort bestimmen [18].

Die Eingabeschicht muss nicht geändert werden, da der Tokenizer damit umgehen kann. Dabei wird das Klassifikationstoken (CLS) verwendet, um den Beginn der Eingabesequenz zu kennzeichnen, und das Trenn-Token (SEP), um die Frage vom Kontext zu trennen

sowie das Ende der Sequenz zu markieren. Das Modell muss nun eine Klassifikationsaufgabe lösen, um die korrekte Antwortspanne vorherzusagen [18].

Um die Laufzeiteffizienz der Modelle bewerten zu können, wird auf zwei Metriken zurückgegriffen: den Exact Match (EM) und den F1-Wert. EM misst, ob das Modell exakt die richtige Antwortspanne vorhersagt. F1 gibt an, wie viele Prozent der vorhergesagten Tokens mit der tatsächlich korrekten Antwortspanne übereinstimmen [19].

FEINABSTIMMUNG

Für den Vergleich wurden vorhandene deutsche Transformer-Encoder Modelle gesucht, um diese mit dem Winfy-ELECTRA-Base zu vergleichen. Dabei standen die beiden Modelle deepset/gbert-base und deepset/gelectra-base zur Verfügung [20].

Für die Feinabstimmung wurden die Modelle, wie auch im Artikel „Improving Non-English Question Answering and Passage Retrieval“ [13] beschrieben, zuerst auf dem SQuAD-De „aufgewärmt“, da dies die Ergebnisse auf den folgenden Datensätzen verbesserte. Alle Modelle wurden einheitlich auf den drei Datensätzen trainiert. Als Hyperparameter wurden folgende Werte genutzt:

Hyperparameter	SQuAD-De	German QuAD	Winfy QuAD
Sequence length	384	384	384
Stride	128	128	128
Learning rate decay	Linear	Linear	Linear
Warmup steps	500	500	500
Learn rate	3e-5	3e-5	3e-5
Total Size (num. of entries)	89.996	13.722	1.746
Epochs	2	2	6
Batch size	24	24	24

Tabelle 3: Hyperparameter EQA-Feinabstimmung

Die Feinabstimmung dauerte aufgrund des deutlich geringeren Trainingsumfangs im Vergleich zum Vorab-Training für jedes Modell nur 30 Minuten.

Da die Performance des Modells in der Pre-Training-Aufgabe nicht automatisch Rückschlüsse auf die Leistung im Downstream-Task zulässt, wurde während des Vorab-Trainings alle 50.000 Schritte ein Sicherungspunkt des Modellzustands erstellt, um später jeden einzelnen Sicherungspunkt betrachten zu können. Die Feinabstimmung wurde für jeden dieser Sicherungspunkte durchgeführt, um den besten Zustand des Modells für die EQA-Aufgabe zu finden.

Abbildung 5: Winfy-ELECTRA-Base Performance aller Sicherungspunkte

In der Grafik sind der EM- und der F1-Wert auf dem WinfyQuAD für jeden Sicherungspunkt zu sehen. Es zeigte sich, dass das Modell beim Trainingsschritt 750.000 den höchsten EM-Wert erreicht.

AUSWERTUNG UND VERGLEICH

Die Feinabstimmung wurde für alle Modelle durchgeführt, und die Ergebnisse wurden dokumentiert.

	SQuAD-De-Test		GermanQuAD-Test		WinfyQuAD-Test	
Modell	EM	F1	EM	F1	EM	F1
deepset/gbert-base	50,35	65,65	58,21	75,61	45,80	64,79
deepset/gelectra-base	**53,23**	68,51	**62,93**	79,43	55,34	68,75
Winfy-ELECTRA-Base	52,91	**68,63**	**62,93**	**80,70**	**60,68**	**73,13**

Tabelle 4: EQA-Vergleich der Modelle (Angaben in %)

Es zeigte sich, dass Winfy-ELECTRA-Base sowohl auf dem GermanQuAD als auch auf dem WinfyQuAD am besten abschnitt. Auf dem GermanQuAD waren die Unterschiede nur marginal, während auf dem WinfyQuAD eine deutlichere Verbesserung zu verzeichnen war. Das deepset/gbert-base performt auf dem SQuAD-De und dem GermanQuAD ähnlich gut wie die anderen Modelle, ist allerdings deutlich schwächer auf dem WinfyQuAD. Dies kann darauf zurückzuführen sein, dass es während des Vorab-Trainings mit den wenigsten Daten trainiert wurde und somit vermutlich nicht mit Texten aus dem Hochschulkontext in Berührung gekommen ist da die verwendeten Datensätze nur „Wikipedia“ „OpenLegalData“ und „News“ waren [22]. Das deepset/gelectra-base dagegen wurde mit 163,4 GB an Daten trainiert, von denen ein Großteil (89 %) aus dem OSCAR-Datensatz bestand [20]. Somit liegt es nahe, dass unter diesen Daten auch Hochschultexte enthalten waren. Dabei handelt es sich aber lediglich um eine Vermutung, da es sich nicht überprüfen lässt.

INTEGRATION

Abschließend wurde das Winfy-ELECTRA-Base, welches auf das EQA feinabgestimmt wurde, in den Winfy-Chatbot eingebaut. Dadurch, dass das Training mithilfe der Transformers-Bibliothek von Huggingface durchgeführt wurde, lässt sich das Modell sehr einfach implementieren.

```python
from transformers import pipeline, AutoTokenizer, AutoModelForQuestionAnswering

model_path = "winfy-electra-base"
tokenizer = AutoTokenizer.from_pretrained(model_path)
model = AutoModelForQuestionAnswering.from_pretrained(model_path)

qa_pipeline = pipeline(
    "question-answering",
    model=model,
    tokenizer=tokenizer
)

context = "Spätestens am letzten Tag der Bearbeitungsfrist ist eine digitale pdf-Version der Abschlussarbeit an das MND-Dekanat zur Fristwahrung zu mailen..."
question = "Wann ist spätestens die Abschlussarbeit abzugeben?"

result = qa_pipeline(context=context, question=question, max_length=384)
```

Listing 3: Winfy-ELECTRA-Base EQA mit Huggingface

Als Rückgabewert erhält man die Anfangs- und Endposition der Antwort innerhalb des Kontexts, die extrahierte Antwortspanne als Text sowie einen Score, wie sicher das Modell sich mit der Vorhersage ist.

Es wurde beibehalten, dass zuerst der Antwortblock mithilfe des „German-Semantic“-Modells durch eine semantische Ähnlichkeitsüberprüfung vorausgewählt wird. Dieser Antwortblock wird nun zusammen mit der Frage an das EQA-Modell übergeben. Dies wird aus Performancegründen getan. Übergibt man alle Antwortblöcke gemeinsam mit der gestellten Frage an das EQA-Modell, so beträgt die Verarbeitungszeit auf einer CPU vier Minuten, während der eben beschriebene Ansatz nur zwei Sekunden dauert.

Das EQA-Modell gibt die exakte Position der Antwort zurück. Diese Information wird genutzt, um den Absatz zu bestimmen, der die Antwort enthält. Dieser wird an den Endnutzer ausgegeben (siehe Anhang).

FAZIT UND AUSBLICK

Es zeigte sich, dass das domänenspezifische Pre-Training seinen gewünschten Zweck erfüllen konnte. Das Winfy-ELECTRA-Base erzielte die besten Werte auf dem EQA-Hochschuldatensatz. Weiterhin konnte durch die Integration des neuen EQA-Modells die Antwortgenauigkeit des Chatbots verbessert und somit das Ziel der Arbeit erreicht werden.

Es wäre zudem interessant, das Modell mit mehr Parametern zu trainieren, um weitere mögliche Verbesserungen zu untersuchen. Auch könnte ein fortschrittlicherer Architekturansatz, wie zum Beispiel DeBERTaV3 [21], verfolgt werden.

Aufgrund der für das Training verwendeten EQA-Datensätze versucht das Modell, immer eine Antwort zu finden. Lässt sich die Frage nicht anhand des Kontexts beantworten, würde das Modell dennoch eine Textspanne ausgeben.

Dieses Problem wurde im SQuAD v.2.0 angegangen. Dieser ist allerdings zum Zeitpunkt der Projektdurchführung nur in englischer Sprache verfügbar. Sobald er auch in deutscher Sprache zugänglich ist, wäre es interessant, das Modell darauf zu trainieren.

LITERATUR

[1] Beltagy, Iz; Lo, Kyle; Cohan, Arman: SciBERT: A Pretrained Language Model for Scientific Text, o.O., 2019, DOI: https://doi.org/10.48550/arXiv.1903.10676

[2] Gu, Yu; Tinn, Robert; Cheng, Hao; Lucas, Michael; Usuyama, Naoto; Liu, Xiaodong; Naumann, Tristan; Gao, Jianfeng; Poon, Hoifung: Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing, in: ACM Transactions on Computing for Healthcare vol. 3 no. 1, 2021, S. 1-23, DOI: https://doi.org/10.1145/3458754

[3] Tomar, Sahaj: Sahajtomar/German-semantic, huggingface, o.O., o.J., online im Internet: URL: https://huggingface.co/Sahajtomar/German-semantic [Stand 19.12.2024]

[4] Ritz, Harald; Tansel, Dogus: Entwicklung eines KI-basierten FAQ-Chatbots für die Hochschule im Bereich Prüfungsangelegenheiten, in: Anwendungen und Konzepte in der Wirtschaftsinformatik (AKWI) Nr. 17 (2023), S.81-92, DOI: https://doi.org/10.26034/lu.akwi.2023.n17

[5] Wikipedia, o.O., o.J., online im Internet: URL: https://www.wikipedia.org/ [Stand 02.01.2025]

[6] Wikivoyage, o.O., o.J., online im Internet: URL: https://www.wikivoyage.org/ [Stand 02.01.2025]

[7] May, Philip; Reißel, Philipp: GC4 Corpus - The German colossal, cleaned Common Crawl corpus, o.O., 2021, online im Internet: URL: https://german-nlp-group.github.io/projects/gc4-corpus.html [Stand 25.12.2024]

[8] CommonCrawl - Common Crawl maintains a free, open repository of web crawl data that can be used by anyone, o.O., o.J., online im Internet: URL: https://commoncrawl.org/ [Stand 02.01.2025]

[9] Hiemstra, Djoerd: A probabilistic justification for using tf×idf term weighting in information retrieval, in: Int J Digit Libr 3, 2000, S. 131-139, DOI: https://doi.org/10.1007/s007999900025

[10] Proisl, Thomas; Uhrig, Peter; Cook, Paul; Evert, Stefan; Schäfer, Roland; Stemle, Egon (Hrsg.): SoMaJo: State-of-the-art tokenization for German web and social media texts, in: Proceedings of the 10th Web as Corpus Workshop, Berlin, 2016, S. 57-62, DOI: https://doi.org/10.18653/v1/W16-2607

[11] Wenzek, Guillaume; Lachaux, Marie-Anne; Conneau, Alexis; Chaudhary, Vishrav; Guzmán, Francisco; Joulin, Armand; Grave,

Edouard: CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data, o.O., 2019, DOI: https://doi.org/10.48550/arXiv.1911.00359

[12] Lewis, Patrick; Oguz, Barlas; Rinott, Ruty; Riedel, Sebastian; Schwenk, Holger; Jurafsky, Dan; Chai, Joyce; Schluter, Natalie; Tetreault, Joel (Hrsg.): MLQA: Evaluating Cross-lingual Extractive Question Answering, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Stroudsburg, PA, USA, 2020, S. 7315–7330, DOI: https://doi.org/10.18653/v1/2020.acl-main.653

[13] Möller, Timo; Risch, Julian; Pietsch, Malte: GermanQuAD and GermanDPR: Improving Non-English Question Answering and Passage Retrieval, in: Proceedings of the 3rd Workshop on Machine Reading for Question Answering, Punta Cana, Dominikanische Republik, 2021, S. 42-50, DOI: https://doi.org/10.18653/v1/2021.mrqa-1.4

[14] Tkachenko, Maxim; Malyuk, Mikhail; Holmanyuk, Andrey; Liubimov, Nikolai: Label Studio: Data labeling software, o.O., 2020, online im Internet: URL: https://github.com/HumanSignal/label-studio [Stand 20.12.2024]

[15] Clark, Kevin; Luong, Minh-Thang; Le, V. Quoc; Manning, Christopher D.: ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators, o.O., 2020, DOI: https://doi.org/10.48550/arXiv.2003.10555

[16] Wolf, Thomas; Debut, Lysandre; Sanh, Victor; Chaumond, Julien; Delangue, Clement; Moi, Anthony; Cistac, Pierric; Rault, Tim; Rémi, Louf; Funtowicz, Morgan; Davison, Joe; Shleifer, Sam; von Platen, Patrick; Ma, Clara; Jernite, Yacine; Plu, Julien; Xu, Canwen; Le Scao, Teven; Gugger, Sylvain; Drame, Mariama; Lhoest, Quentin; Rush, Alexander M.: HuggingFace's Transformers: State-of-the-art Natural Language Processing, o.O., 2020, DOI: https://doi.org/10.48550/arXiv.1910.03771

[17] Wang, Luqi; Zheng, Kaiwen; Qian, Liyin; Li, Sheng: A Survey of Extractive Question Answering, in: 2022 International Conference on High Performance Big Data and Intelligent Systems (HDIS), Tianjin China, 2022, S. 147-153, DOI: https://doi.org/10.1109/HDIS56859.2022.9991478

[18] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: Proceedings of NAACL-HLT 2019, Minneapolis, Minnesota, 2019, S. 4171-4186, DOI: https://doi.org/10.18653/v1/N19-1423

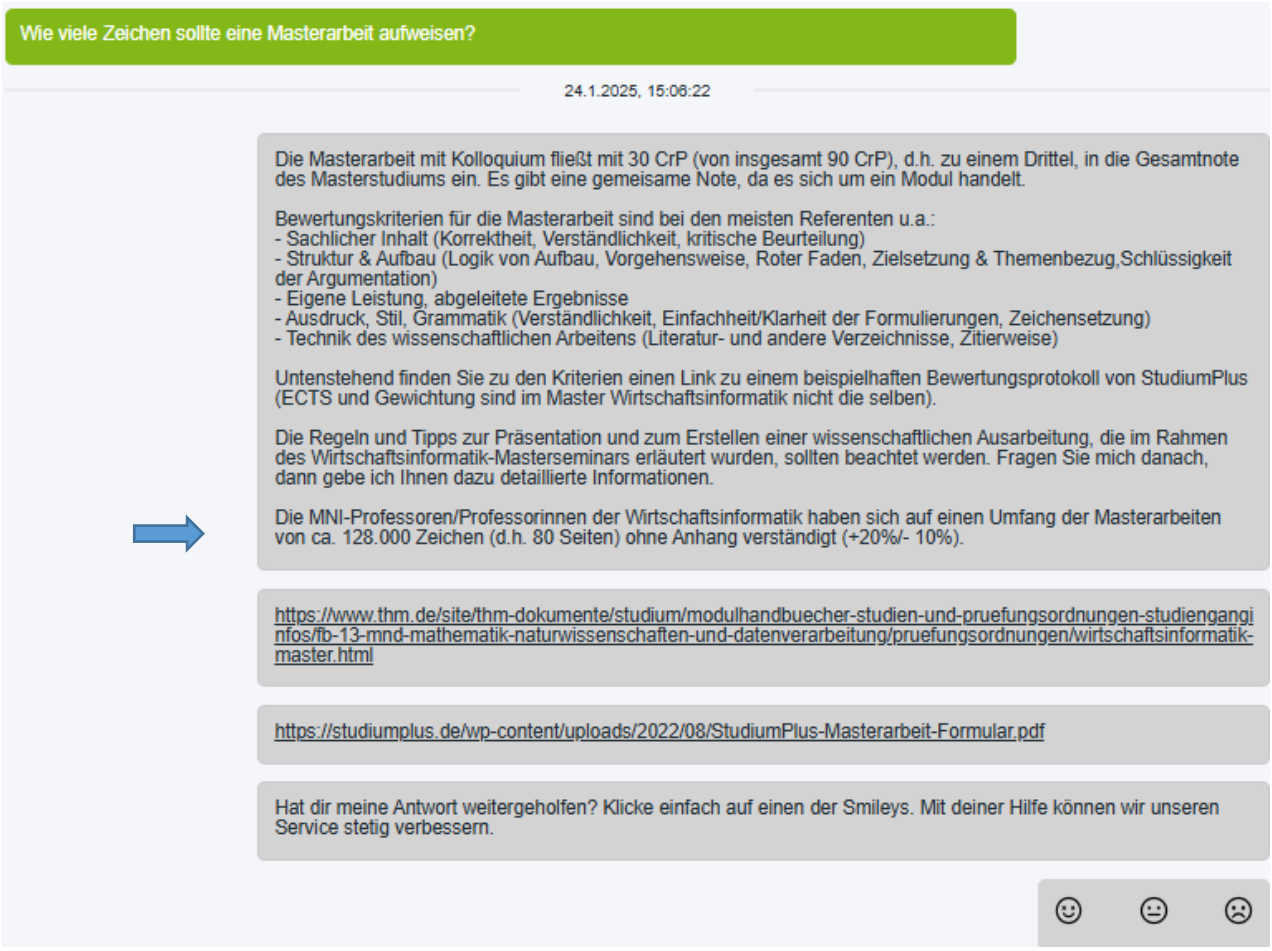
[19] Rajpurkar, Pranav; Zhang, Jian; Lopyrev, Konstantin; Liang, Percy: SQuAD: 100,000+ Questions for Machine Comprehension of Text, o.O., 2016, DOI: https://doi.org/10.48550/arXiv.1606.05250

[20] Chan, Branden; Schweter, Stefan; Möller, Timo; Scott, Donia; Bel, Nuria; Zong, Chengqing (Hrsg.): German's Next Language Model, in: Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spanien, 2020, S. 6788–6796, DOI: https://doi.org/10.18653/v1/2020.coling-main.598

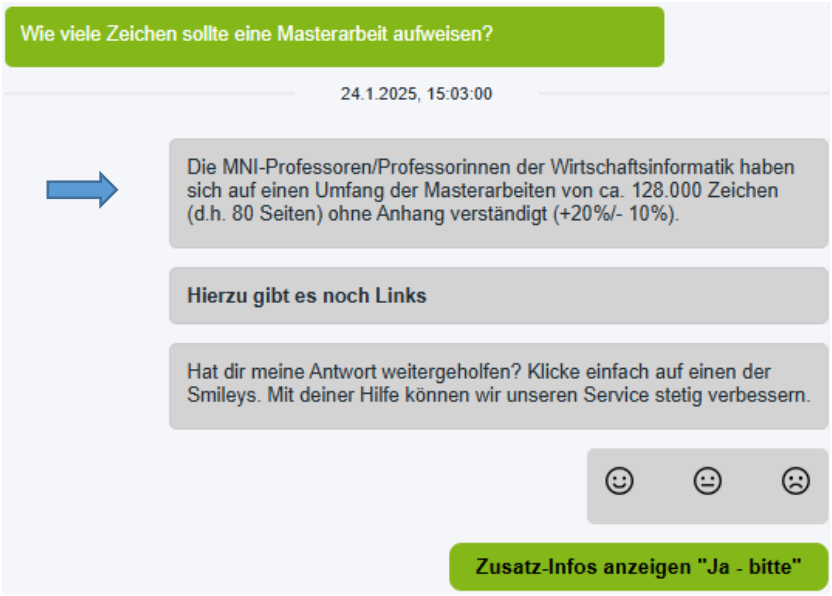
[21] He, Pengcheng; Gao, Jianfeng; Chen, Weizhu: DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing, o.O., 2023, DOI: https://doi.org/10.48550/arXiv.2111.09543

[22] Chan, Branden; Möller, Timo; Pietsch, Malte; Soni, Tanay: German BERT, o.O., 2019, online im Internet: URL: https://huggingface.co/google-bert/bert-base-german-cased [Stand: 30.12.2024]

ANHANG



Anhang 1: Antwortumfang vorher



Anhang 2: Antwortumfang nachher