University of Stavanger

Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: Computer Science Data Science	Spring semester, 20.22. Open / ~~Restricted access~~
Writer: Ole Petter Nilsen	..............Ole Petter Nilsen.............. (Writer's signature)
Faculty supervisor: Vinay Jayarama Setty External supervisor(s):	
Thesis title: Conversational AI for Serving Fact-Checks	
Credits (ECTS): 30	
Key words: Conversational AI, Fact-Checks	Pages: ....79......... + enclosure: Code included in PDF Stavanger, June 15 2022. Date/year

University of Stavanger

Faculty of Science and Technology
Department of Electrical Engineering and Computer Science

Conversational AI for Serving Fact-Checks

Master's Thesis in Computer Science
by

Ole Petter Nilsen

Internal Supervisors

Vinay Jayarama Setty

June 15, 2022

Abstract

The purpose of this thesis was to create a conversational AI for serving fact-checks, using a collection of already existing fact-checking articles. The conversational AI uses a hybrid system, combining both a question answering agent, chitchat agent, and multiple non-AI based skills to perform the task.

The program created consists of a user interface, broker, and seven different skills. For the implementation multiple existing pre-trained deep learning models were used, where many are based on the Transformer architecture. Already fine-tuned versions of these models were used. The conversational AI can present fact-checking articles in multiple ways, fact-check a claim presented, and has some multi-turn capabilities.

The result is a functional conversational AI which is capable of serving fact-checks from a collection of fact-checking articles. Although the conversational AI is functional, there are several issues that should be addressed, and further work to be done.

Acknowledgements

I would like to thank my supervisor Vinay Setty for his guidance and many suggestions regarding the project. His interest in the project helped with keeping motivation throughout the semester. Additionally, I would thank Factiverse for hosting the article collection, and lending their AI. I would also like to thank Mediafutures for showing interest in the project, and hosting the AI. Lastly, I would like to thank everyone who provided feedback on the AI.

Contents

Abbreviations

AI	**A**rtificial **I**ntelligence
AJAX	**A**synchronous **J**avascript **A**nd **X**ML
API	**A**pplication **P**rogramming **I**nterface
BM25	**B**est **M**atching **25**
CDN	**C**ontent **D**elivery **N**etwork
CSS	**C**ascade **S**tyle **S**heets
DBMS	**D**ata**B**ase **M**anagement **S**ystem
HTML	**H**yper**T**ext **M**arkup **L**anguage
IR	**I**nformation **R**etrieval
JS	**J**ava**S**cript
JSON	**J**ava**S**cript **O**bject **N**otation
MRC	**M**achine **R**eading **C**omprehension
NER	**N**amed **E**ntity **R**ecognition
NLP	**N**atural **L**anguage **P**rocessing
NLU	**N**atural **L**anguage **U**nderstanding
REST	**RE**presentational **S**tate **T**ransfer
URL	**U**niform **R**esource **L**ocator
SQL	**S**tructured **Q**uery **L**anguage

Chapter 1

Introduction

In today's society the rate at which information spreads has rapidly increased. Firstly with the development of the internet which made information easily accessible for anyone. Secondly with social media where information can be spread even faster. There is a catch though, anyone can author this information. This causes a problem since the information provided might be false. That be for financial gain, political gain, just for fun, and many other reasons. To combat this spread of false information, multiple fact-checking websites have been created, such as Politifact and Snopes. They check factual statements with credible sources and provide a verdict along with the factual information. Although these fact-checking sites are rising in popularity, false information are still being spread. The objective of this thesis is to provide a new way of accessing the fact-checks provided by these websites, namely a conversational AI.

1.1 Motivation

The motivation for this thesis is to improve the accessibility of factual information. The way factual checks are presented today provides a barrier for some people. To check a statement, one needs to visit the website focused on that specific type of topic and search for the factual claim. Having a fact-checking as easily accessible as telling your phones virtual assistant to call your friend, could help breaking down this barrier. An example of the use of false information for political gain is the Cambridge Analytica scandal. There false information was used to skew voters towards voting for a certain party, as is told by a former employee in an interview with BBC [1].

It is also a technically interesting problem. As many components are involved for the creation of a conversational AI, where multiple parts can best be solved with deep learning.

1.2 Problem Definition

This thesis focuses on presenting already existing fact-checking articles through a conversational AI. The problem can be broken into the following main task:

- Create a conversational AI that can serve fact-checks.

This objective can be broken into the following subtasks:

- The conversational AI should be able to answer the question based on the information present in the collection.
- The conversational AI should be able to serve the fact-checks in multiple ways.
- The conversational AI should have an intuitive use interface.
- The conversational AI should be able to handle multi-turn questions.
- The conversational AI should be human like in the conversation.

The conversational AI should be able to present fact-checks of interest to the end user, and the conversational capabilities of the AI should extend beyond a single question and response.

1.3 Use Cases/Examples

The functioning of the program is best shown through a dialogue between the AI and the user, as is shown in Figure 1.1.

Turn	Speaker	Message
1	User	Hi
	Bot	Hello, how can I help?
2	User	is the earth flat?
	Bot	No, the earth is not flat.
3	User	Okey, what shape is it then?
	Bot	The earth is round.
4	User	Thank you.
	Bot	You're welcome.

Table 1.1: Example conversation between the AI and end user.

A common use case for the conversational AI would be: A person sees a post on social media that seems questionable. The person can then ask the conversational AI about this claim and find out if the claim is correct.

1.4 Challenges

There are multiple challenges present for the task at hand. Some can be solved using already existing solutions, While some are new, and therefore need a new solution. Although many coversational AIs exist, none exist that are based on using a collection of available fact-checking articles. Therefore creating such a Conversational AI is a challenge in of itself.

1.5 Contributions

The contributions of this thesis is the creation of a conversational AI, which can serve fact-checks based on a collection of preexisting fact-checking articles.

1.6 Outline

The rest of the thesis is structured accordingly:

Chapter 2 introduces the concepts and models used in the creation of the conversational AI.

Chapter 3 presents the architecture and design decisions made, to create the conversational AI.

Chapter 4 presents the implementation of the architecture established in the previous chapter.

Chapter 5 presents the results from the feedback collected.

Chapter 6 discusses the results and the known limitations of the implemented conversational AI.

Chapter 7 presents a conclusion and further work suggestions.

Chapter 2

Related Work

2.1 Conversational AI

Conversational AI is an AI able to converse with a human. This could be day-to-day conversations, answering questions , assisting with some task or anything in between. There have been a large improvement in this field recently, due to the release of large datasets and progress in deep learning for multiple IR and NLP tasks. The conversations and tasks a conversational AI might need to handle can differ. Gao et al. [2] proposed to categorize conversational systems into three categories: Question answering agents, task-oriented dialogue agents, and chatbots. Task-oriented dialogue agents will not be mentioned, as they are not relevant for this project.

2.1.1 Question answering

This paragraph is based on Gao et al. [2] and their explanation of a question answering agent. An question answering agent provides an answer to a user query. This answer should be concise and direct. To generate this answer the agent will use a data source such as a text collection. There are two types of question answering agents, KB-QA agents, and text-QA agents. The former uses a knowledge base to create the answer, and the latter uses a text collection. Machine Reading Comprehension is the task of answering a question based on a provided passage. Such a model is the core of a text-QA agent.

In the case of this project the data source is a semi-structured text collection, so a text-QA agent will be made. The MRC model will be explained in detail in Section 2.3.1, and how to arrive at the passage for the MRC model will be explained in Section 2.2.

2.1.2 Chatbot

This paragraph is based on Gao et al. [2] and their explanation of a social bot. A social bot should be able to hold a conversation with a user both seamlessly and appropriately. In recent years end-to-end approaches using the sequence-to-sequence framework [3] has been used. An end-to-end approach means whole response generation is handled by the model.

Long short-term memory models [4] has been a popular choice for creating such end-to-end models. But much of this popularity has shifted towards pretrained transformer models [5]. More details on transformer models, and how they can be used as a social chatbot, is provided in section 2.3.1.

2.1.3 Hybrid System

When developing a Conversational AI on might want an AI that can both perform tasks and answer questions, or small talk and answer questions. To do this one could use a hybrid system, where a broker would handle the overall conversation, and multiple skills would be responsible for handling an individual turn [6].

2.2 IR

The field of study focused on retrieving relevant information is called information retrieval. A basic example of IR is: looking up a phone number of a person in a Telephone directory. A common IR tool used today is a search engine. IR is not limited to text, and does also concern other media. In the case of this project the interest is in arriving at the right fact-checking article, which contains the answer to the end users question.

2.2.1 Text Preprocessing

Text preprocessing is the process of preparing the input query for use in retrieval. There are many ways a preprocessing pipeline can be designed, depending on the collection at hand, and the retrieval model to be used. Stopwords can be removed, which are words of little meaning to the query. A stopword list can be designed in multiple ways, but it is common to include words such as: "the", "and", and "a". It is also common to tokenize the words. Tokenizing a sentence means splitting the query into tokens, where tokens can be a character, word, or what suits the task at hand. This means the tokenization process can be as simple as to splitting a query into tokens on every space. It could also involve a more complex technique to create lexical correct tokens.

2.2.2 Inverted Index

An inverted index is used to find documents based on the words it contains. Commonly words are kept as keys and documents as values for the index. Other information such as the frequency of the word in the document, can also be stored. The reason for creating such an index is for use in a ranking function. The drawback of using an inverted index is the cost of creating and maintaining such an index. The advantage is in removing the need for going through the whole collection of documents when ranking, thus decreasing the computational cost of ranking significantly. An example of how an inverted might look is shown in Table 2.1.

Key	**Value**
also	document 1, document 2, document 3
theme	document 2, document 4
world	document 1
zebra	document 4

Table 2.1: Example of an inverted index.

2.2.3 BM25

Best Matching 25 is a ranking function commonly used in IR. It is bag-of-word based, meaning it uses the words and their frequency to calculate the score. The use of BM25 in this project is for initial document retrieval.

2.2.4 Elasticsearch

Elasticsearch is a search and analytics engine [7]. It supports both structured and unstructured data [7]. Elasticsearch handles the storage of the data, and creates an inverted index. It also has BM25 built in for ranking the documents. To interact with Elasticsearch it provides a REST API. The collection of fact-checking articles used in this project were provided through an Elasticsearch instance. This instance, with the articles, was set up by the supervisor.

2.2.5 Initial Retrieval and Re-Ranking

In IR it is common to divide the ranking process in two. Better performing scoring models or functions, usually require more computing power. Therefore an initial retrieval is first done over the whole collection. From this initial retrieval a more advanced model is used to score the top n documents again. In short doing an initial retrieval, and then re-ranking, shortens the computing time used to retrieve top n documents.

2.3 NLP

Natural language processing is the field focused on processing natural language for computational analysis. The meaning of this is to find some information in the natural language presented.

2.3.1 Transformer Models

Vaswani et al. [5] and the title of their article "Attention Is All You Need", which introduces the Transformer network architecture, summarizes the the architecture well. The architecture has been shown to perform well on many different NLP tasks [5]. Another advantage is the possibility of parallelization [5]. Multiple different models based on this architecture has been created. In this project some of these models will be used to solve many of the NLP tasks present, for creating the conversational AI.

Tokenization

Before using a Transformer model the input text must be processed into tokens, which the model can work with. These tokens are translated into token identification numbers. A corpus is used to track the translation from word tokens to integer tokens. As for the actual word tokens this can be as simple as the tokenization mentioned in 2.2.1. Most of the Transformer models use a more advanced form of tokenization. One such tokenizer is WordPiece [8], which BERT [9] uses. WordPiece is a balance between word an character tokens, which results in improved accuracy [8]. To distinguish different parts of the input it also uses special tokens. Such tokens could be "[CLS]" and "[SEP]" which BERT uses, for classification and separating input respectively [9]. The naming of these tokens vary between the different tokenizers.

Huggingface's Transformers

Huggingface's Transformers library [10] is a library making pretrained Transformer models easily available for use. The library provides an API for use of the models. A website is used for browsing the different models available, and the models can also be tested in on the website. This was used, both to explore models to use, and for simple download and use of multiple of the models in the project. The library also provides the appropriate tokenizer for the models, where the function also handles the insertion of special tokens where they are need.

BERT

Bidirectional Encoder Representations from Transformers [9], as the name implies is designed to capture the context of a word from both directions. BERT is pretrained on BookCorpus [11] and English Wikipedia, using two different tasks, Masked LM and next sentence prediction [9]. Along with the token embedding, a segment and position embedding is provided [9].

To rank passages using BERT, two passages are input to the model. The first passage being the query in this case, and the second being the passage of the document to score. On top of the "[CLS]" there is an output layer, which can be used to score relevancy of the passage [9]. The model used in this project was fine-tuned by Reissel and Manaj [12], they state that they finetuned using the method developed by Nogueira and Cho [13]. This method by Nogueira and Cho [13] achieved state of the art performance on the TREC-CAR dataset, when released in 2020 [13].

The solve the MRC task BERT fine-tuned on SQuAD [14] is used, which was released by Devlin et al. [9], the creators of BERT. This paragraph is based on Devlin et al. [9] and their description of using BERT for question answering. The question and the passage containing the answer is provided as input, embedded as two different segments. As a start and end vector was introduced in training, each token can be scored by taking the dot product of theses vectors and the tokens. Each token now has a start and end score. Candidate spans can be created by adding the score of a start and token together, where the end token comes after the start token. The highest scoring candidate pair is chosen as the answer. The model is also trained, to know when an answer is not present in the passage. In that case the "[CLS]", which is always the first token, will be highest scoring, for both start and end. A simplified example of how BERT for question answering works is shown in Figure 2.1. Keep in mind the model works on token ids and not words, and a question would also be present.

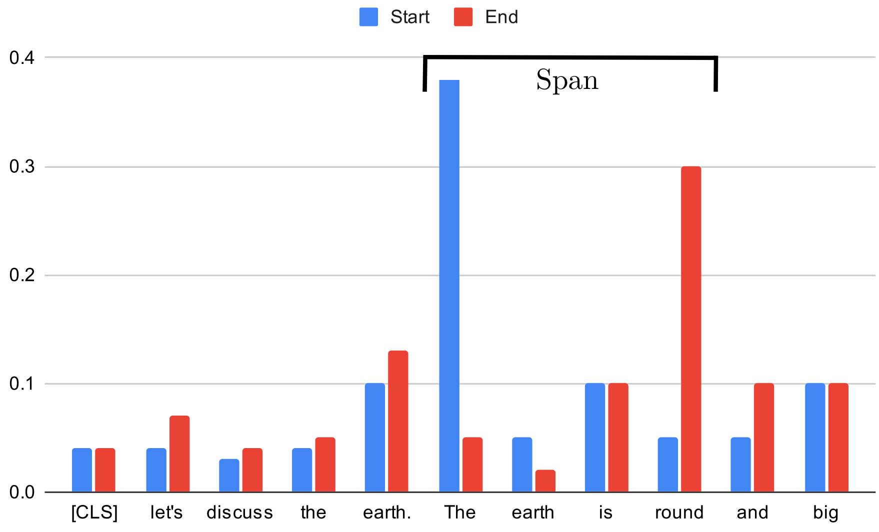

Figure 2.1: A simplified example of how BERT is used for QA, where the posed question could be: is the earth flat? The answer would be the marked span.

GPT-2

GPT-2 created by Radford et al. [15], is a large transformer model. Their largest model contains 1.5B parameters, and they have shown that increasing the amount of parameters result in a better performing model [15]. To train the model they used a dataset containing millions of webpages, which they named WebText [15].

It should be mentioned that a newer generation, namely GPT-3 [16], has been released. This has 175B parameters, with strong performance on multiple NLP datasets [16]. Due to the large size it would require too much computing resources to use this model.

In this project the interest is in using the model for chitchat. To be able to provide a persona for the AI, PersonaGPT [17] is used. This model is built on the DialoGPT [18], medium version, which in turn is based on GPT-2 [17, 18]. The model is finetuned on the PERSON-CHAT dataset [19] which enables the use of a persona for the chitchat model [17].

BART

BART, introduced by Lewis et al. [20], is a denoising autoencoder, which combines the bidirertional encoder of BERT, and the left-to-right decoder of GPT [20]. To train the model they used five different methods, where the goal of the methods is to corrupt the document, the reconstruction loss is then used for optimization [20]. For use in generating summaries Lewis et al. [20] used the CNN/Daily Mail dataset [20].

Summaries can be either abstractive, or extractive. Where abstractive summaries generates new text, while extractive summarizes by extracting the key sentences from the text. The summaries created by the BART model, finetuned on the CNN/Daily Mail dataset, tend to be more extractive [20]. This is due to the summary sentences in the finetuning dataset being similar to the sentences contained in the text to be summarized [20].

It should be mentioned that Zhang et al. [21] created a model, named PEGASUS, for summarization. This model has better performance than the other models on multiple summarization datasets [21]. On the CNN/Daily Mail dataset the ROGUE results between BART and PEGASUS are comparable [21]. The reason for going with BART is the smaller amount of parameters, where BART has about ten percent more parameters than BERT (340M parameters), resulting in around 374M parameters [9, 20]. while PEGASUS has 568M, which is around 52 percent more parameters [21]. The parameter sizes used here are for the large models. If abstractive summaries were to be used, PEGASUS would be a better choice, but using a smaller model requires less time, or computing resources, to generate the summaries. This was prioritized over getting abstractive summaries.

2.3.2 NER

Named Entity Recognition is the process of finding entities in a text. The entities found are then grouped into categories, the categories can vary based on NER system used. Some examples of entities and a category they could be put into is shown in Table 2.2. For NER in this project, spaCy [22] is used.

Entity	**Category**
Donald Trump	Person
Jens Stoltenberg	Person
Norway	Country
Dollar	Money

Table 2.2: Examples of entities and categories for NER.

2.3.3 Coreference resolution

Coreference resolution is the task of resolving coreferences in a text. A coreference is when a word references another word. An example of a sentence with a coreference and a resolved coreference is shown in Figure 2.2.

Her name is **Annie**. **She** loves to take long walks on the beach.
Her name is **Annie**. **Annie** loves to take long walks on the beach.

Figure 2.2: An example of a summary that was created.

To resolve coreferences in this project the end-to-end coreference resolution model introduced by Lee et al. [23] is used. The implementation used is from the AllenNLP platform [24].

2.3.4 Intent classification

To classify intent DeepPavlov's intent catcher [25] is used. This intent catcher uses a Transformer model to classify the intent [26]. The model has multiple fully connected layers on top [26]. This is then fitted on Universal Sentence Encoder [27] embeddings [26].

The intent classifier uses regular expressions in the training data to upsample the data [26]. Not much information is provided on how the regular expressions should structured so this was inferred from their tutorial notebook [28]. This means the description might not be correct, or lacking, but this worked when creating the training data in this project. The "|" character is used as an "or" meaning one of the provided text should be selected. The text which the "or" selection should select between needs to be encapsulated in parentheses, then encapsulating the whole selection. Any encapsulated text can be set to optional by providing "{0,1}" after the closing parenthesis. An example of a sentence with the regular expression, and the resulting sentence is shown in Table 2.3. It is also possible to nest the rules described, resulting in more advanced data, and therefore more samples.

Regex data	**Up-sampled data**
not (feeling){0,1}so ((good)|(great))	not feeling so good
	not feeling so great
	not so good
	not so great

Table 2.3: An example of a sentences and the resulting samples.

2.4 Fine-tuning Datasets

While this project uses already trained and fine-tuned models, except the intent classifier, where the dataset was created and the model fine-tuned on this. The fine-tuning datasets for the transformer models provides insight to how they were trained to perform the tasks at hand.

2.4.1 SQuAD

The following information is from Rajpurkar et al. [14], the creators of SQuAD. Short for Stanford Question Answering Dataset, it consists of over a hundred thousand questions. The questions were asked by crowdworkers, and the answers are a segment of a passage from a wikipedia article. A new version of the dataset was released, also containing questions where no answer exists in the passage.

2.4.2 MS MARCO

The following information is from Nguyen et al. [29], the creators of MS MARCO. MS MAchine Reading COmprehension dataset is a collection of large dataset containing over a million questions, taken from the Bing search engine, meaning it is questions posed by humans. It contains almost nine million passages. And the datasets were created for use in question answering, MRC, and passage ranking. In this case the interest is in the dataset made for passage ranking.

2.4.3 PERSONA-CHAT

Zhang et al. [19] addressed the issue of specificity and consistency in personality, for chitchat models [19]. Along with this they created the dataset PERSONA-CHAT [19]. The data collection consisted of three stages, personas, revised personas, and persona chat [19]. The result is a dataset containing conversations, based on personas [19].

2.4.4 CNN/Daily Mail

Hermann et al. [30] proposed a new dataset for teaching models reading comprehension. The dataset was created by collection a large amount of news articles from CNN [1] and the Daily Mail [2] [30]. Through the use of NER and anonymization algorithms they made the articles into a dataset containing: context, query, and answer [30].

2.5 Spelling Correction

To correct spelling errors DeepPavlov's spelling correction model [25] is used. This model is based on the spelling correction model proposed by Brill and Moore [31] [26]. Their

[1] https://edition.cnn.com/
[2] https://www.dailymail.co.uk/home/index.html

model uses a large dictionary of words, and the model is trained to provide the probability of a word being a word in the dictionary [31].

2.6 Fact-checks

Fact-checks are an important part of this project. As the goal is not for the Conversational AI to do the fact-checking, but to use existing fact-check to answer questions. These fact-checks come from multiple sources. The fact-checks were stored in a remote Elasticsearch instance, which could be accessed through the REST API. This Elasticsearch instance, with the collection of fact-checks, were provided by the supervisor. The fact-check are stored in a semi-structured manner. Each fact-checking article is given an identification, and the content is stored in fields. Each field identifies a specific part of the article. The fields vary depending on the source it comes from. Some fields are present in all. The fields present for all articles were focused on in this project. The fields of interest are listed in Table 2.4, along with a description of the content of the field.

Field name	Description
claim	The statement which is the claim originating the fact-check.
label	The verdict of the claim given by the fact-check. That be true, false, or mixed. The actual name given for labels vary depending on the source.
doc	The main content of the article.
fact_source	The source at which the fact-check comes from.
url	The URL that points to the original fact-check article.
summary	A summary of the article. Only a few articles had this field populated.

Table 2.4: The fields in the fact-checks collection used.

It should be mentioned that this collection of fact-checking articles is not provided along with the source code. So to use this Conversational AI one must first create an Elasticsearch instance. As long as the collection contains the fields listed in Table 2.4, with the described content, it should work.

The collection of fact-checking articles was set up to update automatically. Therefore the amount of articles will continuously increase. Thus approximate values will be used, when addressing amount of articles. With this in mind the collection consist of around 43 000 articles.

Chapter 3

Design and Architecture

3.1 Introduction

This chapter will discuss the design choices made in the creation of the conversational AI. It will also explain the architecture. Some existing approaches will be mentioned, before explaining the overall architecture. And then going into details about the different components.

3.2 Existing Approaches

There exist a large range of Conversational AIs, especially in the commercial scene. Customer support is one place where it is often used. Many questions are frequently asked where the tasks or answers can be automated. In this case a Conversational AI can reduce the need for customer support staff, where they are only need for the tasks and questions the conversational AI can not answer or solve. Home assistants is another area where conversational AIs are commonly used. Many of the large tech companies have their own versions, such as Google's Google Assistant, Amazon's Alexa, and Apples Siri. There are many other commercial areas where a conversational AI might be used, but these are the ones that most people would be familiar with. Since these approaches are commercial there exist no literature on their approaches so these can not be used for reference.

There currently exist no approach on a conversational AI for serving existing fact-checking articles, to the authors knowledge. This means that there exist no direct comparison to the conversational AI that will be created here.

3.3 Proposed Solution

There are many approaches that would be fitting for a conversational AI that would serve fact-checks. From the categories mentioned in section 2.1 a question answering agent would be most fitting. Where the user would ask about a claim and the agent would respond with an answer about the claim, using the relevant fact-check. A problem arises with using this approach alone, the conversational AIs capabilities and use case becomes limited. This was experienced early in the project, where it became obvious that it is hard to come up with multiple claims on the spot. This limits the use case for the conversational AI mainly to the following scenario; The end user sees or remembers a claim they want fact-checked.

Based on the observation mentioned in the previous paragraph. A hybrid system where multiple agents are combined together could expand the use case of the conversational AI. Along with the previously proposed agent, an agent for serving a random fact-check could be added. This would make the Conversational AI usable without needing to have a claim at hand. Thus the end user could use the conversational AI at any time and not only when a question comes to mind.

Another problem that arises is overall conversation would not seem natural, as it would only respond to fact-checking questions or requests for articles. A typical human conversation would incorporate chitchat elements around the actual question and answering turn. An example of such a conversation is shown in Table 3.1.

Turn	Speaker	Message
1	Human 1	Hello.
	Human 2	Hi, how can I help you?
2	Human 1	Is the earth flat?
	Human 2	No, the earth is round.
3	Human 1	Thank you.
	Human 2	You're welcome.

Table 3.1: A conversation with a question and answer turn. Also showing a preceding and succeeding turn.

It would be natural in a human conversation to also have some chitchat elements in the same conversation where the question was asked. A solution to handle this would be to also include a chatbot agent. This hybrid system would then use a broker to control the overall conversation and choose the appropriate skill. That be chitchat, providing a random fact-check, or any other skill created. It should be mentioned that there exist multiple open source frameworks for creating conversational AIs, such as RASA [1] [2]

[1]https://rasa.com/

[2]They have both an Enterprise, and open source solution

and DeepPavlov[3]. It was decided to not use a specific framework and instead combine different components from multiple packages to create the Conversational AI, and some parts come from the DeepPavlov framework. This gives full control over the whole stack of the conversational AI.

Based on the earlier expressed design needs the following components have to be created: A user interface, so the end user can interact with the system. A broker to handle the overall conversation, and non skill specific parts. Multiple skills will then handle a single turn. And a DBMS for handling long term storage. The different components and the interaction between them is shown in Figure 3.1.

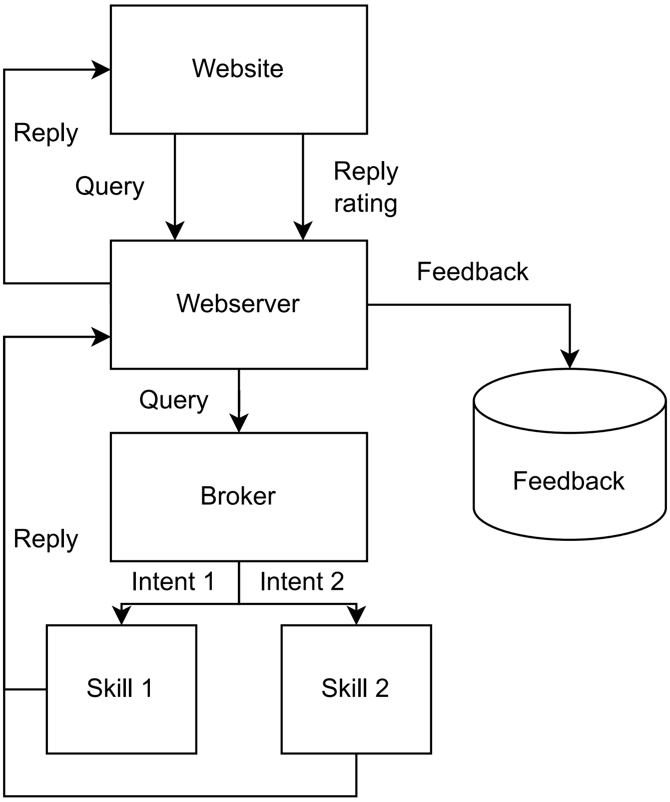

Figure 3.1: Interaction between the different components.

This design draws inspiration from the design of the RASA [32], and DeepPavlov [25] frameworks. All the deep learning models for the AI should be loaded into memory when the AI first starts up. This will reduce the time taken to generate a response, as opposed to loading a model when it is need. Only loading a model when needed would result in longer response time, but less memory requirement. For the conversational AI the response time should be as fast as possible, therefore response time should be prioritized over resource requirements. An assumption that should be made about the user input is: In many cases the input text will not be properly letter cased. There are two possible

[3]https://deeppavlov.ai/

solutions to this, preprocess the input to properly letter case. Or use models that are case-insensitive. The latter solution will be used, where possible.

The next part will go into details on the design and decisions of the different components that were established.

3.3.1 User Interface

Even though the user interface used in this project is a website. The conversational AI was designed in such a way that it is adaptable to other user interfaces. Other types of user interfaces could be a social robot or a web browser extension. Ideally for simplicity the communication between the server component and the conversational AI would be user input in and a response back. Since a website needs to handle multiple users at the same time, multiple conversations need to be tracked simultaneously for the broker. The tracking of the data for the broker is therefore done using web server sessions. But implementing such tracking for other user interfaces would not require much work. Another approach would be to store this data in a database. But in the case of a user interface where multiple end users can interact simultaneously an identification must be sent to the broker to know which conversation data that should be used. Therefore for simplicity and faster implementation session is used. The reason for not only providing the response from the conversational AI is to display extra information. If the user interface does not support displaying extra information the AIs reply could be extracted from the response dictionary before returning the response.

The next part will go into the design of the user interface created in this project. This will also include the backend design which will communicate with the website. The design of the web page, which represents the user interface is shown in figure 3.2.

The conversation itself is shown inside a chat window. Using a familiar type of interface for displaying conversations makes it more intuitive to use. The messages are appended under the last message. The AI's message is shown to the left, while the end user's messages are shown to the right. And old messages are scrolled of the window when new messages need space. The topic selection button's serve two purposes:

- Incentivize and assist the user on a topic to converse with the AI about.
- Provide a topic for use with multiple of the skills.

As for the topics to use for the topics selection buttons, this could be any topic. A currently relevant topic would make most sense. The number of buttons, and thus the

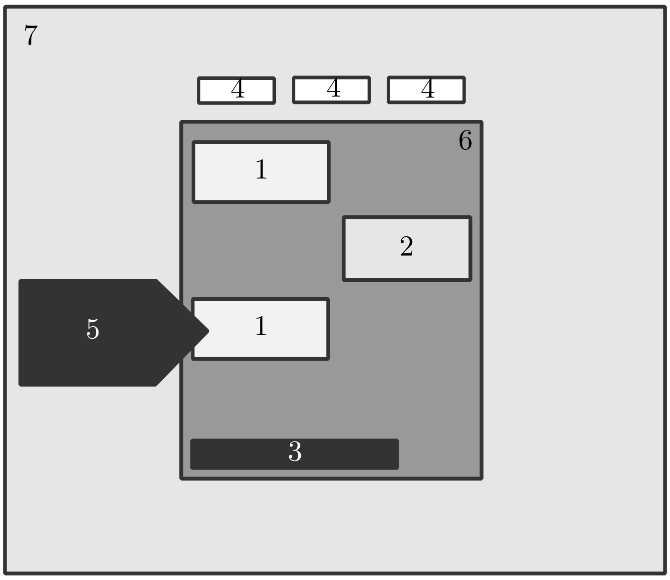

Figure 3.2: Design Of The User Interface. 1. AI's messages. 2. User's message. 3. Input field for the user's messages. 4. Buttons for selecting topic. 5. Tooltip for displaying extra information. 6. Chat window. 7. The webpage.

amount of topics can be changed. One button should also be for all topics. From the brokers perspective this means no topic is selected.

The Tooltip will display information related to the article used for generating message, and is attached to that message. The Tooltip contains the following information:

- Claim, which originated the fact-checking article.
- Label, meaning the verdict given by the article.
- The site which authored the article.

The reason behind providing this information is, the message from the AI may not be correct. By including the extra information the end user can check the message against the information, which can aid the end user in drawing an appropriate conclusion. A link to the original fact-checking is also provided in the response message. The end user consequently has access to the full information, that be because the message and extra information still was unclear. Or simply having an interest in reading the article.

The backend of the website serves a few purposes, display the user interface (website), forward the communication between the AI itself and the front end. Send requests to the DBMS to store the feedback. The final purpose is storing conversation information, through the use of sessions. As mentioned earlier the conversation information is technically a part of the broker, but stored in sessions for simplicity.

3.3.2 Broker

The input from the user interface will be forwarded to the broker. The different components and interactions between the different components of the broker are shown in Figure 3.3.

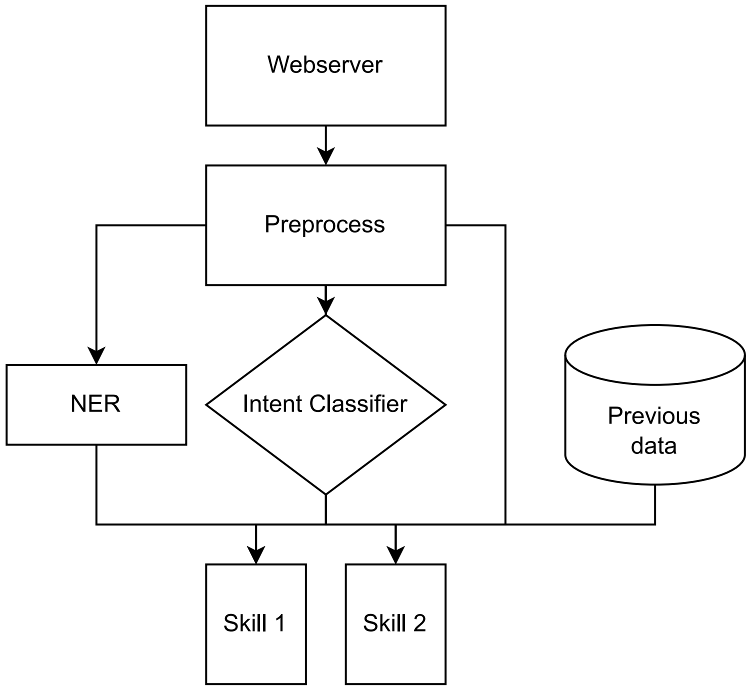

Figure 3.3: Flow Of The Broker.

When the broker receives the user input from the web server, it is preprocessed, where the preprocessing consists of the following, steps in the listed order.

1. Lower-casing all characters.
2. Spelling correction.
3. Expanding the contractions.
4. Resolve coreferences.

Lowercasing all the characters and expanding contractions is for consistency, meaning slightly different queries with the same meaning will render the same result. As the broker receives the raw user input it is possible to have spelling mistakes. therefore spelling correction will remove the wrong responses, which are a direct result of spelling mistakes. To handle multi-turn questions the coreference need to be resolved. This is because multiple skills only use the input from the current turn and would therefore not know what was coreferenced.

Some of the skills can use a topic provided in the input. To identify the topic mentioned, NER is used. The interest is not in what type of entity that was recognized, but that an entity was found. Therefore the entity is provided to the skills, but the category of entity is discarded. Data from earlier turns is used in some of the skills, so this is also provided to the skills.

The intent classifier is responsible for choosing the skill that should handle the input and produce a response. Based on the provided preprocessed input text it will select one skill suited to handle the request. The intent catcher does not provide any input to the skill, only selects which skill to use. Only the selected skill will be run to produce a response. The broker is designed with expansion of skills in mind.

3.3.3 Skills

The skills perform the task of generating a response based on the given input. Seven skills where created in total for this conversational AI. The main skill is the fact-checking skill. The user can ask about a claim, and the AI will use the collection of fact-checking articles to create a response. The skills can take, no input, one input or multiple of the inputs listed bellow.

- The preprocessed text input from the end user.
- A topic chosen by the end user through the user interface.
- The entity found by the NER component.
- The stored data from previous turns.

Since there are two ways of providing a topic to the skills, a decision on which to use should be made. This was solved by using priority. An entity recognized by the NER component should be prioritized, over the selected topic from the user interface. This means that the topic selected through the user interface is only used if there is no topic provided in the text input. The entity found by the NER component is prioritized as this was specifically mentioned by the end user.

The next part will describe the skills and their design, where Table 3.2 gives an overview of the seven skills that were made.

Random Fact-Check

This skill will provide a random fact-check. A topic for the fact-check to return can also be provided. The returned result will be a summary of the chosen random article.

Skill	Description
Random	Provide a random fact-checking article
Fact-check	Fact-check a claim using existing fact-checking articles
Elaborate	Provide more information about the article
Related	Give a fact-check similar to the last one
Chitchat	Small talk
Count	Provide a count of articles about a topic
List	Provide multiple articles

Table 3.2: The different skills for the converstainal AI.

Fact-Check

The fact-checking skill will try to answer a claim input by the user. This is done by finding the relevant fact-check article from the collection. A question answering model will use the relevant article to find the phrase best answering the question given. Figure 3.4 shows the different steps taken to generate a reply based on the input query.

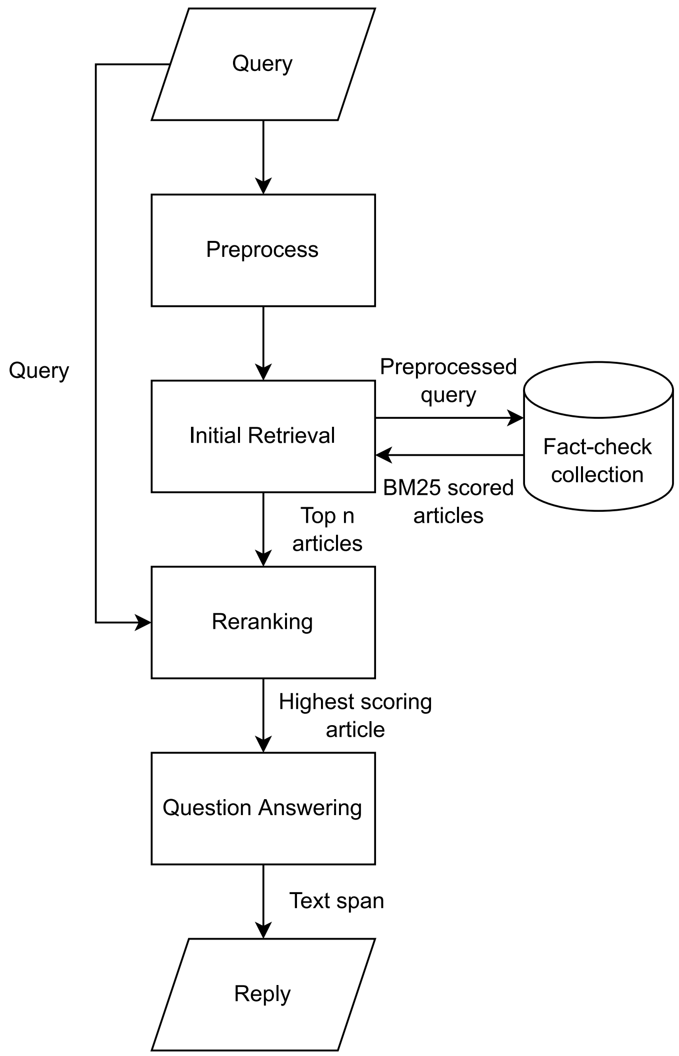

Figure 3.4: The factchecking skill

Due to working with a collection of semi-structured documents, as opposed to a knowledge base, the relevant article must be found. To find this article an initial retrieval is preformed. See Section 2.2 for more information on the process. The initial retrieval uses BM25 and to get better results from the function the query is preprocessed. The following preprocessing steps are used:

- Remove special characters.
- Split into tokens.
- Remove stopwords.

There are more and different ways to preprocess the query. But since the collection is small, as opposed to a search engine for the internet. The difference between the articles should therefore be more significant. The reranking of the retrieved document, will use the original query. As long as the article is among one of the highest scoring it is no problem, due to the reranking. The rerank will be done using a transformer model finetuned for the reranking task.

Using the found relevant article a response must be generated from it. To generate the response a transformer model, finetuned for question answering, is used. The model marks a span that is most likely to be an answer to the query. This means it does not generate new sentences, but rather uses sentences from the article to create a response. This is not ideal, as the responses are not necessarily a direct response to the question, since it must be present in the given passage.

Elaborate On Fact-Check

The following skill exists to supplement the fact-checking skill. The fact-checking skill only provides an answer to the question it does not provide much information on the content of the article. This skill will expand upon the information provided. The elaborate skill uses the stored identification of the previously used article to look up the summary associated with the article. This summary is returned as the response.

Related Fact-Check

This skill is similar to the random fact-check skill. But instead of providing a random fact-check, where a topic can be provided, it uses the topics attached to the previous retrieved fact-check. This means this skill only works if a fact-check has previously been returned. The way it works is that it retrieves the last used article's topics. These topics are then used with a random search to provide a new article. For this new article the summary is sent as a response.

Chitchat

As mentioned earlier the conversation should be more human like. To achieve this a chitchat skill is introduced. It handles the small talk the user will input between the questions. To know the context of the conversation it will use the whole conversation history. A persona is also provided to make the AI more engaging.

Count Articles

This skill will provide the user with information about the collection of fact-checks. The specific information it provides is the amount of articles it contains on a specific topic. This is achieved by getting the topic from the NER component of the broker. If no topics are provided it will return the total amount of articles in the collection.

List of Articles

Similar to the random fact-check skill, the list of articles skill returns random articles. Rather than returning a single article it returns multiple. This skill differs in response, as opposed to the other skills this returns a list of replies. This is so each article can be displayed as a separate message. If the user interface does not support this it can be combined into a single response. Another difference is that this skill returns the claim rather than the summary of the article. This is because a summary is a significant amount of text. To avoid overwhelming the end user with text, only the claim is shown.

3.3.4 Summary

Since many of the articles in the collection do not have a summary, it must be generated for these articles. This will be done using a Transformer model fine-tuned for the summarization task.

The summaries of the articles are stored along with the source code, due to not having write access to the Elasticsearch instance. Ideally they would be stored in the summary field of the fact-checks, on the Elasticsearch instance. This way all the data on the collection would be store in a single place.

The summaries can be stored in a single table. This is because only an identification, for linking the summary to the appropriate article, and the actual summary is needed.

3.3.5 Feedback

To get results on performance of the conversational AI, feedback needs to be gathered and stored. The feedback is anonymized, as there is no need for the feedback to be traceable back to the originator. The whole conversation and the order of the turns should be tracked. By tracking the whole conversation it is possible to get a more holistic view of the interactions with the AI. The rating of the response is tracked through a star rating system. The star ratings are ordinal and ranging from one to five stars. Where one star represents a not satisfactory response and five stars being a satisfactory one. A response can be more nuanced than wrong or correct, therefore three ratings are in between.

Based on parts discussed in the previous paragraph the database can be designed. No relations are needed therefore all data can be stored in a single table. The table should contain the information listed bellow.

- Conversation identification.
- Turn identification.
- End users input to the AI.
- The response provided by the AI.
- Rating provided by the end user.
- The identified intent.
- The version number of the AI.

The version is added to be able to distinguish between different deployments of the AI. Different components can be disabled, or the AI could have been built upon further. When the feedback from different versions are added together they are still distinguishable because of the version number. Storing the user input and the output from the AI provides more insight. Using this information it is possible to identify trends in input that give correct answers, and vice versa. This helps identify which skill or component that should be focused on for improvement.

Another use case for the feedback is further training of the models used in the AI. The plan was to do this, but due to issues with deployment this had to dropped, as the feedback were collected to late, and the deployment consumed to much time.

3.3.6 Storage

The storage of the feedback and summaries must be handled in some way. This will be handled by a DBMS. For simplicity and easy deployment a system integrated with the programming language is used. This way the database can be managed directly in the code. The data can also be stored directly in a file with no external software needed. The summaries and feedback are stored in separate files. This way the feedback can be retrieved without the need for downloading summaries.

Chapter 4

Implementation

4.1 Introduction

This chapter will explain how the design of the Conversational AI described in Chapter 3 were implemented. The ordering of sections will be similar to Chapter 3, with minor changes where needed.

The Conversational AI is mainly programmed in the Python programming language, due to large support for machine learning. The user interface also includes HTML and JS. The DBMS is interacted with through SQL.

4.2 User Interface

As mentioned in Chapter 3 the user interface used is a website. Since the website must handle communications back and forth, as opposed to a static website, a web framework is used. Namely Flask [33], which was chosen due to familiarity.

4.2.1 Fontend

The conversational AI will require a single page with the layout described in Figure 3.2. Some common website elements are added on top, which are not strictly used for the functioning of the AI. Such as a navigation bar and an "about" page. A introduction is displayed the first time visiting the site. This is to introduce the end user to the capabilities of the AI, and therefore how it can be used. A disclaimer is also shown beforehand, but it is not strictly a part of the project so the implementation will be omitted. What the end user sees when first visiting the website is shown in Figure 4.1.

Mobile devices was not taken into consideration for the implementation, so it will not function properly on such devices.

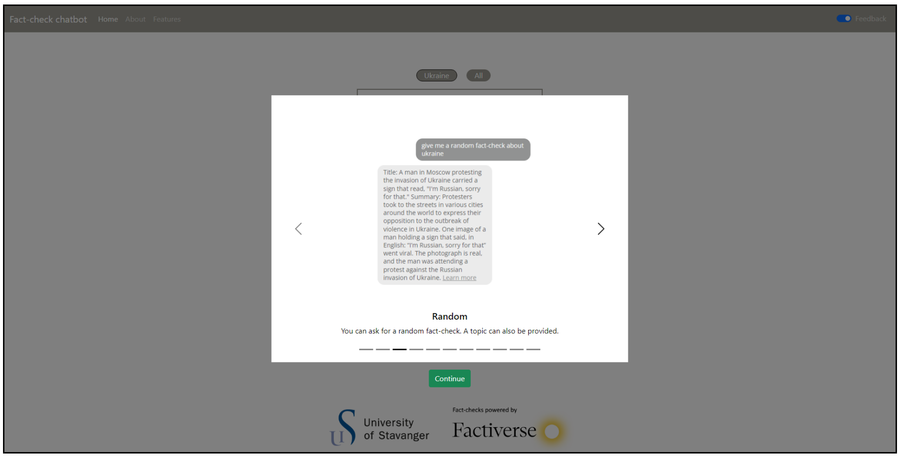

Figure 4.1: The view of the website when visiting for the first time. An introduction is displayed to introduce the end user to the AIs capabilities.

The introduction is displayed as a slideshow, with a image showing an example and some explanatory text. The text is kept concise to retain the attention of the end user. The implementation of the slideshow is made with Bootstrap 5's [34] carousel component. A button is added so the carousel can be dismissed, meaning the parent element is set to hidden. A button exist in the navigation bar to reopen the introduction. The web page after the introduction has been hidden is shown in Figure 4.2.

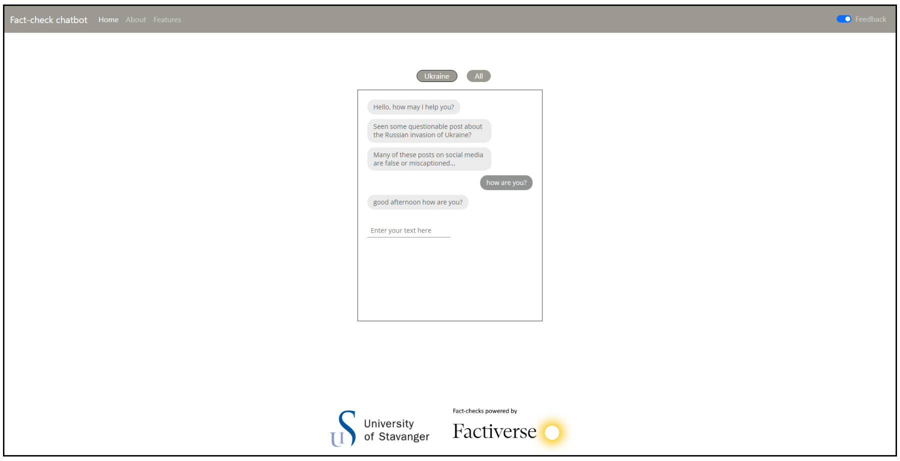

Figure 4.2: The implemented web page.

The topic selection buttons are implemented as a list with CSS to style the list items as buttons. An "onclick" event listener calls a function which changes the topic and the CSS, to indicate the topic selected. The chat window is implemented using a JS framework called BotUI [35]. This handles all the visual elements of the chat window, and comes with an JS API for interaction [35]. The BotUI JS API has three concepts for interacting with the chat window [35]. These are message, action, and use of "then" [35]. Message are used for displaying a message in the chat window [35]. Actions are used for displaying input in the chat window [35]. The "then" concept is to add a callback from the other two concepts [35]. Knowing these concepts the interaction loop can be created. The flow of the interaction loop is shown in Figure 4.3.

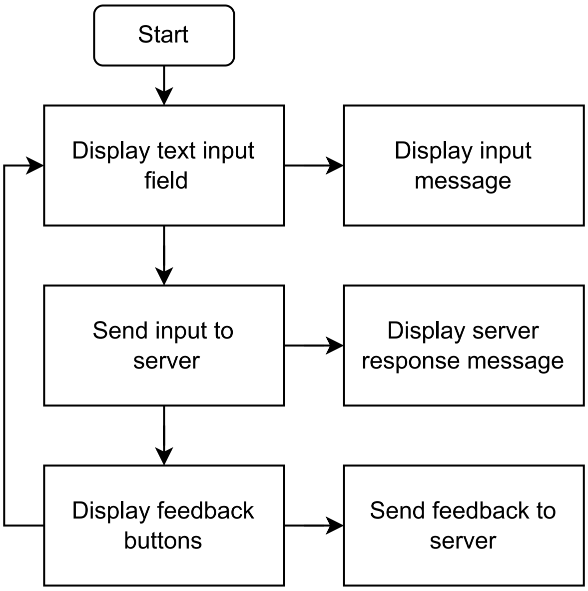

Figure 4.3: Flow of the chat window interactions. Where every transition is made on a callback.

The communication between the conversational AI and the web page should not result in a reload of the full page. To avid this the messages are sent asynchronous, by using AJAX. The AJAX requests sent from the web page are shown in Table 4.1. The different endpoints these messages are sent to are explained in the next subsection.

Description	**Key**	**Value**	**On callback**
Request to update topic	Query	Integer representing topic	Do nothing
Request to store feedback	Query	Integer representing rating	Do nothing
Request to interact with conversational AI	Query	End user's message to the conversational AI	Continue interaction loop

Table 4.1: The different AJAX requests, their content and what is done when the callback is received.

Depending on the skill that generated the response there can be different components that should be added to the message. A link to the fact-check, which the response was generated from and a Tooltip displaying information of the article.

The BotUI framework does not support Tooltips, so this is added to appropriate element after it is displayed. To create the Tooltip the Tooltip plugin for Bootstrap 5 [34] is used. BotUI does support links through a subset of Markdown so this is used to display links [35].

4.2.2 Backend

The web server backend of the website is not complex as it's purpose is to: serve the requested pages, or forward requests to the conversational AI. As mentioned in Chapter 3 the data tracking for the broker is done using sessions, and therefore done on the web server. The endpoints created are shown in Table 4.2.

HTTP request method	**Endpoint**	**Description**	**Input**	**Response**
GET	/	Index page	None	Index.html file
GET	/about	Description page	None	About.html file
GET	/_search	User input to AI	query parameter	AI response as JSON
GET	/_topic	Topic selection value	query parameter	ok as JSON
GET	/_feedback	Feedback rating	query parameter	ok as JSON

Table 4.2: The endpoints implemented in the backend.

The first two endpoints respond with the requested pages, and last three endpoints handle the AJAX requests established in the previous subsection. The topic endpoint changes the topic to the requested one in the session data. The feedback endpoint stores the feedback data. The search endpoint stores the tracked data in the session, and forwards the end users input to the broker. When the response is sent back from a skill it is forwarded to the appropriate client as JSON.

4.3 Broker

Following the design of the broker shown in Figure 3.3, the input is first preprocessed. The NER and intent detection is performed on the preprocessed query. The appropriate skill is selected and provided with its required input, and executed.

4.3.1 Query Preprocessing

Lowercasing all characters is straight forward. Correcting spelling errors is done using the Brillmoore model from DeepPavlov [25, 31]. Using the model requires calling the model with the input query, and the corrected query is provided as the output. The contractions are expanded using the contractions Python package [36], by calling the fix method. To resolve the coreferences the AllenNLP package [24] is used, which implements the model described by Lee et al. [23]. The process of resolving the coreference is shown in Figure 4.4.

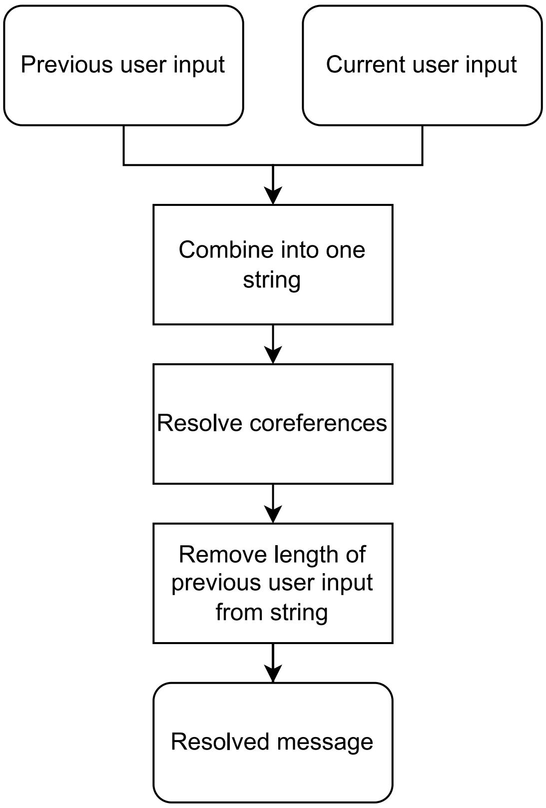

Figure 4.4: The coreference resolution process.

With this implementation an assumption is made, the end user's coreference is contained in the previous message. This means that if the end user's reference is not in the previous turn, the coreference resolution will not work.

4.3.2 Topic identification

The topic identification is done using NER, with the SpaCy package [22]. It should be mentioned that DeepPavlov also has a NER component, but this gave errors when tried. The Spacy model used is "en_core_web_lg" [37]. Although the difference in precision and recall between the small and large model are 0.01, for precision , and 0.02

for recall [37]. Minor tests showed that the small models had issues with input that was not properly cased, therefore the large model was used. The SpaCy models seem to be case sensitive as the recall achieved when testing is lower than the one provided in the documentation [37]. SpaCy is therefore not ideal based on the assumption made, that the message from the user will not be properly cased.

To get the named entities from SpaCy the preprocessed query is used as input to the model. The result contains a list of the named entities found, along with the category it belongs to. This list can be iterated over to extract the named entities found, which will be the list of topics that can be provided to the skills.

4.3.3 Intent detection

To select skill, which should generate the response, an intent classifier is used. As the skills created are specific to this conversational AI the intents are also specific to this AI. This means an already fine-tuned model can not be used, and the dataset must be created.

Dataset

The dataset consists of training data and testing data. The testing data is also used as verification data, as the DeepPavlov training method requires both. The reason for this is to not spend to much time on creating the dataset. All possible ways to specify an intention that could be thought of were added to the training data. The plan was to use the feedback to expand on the dataset, as this would contain other ways people specified an intention. But due to deployment problems there was no time left for this.

The training data is formatted for regular expressions so it can be up sampled, as explained in Section 2.3.4. It was made in a spreadsheet and stored as tab-separated values file, where each column represents an intent and each row is a regular expression question. The first row specifies the name of the intent. Due to the regular expressions in the questions it is not as straightforward as showing the number of rows. A single row can be long and up sampled to ten questions, or short and up sampled to two questions. An overview of the training data is shown in Table 4.3, and examples of training data is shown in Table 4.4.

If an entity was provided it was included in all intents where applicable, to avoid specific entities being learned as being important to an intent.

Intent	**Rows**	**Minimum characters**	**Maximum characters**	**Average characters (Rounded)**
Random	21	21	261	95
Elaborate	10	19	193	71
Factcheck	11	32	142	74
Related	6	45	111	85
Chitchat	16	11	61	32
Count	5	93	190	153
List	4	95	156	122

Table 4.3: Training data information. Where min, max and average are for characters in rows.

Intent	**Example of row**
Random	can you(please){0,1} give me a ((fact check)|(fact-check)|(fact))
Elaborate	((provide)|(give)|(supply)) (more){0,1}((information)|(info)){0,1}
Factcheck	does ((biden)|(trump)|(putin)|(zelensky)) ((think)|(believe)) the earth is flat
Related	((tell)|(show)|(give)|(provide)) (me){0,1}((a)|(an)|(some)) ((similar)|(related))
Chitchat	can you ((help)|(assist))(me){0,1}
Count	((how many)|(count))((about)|(on)){0,1} ((ukraine)|(trump)|(putin)|(russia)|(the invasion))
List	i want ((a list of)|(few)|(some)) ((fact checks)|(fact-checks)|(facts)|(factchecks)|(articles))

Table 4.4: Example of a training data row for each intent.

Training

The training was done in a google colab notebook. The notebook is based on the notebook provided by DeepPavlov for training the intent classifier [28]. The training and testing data mentioned earlier is loaded. Some modification to the training configuration file is made, and the model is trained. The trained model is downloaded and added to the project folder, for use in prediction of intent. The key information regarding the training of the intent classifier is shown in Table 4.5. A line chart showing the progression of the accuracy, sampled every fifth epoch, is shown in Figure 4.5. The testing data consists of 76 sentences.

Predicting

The trained model is loaded by changing the path to the model in DeepPavlov's intent catcher configuration. After the model is loaded it can be used by providing the preprocessed query as input. The model provides the predicted label as output.

Description	**Value**
Original training samples	73
Average generated training samples	425
Minimum generated training samples	409
Maximum generated training samples	436
Testing data count	76
Training epochs	60
Finish accuracy	0.9079
Finish f1 macro score	0.8985

Table 4.5: Key information about the training of the intent classifier.

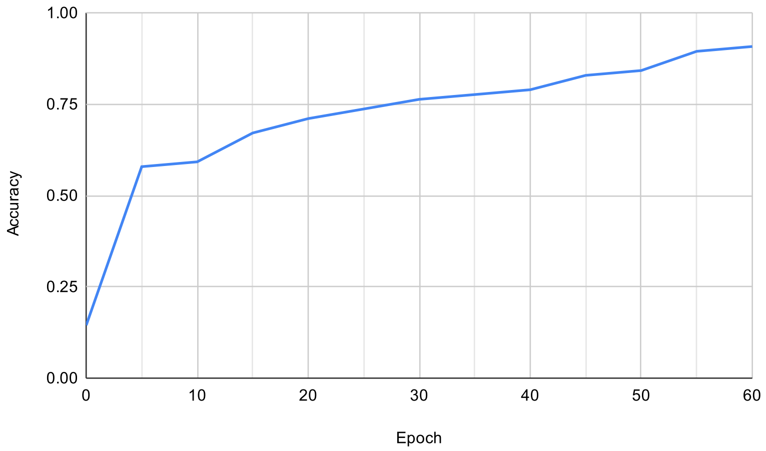

Figure 4.5: The accuracy of the intent classifier through training epochs. Calculated every fifth epoch.

4.4 Skills

This section covers the implementation of the seven skills designed. Due to some of the skills being similar and therefore using the same functions they will be grouped and explained together. So the implementation will not be explained on a per skill basis. But rather the functionality and then the skill built upon this functionality.

4.4.1 Summaries

Since most fact-checking articles do not have a summary, the summaries had to be generated. The model used for generating the summaries was BART large-sized, already finetuned on the CNN Daily Mail dataset, released by the team behind BART [20]. The HuggingFace transformer library [10] was used to load a pipeline handling the whole process, meaning two lines of code is needed to use the model, one to load the pipeline, with the model and tokenizer, and one to use the model [10]. An example of a summary created is shown in Figure 4.6.

A pendulum-style ride at Kankaria Adventure Park in Ahmedabad, India, collapsed on July 14, killing two people and injuring 29. A cellphone video of the incident shows one arm of the ride breaking off and falling. Six people, including the director of the company managing the site, have been arrested in connection with the accident.

Figure 4.6: An example of a summary that was created by the BART model [20].

To store the summaries the database must be created and initialized. Since SQLite 3 [38] is used a single db file is needed for storage. A connection is then made to the file and a single table is created. The table contains two columns: id, and summary. Both columns are of the "text" datatype, and id is set to be unique. A summary can be retrieved by asking for the summary where the id of the row is the same as the one wanted.

To create the summaries for the whole collection, all the fact-checking article identifications from the collection are collected from the Elasticsearch instance, and stored in a JSON file as a list. A separate JSON file is created to store the index in the list of the current article to create summary for, which is initialized with zero as index. This way the fact-checking articles, which a summary has been created for, are tracked. The creation of summaries do not have to be run in a single execution. A flowchart showing the flow of creating a requested amount of summaries are shown in Figure 4.7.

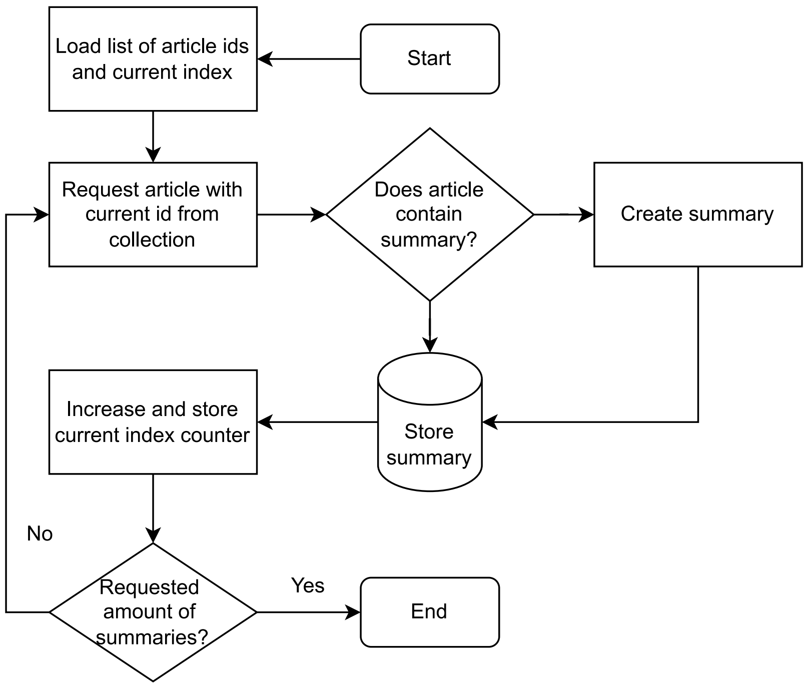

Figure 4.7: The flow of creating summaries.

Due to the large amount of parameters in the BART model it takes noticeable time to generate an output [20]. With the computer used to generate the summaries it was estimated to take around eleven days of non stop execution to generate summaries for the whole collection of articles. To speed up the process multiple virtual machines on the Google Cloud Platform [39] was used. The summary identifications were divided to the different virtual machines and the database file were collected from all the instances and merged into a single database file. This shortened the summary creation time down to a few days.

Since the collection of fact-checking articles are automatically updated, once deployed the database will not not have summaries of the newly added fact-checking articles. To avoid the issue not being able to provide a summary if it does not exist. The database is therefore first queried to check if it contains a summary. If it does not contain a summary, the summary is first created before continuing. This has the added drawback of long response time, when a summary does not exist. A solution to this would be to have a job running which creates and updates the database once in a while. This is a deployment issue and is not covered in this project.

Elaborate On Fact-Check Skill

Since the summaries are already created, the elaborate skill only need to retrieve the correct summary. The skill receives the identification of the previously used fact-checking article. The database is queried for the summary associated with the received identification, and the retrieved summary is the output from the skill.

The identification, which this skill receives, is updated every time the response data from any skill contains an identification. Meaning it is updated when the following skills are called: fact-check, related, and random.

4.4.2 Random

Multiple of the skills use random fact-checks from the collection. The common element for these skills are the query construction, for the Elasticsearch API. To get a random article from the Elasticsearch instance a Boolean query is used. This query contains a must clause, with the scoring function set to random score. This means all the articles in the collection are given a random score, and the highest scoring one can be retrieved, which in turn will be a random one. To get an article on a specific topic a filter is also added to the query. The filtering is made over the terms contained in the topic field. And a list off topics is provided. This means firstly all articles not containing one of the terms provided in the topic field are not part of the scoring. The remaining articles are randomly scored and the top scoring article can be retrieved. This means a random article containing at least one of the provided topic terms are retrieved.

Random Fact-Check Skill

To enable the user to provide a topic for the random fact-checking article, the named entity found by the NER component of the broker is used. If no topic is provided by the NER component, the stored topic is used. This topic or list of topics is used to generate the query as explained earlier. The identification of the retrieved article is then used to get the summary associated with the article from the summary database. This summary, along with the claim, is the response from the skill. An example of a response from the skill is shown in Figure 4.8.

Claim: Photo of injured Ukrainian woman was from 2018 Russia gas explosion. Summary: Olena Kurilo, a teacher, was injured on Feb. 24, 2022, when a Russian missile strike hit her apartment complex in Chuhuiv, Ukraine.Her photograph was taken by at least three journalists that day and she was interviewed on video.Photos in news reports that day show the apartment complex is not the building damaged by a 2018 gas explosion in Russia.

Figure 4.8: An example of a response from the random fact-check skill.

List of Articles Skill

This skill uses the same implementation as the previous mentioned skill. The difference is that in stead of getting one article it retrieves a list of articles. The amount of articles can be set in a separate configuration file. Instead of the summary being used as a reply, the claim is used. Therefore there is no need to query the summary database, and the claim is already contained in the response from the Elasticsearch instance.

Related Fact-Check Skill

This skill uses the topics from the previously retrieved fact-checking article. This means it takes last retrieved article's identification as input. Since the topics for this article is not stored, this must be retrieved first. A request is sent to the Elasticsearch instance to get the topics for the article. The rest of the implementation is the same as the random fact-check skill, but the topics provided are the topics retrieved from the article.

4.4.3 Fact-Checking Skill

The process of finding an answer in a fact-check is divided into the following three steps: Preprocess, document ranking, and question answering.

Preprocessing

To get better results from the inital retrieval the query is preprocessed. The special characters are replaced with spaces using regular expressions. To be more specific, every character that is not not a lowercase letter or number is replaced with a space. The query is split into a list of tokens by splitting on spaces. every token is checked against a list of stopwords from the NTLK package [40]. If the token exist in the list it is removed. The list of tokens is joined into a single string, which is the query used for the initial retrieval.

Document Retrieval and Ranking

The ranking process is shown in Figure 4.9.

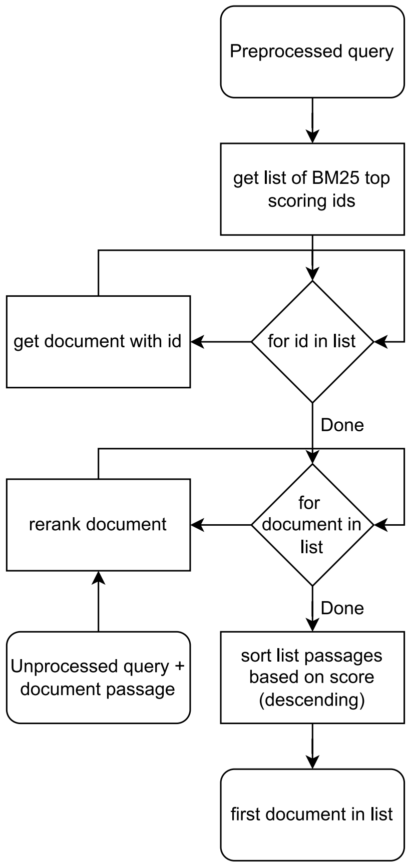

Figure 4.9: The flow of the document ranking process.

The initial retrieval is handled by the Elasticsearch instance, where the preprocessed query is given to the instance. The Elasticsearch has a built in BM25 function [41]. The default values of "b" being 0.75 and "k1" being 1.2 were used [41]. A higher "b" value punishes longer documents, and a higher "k1" score lowers the effect of the frequency of a single word [41]. No tuning of these values were performed as the results from minor testing showed good results.

The reranking is performed using the fine-tuned BERT model by Reissel and Manaj [12]. This model is multilingual and uncased [12]. The input provided to the model is the unprocessed user input, except from the preprocessing done in the broker. Along with this the first passage, combined with the claim of the article, is added. The first passage should work great in the case of the collection in this project. Because the fact-checking articles usually states the false information in the beginning of the article. And the false information is similar the queries input by the end user. The Transformers library [10] is used for working with the model, requiring few lines of code to use. How the BERT model can be used for reranking is mentioned in Section 2.3.1.

Question Answering

For the question answering the fine-tuned model released by the BERT team is used [9], which was fine-tuned on SQuAD [14].The flow created based on their explanation is shown in Figure 4.10. How the BERT model can be used for question answering is mentioned in Section 2.3.1.

As opposed to using the first passage, the last passage is used for the question answering. This is done for two reasons, the fact-checking part of the article is usually contained within the last part of the article. And to partly avoid an issue where the model uses the part of the article stating the claim as an answer. An article could be small enough to for the whole article to fit as a single passage. But for the articles longer than the maximum token size of 512 tokens [9]. The last passage is a better choice. It should be mentioned that Devlin et al. [9] also uses softmax before the output. In the implementation the logits are used, as this is the output from the BERT for question answering class from the Transformer library [42]. As softmax is not strictly needed it was not added.

4.4.4 Chitchat Skill

The chitchat skill uses the model created by Tang et al. [17], which is explained in Section 2.3.1. And the code released by the authors [43] is used, but adapted for the program. The predefined persona sentences and the message history can then be used on the personaGPT model to generate a response [17]. The chat history must therefore be stored. To avoid tokenizing the whole chat history every turn it is stored in tokenized form. Meaning only the new input and response need to be tokenized on each turn. The predefined persona sentences are shown as follows:

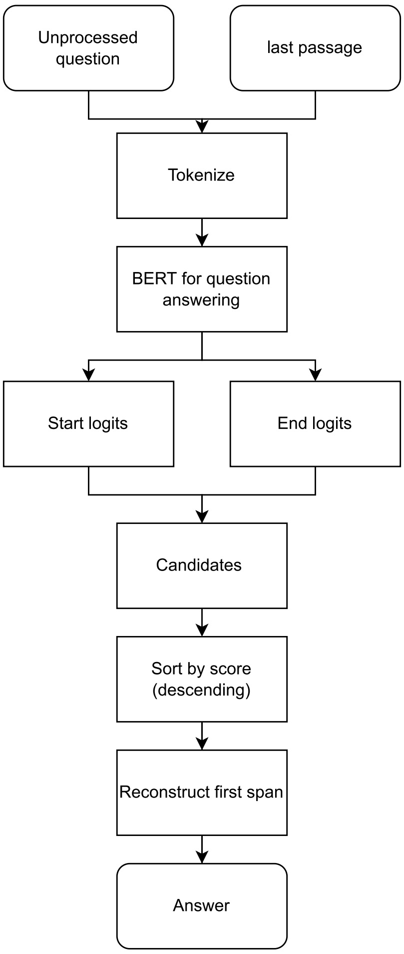

Figure 4.10: The question answering process. Based on the explanation by Devlin et al. [9].

- I am a fact-checking chatbot.
- I serve to deliver the truth.
- I will try to find existing fact-checks for you.
- I don't have a name.
- My features are random, elaborate, fact-check, related, chitchat, count and list.
- I like to read fact-checks.

4.4.5 Count Articles Skill

To get a count of articles on a topic the Count API for Elasticsearch [7] is used. The skill takes a topic as input and returns the count. If no topic is given as input a match all query is sent, meaning the count will be of the whole collection. If a topic is provided a Boolean query is used. The query uses a filter on the topic field in the collection, with the provided topic as term. If multiple topics are provided to the filter as a list of separate terms. This way the topic field must contain both terms, and not only one of the terms. The response message is predefined, with the amount of articles, and the topic found. The topic is told back to the end user, so they can verify the count is for the topic they asked about. Some examples of questions and responses from the skill is shown in Table 4.6

Question	Response
how many factchecks do you have about russia	There are 134 fact-checks about russia.
how many articles about biden	There are 352 fact-checks about biden.
how many factchecks do you have?	There are 42925 fact-checks in total.

Table 4.6: Examples of responses from the count articles skill, with the question asked.

4.5 User Feedback

The user feedback is not directly used in the Conversational AI. Therefore it is only initialized, and contain a function for inserting a new column. A function for extracting all the stored data to a comma-separated file is also made. This way the data can be loaded into a spreadsheet program or dataframe, to be worked upon. A single table is created in the database to store all the data, and the columns of the table are shown in Table 4.7.

Column name	Datatype	Description
Id	Text	Id for tracking whole conversations. In this case connection time is used
Time	Timestamp	Message store time, to track the ordering of turns
Question	Text	End users input to the conversational AI
Answer	Text	The response from the conversational AI, only the message
Intent	Text	The intent identified by the intent classifier
Evaluation	Integer	The rating provided by the end user for that turn
Version	Integer	The deployment version

Table 4.7: The columns of the table for storing feedback.

All data need for the feedback is stored in the session data as it is received or generated. That means the id is created and stored when the user first connects. The time is the current system time when the row is stored. The question is stored when it is received on the search endpoint. The answer and intention is stored before it is sent to the website. And the version is set as a variable before the application is deployed. When a rating is received on the feedback endpoint all the other data is retrieved from the session, and a query is executed to store the row in the database.

4.6 Security

Since the user interface is implemented as a website, security should be taken into consideration. There are different issues attached to deploying a web application which will not be discussed here, as this mostly comes down to correctly configuring the web server.

All the endpoints use GET as http request methods. All endpoints therefore only accept GET requests. The default endpoint does not accept any parameters. Most automated bots only target the default endpoint, but this does not stop a targeted attack. The user input is never responded back to the website. The topic change endpoint and feedback endpoint only executes if the input parameter is an integer, and the response is always the same. User input for the search endpoint is converted into a string. The data provided as a response from the conversational AI is filtered to only contain needed information. When storing the user's input for the feedback database, it is escaped, and therefore SQL code will not be executed. A general error message is shown to the end user if the conversational AI encountered an error, meaning the same error message is always shown.

Chapter 5

Experimental Evaluation

5.1 Experimental Setup and Data Set

Since the goal was to create a conversational AI, which could serve fact-checks, the evaluation will be on the performance of the AI. Another way to evaluate the AI would be on user satisfaction. But since user feedback of this type were not gathered, this will not be evaluated. Due to encountering several problems on deployment the conversational AI were not deployed to the general public in time. As a result of this the feedback collected is limited, and only provided by persons familiar with the project, or known by the author. Due to this, there might be some bias in the ratings provided.

The deployed version used to collect the feedback had the spelling correction, and coreference resolution removed. This was done as the provided instance for hosting did not have enough computational resources to support the model with all the components. A fallback method for the fact-checking skill was also added. If the skill did not generate a response the question was sent to the Factiverse AI [44], which could fact-check the question. The structure of the feedback gathered is explained in Section 3.3.5. To convey strengths and weaknesses in the conversational AI, some of the data from the feedback will be shown. A conversation between the author and the AI is also shown to show the capabilities of the AI.

5.2 Experimental Results

The average rating of each intent, and the overall average rating is shown in Figure 5.1. The two intent "list" and "related" are not shown as no feedback were received for these intents. The word intent is used instead of skill, as the intent decides which skill to use, and the intent may not be properly recognized.

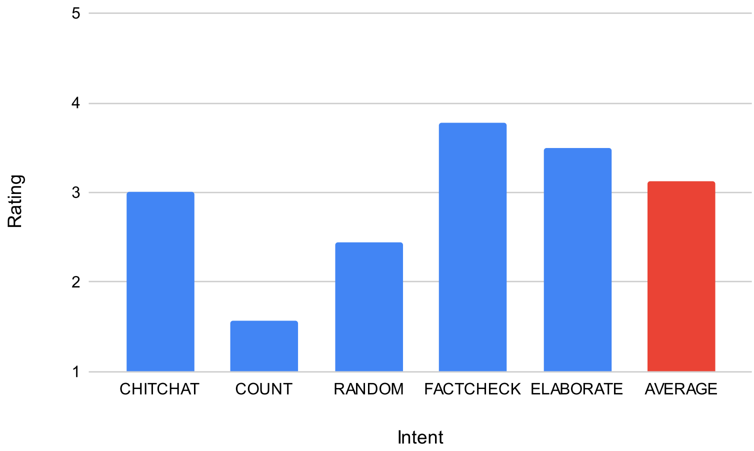

Figure 5.1: The average rating each skill received, and the overall average rating.

The number of times an intent was recognized is shown in Figure 5.2, and the total number of feedback received was 129. 26 of the 129 feedback were miss classified by the intent classifier. Giving it an accuracy of about 0.80, where 24 of them where incorrectly classified as "count", and two for "random". Four of the questions had an intent not present in the conversational AI, so by disregarding these the intent classifier gets an accuracy of 0.82.

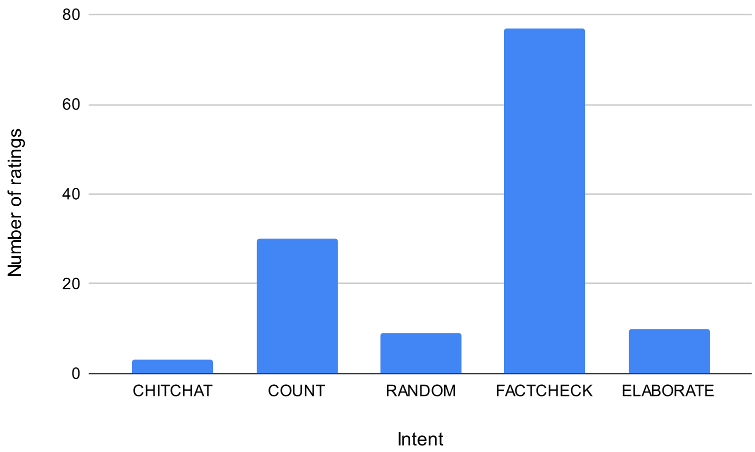

Figure 5.2: The number of feedback received for each skill.

Table 5.1 shows some selected sentences from the collected feedback. This will be used in Chapter 6 to argue the strengths and weaknesses. Table 5.2 shows a conversation between the author and the AI. This displays how a full conversation with the AI could look like.

#	Question	Response	Intent
1	Are there any fact checks about Monkeypox?	(response omitted)	RANDOM
2	Is Hitler dead?	hitler died by his own hand in his berlin bunker in the closing days of world war ii	FACTCHECK
3	Who is China's leader?	There are 133 fact-checks about china.	COUNT
4	is obama the president of USA?	barack obama is former president of the united states of america	FACTCHECK
5	Do EU support Ukraine in the war	following russia ' s invasion of ukraine	FACTCHECK
6	How many fact checks do you have on Putin?	There are 20 fact-checks about putin.	COUNT
7	How many fact checks do you have on Trump	There are 42881 fact-checks in total.	COUNT
8	Will Finland be approved as a NATO member?	finland ' s parliament still must vote on whether to join nato	FACTCHECK
9	did elon musk buy twitter?	elon musk was in the process of purchasing twitter	FACTCHECK
10	tell me more	In April 2022, the country music website SavingCountryMusic.com published an article stating that the billionaire had also purchased country music for $89 billion. While savingcountrymusic.com is not a satire website, the author of this article made it abundantly clear that this was a work of fiction. The article was also tagged with words "fake news." A number of similar jokes have been made on social media. In fact, Musk has made some facetious claims himself, joking that he was going to buy Coca-Cola so that he could restore the drinks original cocaine-filled recipe.	ELABORATE
11	was the moon landing fake?	never says in this video that the moon landing was fake	FACTCHECK
12	did congress want obama to have a third term?	congress moves to appoint obama a third term in 2016	FACTCHECK

Table 5.1: Some selected questions and responses.

Turn	Speaker	Message
1	User	how are you?
	AI	i'm fine, how are you?
2	User	im fine thx
	AI	good to hear, just checking my chats
3	User	does trump think the earth is flat?
	AI	trump never said this
4	User	is it flat?
	AI	if the earth was flat , you would not be able to take both routes to and from a single destination
5	User	can you tell me more?
	AI	Rapper B.o.B. tweeted a picture of the Earth with no discernable curve on the horizon. astrophysicist Neil deGrasse Tyson responded with a series of tweets. We asked astronomers what they would tell the rapper to convince him the Earth is round.
6	User	Awesome!
	AI	i am a factchecking chatbot that reports facts. i am not a rapper.
7	User	ok, bye then
	AI	goodbye to you too.

Table 5.2: Possible conversation with the conversational AI.

Chapter 6

Discussion

6.1 Introduction

This chapter will discuss and interpret the results from the feedback mentioned in Chapter 5. The goal of this discussion is to establish how well the conversational AI performs, and thus the success of the AI.

6.2 Ratings

Focusing on ratings in Figure 5.1 and number of feedback in Figure 5.2, the majority of interest is in the fact-checking skill. The chitchat, random and elaborate skills have few ratings making it hard to draw conclusions. The overall average rating is about three, where the highest scoring skill is the fact-check. The lowers scoring is the counting skill. The poor rating of the count skill is mainly explained by the miss classification of intent. Most questions classified with "count" as intent, had an intention of fact-check.

Why were most of the ratings for the fact-checking skill? This might have been because of the how the persons were asked to test the AI. But the other skills were also built to supplement the fact-checking skill. The interesting result here is the lack of interest in the other skills, but it is not possible to draw any conclusions on the basis of the feedback here. Since it was done over a short time span, with few participants.

6.3 Conveying the Purpose of the AI

An issue with the AI created is the conveying of its purpose. As many of the questions asked to the AI, are not fact-check related. This issue comes from the problem, most fact-checking articles originate from a social media post, or statement from a public person. Therefore the fact-checking articles are specific in content. A few examples of this are shown in Figure 6.1.

Claim: Photo shows Modi touching a woman's foot with his chin.
Claim: The word "shit" comes from an acronym for "Ship High in Transit."
Claim: Media was not "silent" about the NXVIM case.

Figure 6.1: A few examples of claims in articles.

From the feedback data, about 64 percent of the fact-check went to the Factiverse AI [44], meaning the fallback method. Some of these could be explained by not finding the right article, or the MRC not finding a response. But the majority of questions seem to be due to non fact-check related questions.

6.4 Broker Components

Some of the skills, only rely on the components of the broker working as intended, such as random, list, and count. A response will therefore be correct as long as the broker works.

6.4.1 Intent classifier

The accuracy of the intent classifier, achieved on the feedback, is worse than the accuracy based on the testing data. This is expected as the testing data is small, and no verification set were created. The data revealed an issue where "fact-checking" intents were classified as the "count" intent. There might be an issue in the training data provided for the count intent, or just the lack of training data. The feedback mostly focus on a single intent, making it hard to establish a good accuracy measure. Row one and two in Table 5.1 shows examples of miss classification by the intent classifier.

6.4.2 NER

The feedback data shows the issue mentioned in Section 4.3.2 on the NER model used, the model seems to be case sensitive. Examples of the NER working and not working are shown in row six and seven of Table 5.1. The recall should be 0.85 [37], but the feedback gives a recall of 0.73, this was calculated from all feedback classified as the "count" intent, as the entity is responded back. Many of the questions contain multiple entities, but the "count" will only show one of them. This means the recall could be lower 0.73. It could also be that the entities encountered are not similar to the data it was trained on. In both of the cases further training would solve the problem.

6.5 Fact-check skill

The fact-checking skill is the most complex of the skills, and the core of the conversational AI. This section will discuss the performance of this skill. The rows from Table 5.1 will be used as examples, meaning all rows referenced in this section refers to rows in Table 5.1.

Row two shows an example of a response answering the question asked. It could be argued that the response could also be a simple "yes". But as stated in Section 2.3.1, the model can not generate new answers only use answers from the passage supplied. On the other hand the response give a detailed explanation to the answer, which would be preferred as it's working with fact-checks.

The rows four and nine, shows the IR manages to find articles containing the answer when the article is about something else, and briefly mentions the answer to the question. The summary for the article used in row nine is shown in row ten to show what the article is about. It might not always provide a sensible response as shown in row five, but it is probable that the collection does not contain an answer to the question. Rows five and eight, also brings forward an issue that is not accounted for in the implementation. Some questions are time frame dependent, meaning the answer to the question could be updated in a new article. As the collection contains fact-checking articles dating way back, the information it responds with could be outdated. Row eleven is another example of a unnatural response, which technically answers the question, but not in the expected manner. The MRC model is not fine-tuned for fact-checking articles, an issue with this is that the model can not distinguish between phrases stating the false information in the article, and the actual fact-check answer. As a result of this the answer it finds can be false, as shown in row twelve.

Although there are many cases where the MRC model will not provide an answer, or the wrong answer to a question, the user interface is designed to supply the information necessary for the end user to get the right information. This is through linking to the article used, and a Tooltip with information about the article and the verdict. Although not optimal it is a solution.

Chapter 7

Conclusion and Future Directions

7.1 Conclusion

The goal of this project was to create a conversational AI that can server fact-checks. To conclude if this goal was achieved the subtasks mentioned in Section 1.2 will first be addressed, which in turn as a whole answers the overall goal.

The first subtask was to provide answers to fact-checking questions based on the provided collection. This task was addressed by the fact-checking skill. The article most likely to contain the answer is found. This is done in a multistep process, preprocessing of query, initial retrieval, and reranking. An answer to the query is generated from the article through the use of a MRC model for question answering. There are multiple issues resulting in an incorrect or non-answering reply. These issues were addressed in a non-optimal way through the use of the user interface providing extra information.

Subtask number two was to serve the fact-checking article in multiple ways. This was addressed by the creation of multiple skills which serves a related, random, or list of articles. A broker system was introduced to handle the overall conversation, so the appropriate skill could be used to generate the response.

An intuitive user interface for the conversational AI were created. This was through a website, where a chat window is used to converse with the AI. The extra information provided by the AI were presented through a Tooltip and a link. This way of interacting is common, thus making the AI intuitive to interact with. A slideshow also introduces the end user to the capabilities of the AI. An issue which became apparent, and is not address with the user interface is the introduction to the concept of a fact-check. The end user is not properly introduced to what a fact-checking article is, and in many cases will start asking non fact-checking related questions.

To address the task of multiturn questions, a coreference resolver was introduced. This resolver can only handle the previous turn question, so the issue is not fully addressed. The elaborate skill was also introduced, enabling possibility to get more information about a response. This response is a pregenerated summary of the article and is therefore not a direct elaboration of the previous response.

To make the conversation more human like, a chitchat skill was introduced. This makes AI able to chitchat in between the turns of the other skills. Which gives the sense of the conversational AI being more human like, as it also converse with the end user, thus responding to every message by the end user.

All though there are multiple issues with the conversational AI created, it is a functioning AI capable of serving fact-checks. It is capable of answering a fact-checking question, but extra information must be displayed as the answer alone could be wrong. The AI can handle multi-turn conversation though coreference resolution and the elaborate skill, but it is limited. Through the use of a chitchat agent the AI can also small talk, making the conversation seem more human like. With this said, the goal of this thesis was achieved. But there are many issues that should be solved, and many improvements that could be made.

7.2 Future Directions

The conversational AI created is just the beginning and there are several issues that should be addressed. There are also many possibilities for expansion on the model to make it perform better.

The issues covered in Chapter 6 should be addressed. Where the issue of the responses by the fact-checking skill using the false information in the article as an answer, should be prioritized. This would remove the need for displaying extra information, making it simpler to add other user interfaces. The dataset for the intent classifier should be expanded, where the issue of the fact-checking intent being classified as the count intent is the largest issue. The issue of NER should also be addressed, either by changing the NER model, to one that is trained on lowercase text, or properly case the user input.

Lexical normalization could be used, in the preprocessing of the broker. This could increase the performance of the AI as most models used are mostly trained on more lexical data. A pointer to doing this would be to look into the work of Bucur et al. [45]. Another improvement would be to make the response to the questions a direct response. Most skills use a pre-made response and to make them seem more human like it would be better if the response took the question asked into account. The priority of topics can also be improved, through a multistep process. Rather than checking if one topic matches they should be scored based on how many topics match. If there were no matches it could then try filtering for one of the provided topics. With the current implementation, an article with one matching topic has the same probability of being chosen as an article where two topics match.

A browser extension for user interface would be interesting as it would be easy for an end user to open when encountering a claim on social media. This would make the AI more accessible as the AI could overlay a small portion of the web page and the end user would not have to change the tab or leave the page where the claim was encountered. Another issue is conveying the purpose of the conversational AI itself. It can not answer all questions, the questions should be related to fact-checks.

There do exist some fact-checking related datasets, which could be used to either further fine-tune some of the Transformer models used or to train new model when expanding the conversational AI. One could look into the works of Gupta and Srikumar [46], Wang [47], and Augenstein et al. [48] for this.

List of Figures

List of Tables

Appendix A

Instructions to Run System

For the program to work an Elasticsearch instance must be setup. The collection must contain the fields shown in Table 2.4. The credentials for accessing the Elasticsearch instance must then be added to the esConfig.yml file. The dependencies must be installed, more information on this is contained in a README.md file. Afterwards the system can be run as a Flask app.

Appendix B

Attachments

All code created as part of this thesis is included, this also includes the dataset created.

- Embedded: conversationalAI.7z

Bibliography

[1] 'cambridge analytica planted fake news'. *BBC*, 2018. URL https://www.bbc.com/news/av/world-43472347. Accessed: 2022-06-14.

[2] Jianfeng Gao, Michel Galley, Lihong Li, et al. Neural approaches to conversational ai. *Foundations and trends® in information retrieval*, 13(2-3):127–298, 2019.

[3] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[6] Krisztian Balog. Conversational ai from an information retrieval perspective: Remaining challenges and a case for user simulation. In *Proceedings of the 2nd International Conference on Design of Experimental Search & Information REtrieval Systems (DESIRES '21)*, pages 80–90, 2021.

[7] Elastic NV. What is elasticsearch?, n.d. URL https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html. Accessed: 2022-06-14.

[8] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[11] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.

[12] Philipp Reissel and Igli Manaj. Passage reranking multilingual bert[clockwise vertical arrows][earth globe europe-africa], 2021. URL https://huggingface.co/amberoad/bert-multilingual-passage-reranking-msmarco. Accessed: 2022-06-14.

[13] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

[14] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[15] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[16] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[17] Fengyi Tang, Lifan Zeng, Fei Wang, and Jiayu Zhou. Persona authentication through generative dialogue. *arXiv preprint arXiv:2110.12949*, 2021.

[18] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*, 2019.

[19] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*, 2018.

[20] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad andAbdelrahman Mohamed, Omer Levy andVeselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019. URL http://arxiv.org/abs/1910.13461.

[21] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

[22] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python. 2020. doi: 10.5281/zenodo.1212303.

[23] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. *arXiv preprint arXiv:1707.07045*, 2017.

[24] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

[25] Mikhail S Burtsev, Alexander V Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yuri Kuratov, Denis Kuznetsov, et al. Deeppavlov: Open-source library for dialogue systems. In *ACL (4)*, pages 122–127, 2018.

[26] DeepPavlov MIPT. Deeppavlov documentation, 2022. URL http://docs.deeppavlov.ai/en/master/. Accessed: 2022-06-14.

[27] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[28] DeepPavlov MIPT. Intentcatcher example, 2022. URL https://colab.research.google.com/drive/1l6Fhj3rEVup0N-n9Jy5z_iA3b1W53V6m?usp=sharing. Accessed: 2022-06-14.

[29] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@NIPS*, 2016. URL http://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf.

[30] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28, 2015.

[31] Eric Brill and Robert C Moore. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th annual meeting of the association for computational linguistics*, pages 286–293, 2000.

[32] Tom Bocklisch, Joey Faulkner, Nick Pawlowski, and Alan Nichol. Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*, 2017.

[33] Armin Ronacher. Flask, 2011. URL https://palletsprojects.com/p/flask/. Accessed: 2022-06-15.

[34] Bootstrap Core Team. Bootstap 5, 2021. URL https://getbootstrap.com/. Accessed: 2022-06-15.

[35] Moin Uddin. Botui documentation, 2017. URL https://docs.botui.org/. Accessed: 2022-06-14.

[36] Pascal van Kooten. contractions, 2016. URL https://github.com/kootenpv/contractions. Accessed: 2022-06-14.

[37] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. Spacy documentation on english models, 2020. URL https://spacy.io/models/en. Accessed: 2022-06-14.

[38] Richard D Hipp. SQLite, 2020. URL https://www.sqlite.org/index.html.

[39] Google. Google cloud, 2008. URL https://cloud.google.com/. Accessed: 2022-06-15.

[40] Team NLTK. Natural language toolkit. URL https://www.nltk.org/.

[41] Shane Connelly. Practical bm25 - part 2: The bm25 algorithm and its variables, 2018. URL https://www.elastic.co/blog/practical-bm25-part-2-the-bm25-algorithm-and-its-variables. Accessed: 2022-06-14.

[42] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers documentation on bert, 2019. URL https://huggingface.co/docs/transformers/model_doc/bert#transformers.BertForQuestionAnswering. Accessed: 2022-06-14.

[43] Fengyi Tang, Lifan Zeng, Fei Wang, and Jiayu Zhou. A conversational agent with many personalities (personagpt), 2021. URL https://huggingface.co/af1tang/personaGPT. Accessed: 2022-06-14.

[44] Factiverse, 2019. URL https://www.factiverse.no/. Accessed: 2022-06-15.

[45] Ana-Maria Bucur, Adrian Cosma, and Liviu P Dinu. Sequence-to-sequence lexical normalization with multilingual transformers. *arXiv preprint arXiv:2110.02869*, 2021.

[46] Ashim Gupta and Vivek Srikumar. X-fact: A new benchmark dataset for multilingual fact checking. *arXiv preprint arXiv:2106.09248*, 2021.

[47] William Yang Wang. " liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*, 2017.

[48] Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. Multifc: A real-world multi-domain dataset for evidence-based fact checking of claims. *arXiv preprint arXiv:1909.03242*, 2019.