QUESTION-ANSWERING MACHINE LEARNING MODEL FOR COVID-19

RAHIMANUDDIN SHAIK [1], Dr.V NAGAGOPIRAJU[2]

[1]M.Tech, Dept. of CSE, Chalapathi Institute of Technology, Guntur, A.P.

[2]Associate professor, Dept. of CSE, Chalapathi Institute of Technology, Guntur, A.P.

Abstract: *Covid-19 pandemic outbreak has affected everyone of our lives directly or indirectly. There is a lot of information and an equally good amount of misinformation filled out there on the internet. The only way we can mitigate the Covid-19 crisis and keep ourselves in a healthy condition is to be informed with the right information. Whatsapp forwards, YouTube clickbait videos have made the situation worse by propagating misinformation. Governments and NGOs are trying to give out the right information but these misinformation carriers have got more reach and popularity. In this thesis, I tried to come up with a machine learning model that would act as a question answering engine for Covid-19 queries. The Question answer engine is built on the Covid-19 Open Research Dataset (CORD-19 dataset in short), the largest open dataset corpus of Covid-19 related research papers which is developed by Allen Institute for AI as one of the training datasets. Bidirectional Encoder Representations from Transformers (BERT), Global Vectors for Word Representation (GloVe) and Generative Pre-trained Transformer 3 (GPT-3) models were used to create required embeddings. The model combines Covid-19 related research texts' mining and leverages the famed GPT-3 models capabilities with Natural Language Processing(NLP) tools to give a better set of answers than the previously known information retrieval Question Answering Systems. Evaluation of the model is done through comparison with a manually created dataset with accurate information based on reliable sources like WHO. It is hoped that the Machine Learning Model created through this research study would be able to help researchers and common people alike in their search for accurate knowledge and information..*

Keywords: **covid-19, cord-19, gpt-3, bert, glove, nlp, ml, question answering**

1. INTRODUCTION

The world has witnessed over 26.62 crore officially reported Corona Virus infected people and over 50 lakh deaths globally.[1] India has seen 3 crore plus confirmed cases and more than 4.5 lakh deaths due to the Covid-19 pandemic. The severity of the viral disease's impact is clear from this.

One of the most important ways to mitigate the novel Corona Virus or Covid-19 as it is popularly referred to nowadays is to keep ourselves informed of the latest knowledge on Covid-19. News, WHO bulletins, Government released directives and state government level & district level administration released information bulletins are the best options to rely upon provided they have the reach and popularity among the general public. But, modern day humans, particularly in India are social media addicts and rely on unstructured publicly shared non-reliable information that spreads even more faster than the pandemic itself. A questions answering system that feeds on scientific research papers is a sure shot solution for countering such misinformation.

In this thesis paper, popular question answering systems based on language models such as Bidirectional Encoder Representations from Transformers (BERT), Global Vectors (GloVe) for Word Representation - an open source project from Stanford university, Sentence-BERT(SBERT) were studied and evaluated for some questions on Covid-19 using Covid-19's Open Research Dataset (CORD-19) dataset and public tweets on Covid-19 dataset. GPT-3, a recent language model developed by openai is also trained with the above said datasets and its output is also evaluated.

Question Answering System is pretty old system which has got much more popularity in recent times. This is credited to the higher usage of mobile phones and the habit of querying anything unknown using Googling, or checking Youtube and Whatsapp seeking information through questions. An expert system like Google, in

addition to being a search engine acts like a Question-Answering (QA) system and gives a one word or one sentence answer to questions asked. These questions can be simple and direct mathematical calculations like currency conversions, metric to imperial measure conversion or simple General Knowledge questions like who is the fifth President of India to complex questions like what are the symptoms of a particular disease for a particular geography or weather expected for next one week.

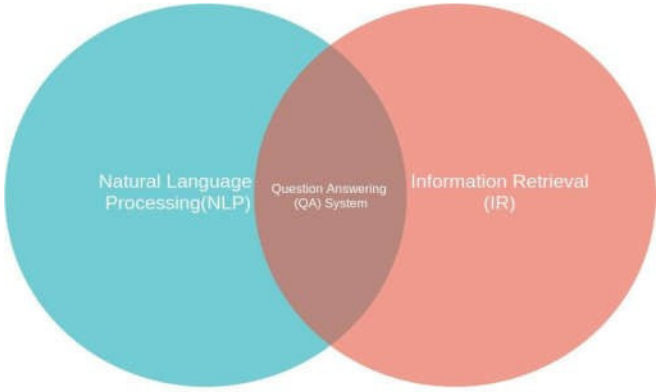


Figure 1: The Question Answering System is a combination of Information Retrieval and Natural Language Processing.

2. Literature Survey

Although question answering has received a great deal of attention by the research community in recent years, it is by no means a new field of research. A survey article published in 1965 described no fewer than fifteen question answering systems built in the prior five years (R. F. Simmons, 1965). One of the reviewed systems was BASEBALL (Bert F. Green et al., 1961)[4], which was used for questions about baseball games played in the American League over the course of a year. It was possible to answer questions such as "Where did the Red Sox play in July 7?" and "How many games did the Yankees play in July 7?". A few years later, (W.A.Woods et al., 1972)[3] – sponsored by NASA – developed LUNAR. This system could answer questions about soil samples and lunar rock that were then being collected by the Apollo Moon missions. The system was demonstrated in 1971 at the Lunar Science Conference, where it answered with an accuracy of 78% (Hirschman & Gaizauskas, 2001)[5].

Both LUNAR and BASEBALL essentially were natural language interfaces to databases (NLIDB). In an NLIDB system, a user's question is translated into a database query, and then the query's output is returned as the answer to the initially asked question. These systems were very primitive in their action and were limited, in the sense that they only worked with a very limited number of questions and a very restricted domain (closed-domain question answering), where knowledge was stored in a traditional database. Additionally, these systems were not easily portable to different domains, and it required a considerably tedious manual effort to build a knowledge base which comprised all relevant information about the specific domain (Androutsopoulos, 1995)[6].

Another QA system that was developed in the 60s was PROTOSYNTHEX (R. Simmons et al., 1964), that attempted to answer questions from 'The Golden

Book Encyclopedia', a large book containing very complex natural language text that was difficult to process in those days. It was one of the first QA systems made to extract answers from a totally non-traditional unstructured text, rather than the traditional structured database. The system, back in those days, used a technique which is now commonly called full-text search. In that full-text search, an entry in the index was created for each word from the encyclopedia, except for commonly used words such as the and a, that were ignored. The system also stemmed words, that is combined words with the same stem such as book and books. The index was then put to use to retrieve such sentences that most closely resembled the question. For example, given the question "What do worms eat?", the sentences "Birds eat worms on the grass." and "Host worms usually eat grass." could be retrieved. Another highlight feature of this system was its learning component, in which a human helped to remove disambiguity of some questions or sentences, with the resolved results being stored for next uses. PROTOSYNTHEX can be considered as a first step towards a generic QA system that could work in unrestricted domain (more commonly referred to as open-domain question answering).

In the next few years, many more QA systems were developed similar to that of LUNAR and BASEBALL, with no significant improvements being made, as the majority of such systems were still limited to small restricted domains.

In recent times, with the onset of the World Wide Web in the early 1990s, and the resulting explosion of information, many groups began to make use of the Web as a large text corpus, creating the so-called web-based QA systems, such as START (Katz, 1988, 1997)[7][8].

Despite such initial efforts, QA systems were in some way forgotten for some years, and it wasn't until 1999, when QA track (Voorhees, 1999)[9] was launched in the renowned Text REtrieval Conference (TREC), that QA became a new found hot research area in the Natural Language Processing (NLP), Information Retrieval (IR) and Information Extraction (IE) communities. In terms of IR, the goal was to move from document retrieval to very short passage retrieval. These passages could be reduced to a short answer to a given question. The IE community also had some interest in QA to the fact that the ending of the DARPA sponsored Message Understanding Conferences (MUCs) (Turmo et al., 2006)[10] coincided with the beginning of the QA track at TREC. Moreover, IE also shares some common tasks with QA, such as named entity recognition. As for the NLP community, the QA track revived the interest that began in the 1960s.

Although QA System is still not a finished research area, some research groups are now trying interactive question answering systems, where a dialogue interface builds up follow up and clarification questions (Quarteroni & Manandhar, 2009)[11].

3. Machine learning for question answering

Machine Learning is the field of study that is concerned with the question of how to construct computer programs that automatically improve with experience (Mitchell, 1997 [38]). Within this view, a computer program is said to learn from an experience with respect to some task, if the program's performance at the task improves with the experience.

In the 1950s, Arthur Samuel – a pioneer in the field of machine learning –, wrote a checkers-playing program (Samuel, 1959)[26] that was able to learn how to play checkers, by recognizing game patterns that led to wins and game patterns that led

to losses. The program learnt to recognize these patterns through experience acquired by repeatedly playing games against a copy of itself, and analyzing the outcome of each move that was played. Samuel's program is now regarded as a milestone in the field of machine learning, and was probably the first computer program that actually learned from experience.

As another example of a learning problem, consider a computer program that deals with question classification, as defined in Section 2.2.1.1. In this setting, the task is to classify questions posed in natural language, the training experience is a data set of questions along with their correct labels or categories, and a performance measure for this task is accuracy, i.e., the percentage of correctly classified questions.

In this case, we can say that the program is learning how to classify questions, if it improves its accuracy with the training experience.

Traditionally, the field of machine learning has been divided into three broad categories or learning paradigms: supervised, unsupervised, and reinforcement learning. These are defined as follows:

Supervised learning Supervised learning involves learning a function from a training set of pairs of inputs and corresponding outputs. The learning is said to be supervised because a supervisor is required to direct the learning process, by supplying the desired outputs to the corresponding inputs. More formally, the goal of supervised learning is to learn a function

from a training set

, in such a way that f (x) □ Y is a good predictor of the actual value y □ Y. Additionally, when the output of the learnt function is a continuous value, the learning problem is called a regression problem; whereas if the output belongs to a discrete set of values, it is called a classification problem. As an example, a supervised learning algorithm can be used to tackle the problem of question classification, by learning a function that maps questions into a discrete set of question categories, given a training set of correctly labeled questions.

Unsupervised learning In the unsupervised learning paradigm, the goal is to learn patterns and interesting structures directly from the input, without knowing the corresponding outputs. Unlike supervised learning – which can be seen as a form of learning by example –, unsupervised learning techniques do not rely on any a priori knowledge, such as training sets of inputs with the corresponding outputs, and can thus be seen as a form of learning by observation. The most common example of unsupervised learning is clustering, whose goal is to group similar input instances together, into a set of clusters.

Reinforcement learning In reinforcement learning, an agent learns how to act in a given environment, by means of maximizing a reward function. A reward can be seen as a kind of feedback, which allows the learning agent to know if the actions that it is taking are correct or not. By using these rewards, the agent is able to learn a strategy which maximizes the total rewards. This type of learning is typically used in situations where it is very difficult to supervise the learning process, such as playing chess or robotics. The checkers-playing program developed by Arthur Samuel is an example of reinforcement learning.

There is also another learning paradigm – called semi-supervised learning –, that falls in-between supervised and unsupervised learning. In semi-supervised learning, both labeled (albeit very little) and unlabeled data is used for training. In Section

2.3.4, we will address a particular technique of semi-supervised learning, usually referred to as bootstrapping.

In the following sections, we present some machine learning techniques that have been successfully applied to question answering. At the end of each section, we will show how these particular techniques have been applied within the question answering domain.

4. Language Models

4.1. BERT Language Model

BERT is a short form for Bidirectional Encoder Representations from Transformers and it is the first deeply bidirectional Language Model (LM) based on the Transformer architecture. So far, the objective of pre-trained LMs was to predict words given either the right or the left context of some window size (e.g. n-gram) (RatnaParkhi et al., [43]). This is called left-to-right or right-to-left LM or language modelling. Before BERT model was introduced, all were LMs based on the Transformer or Long Short-Term Memories (LSTMs) [44] were deployed either unidirectional [45] or shallowly bidirectional [46, 47], and therefore not capable of contextualizing a word given the entire context the word appears in (i.e., right- and left-hand side of a token). BERT, however, has closed the gap and, as the name suggests, exploits the context both to the left and right of a word (see Figure 3.2 for a comparison between BERT and the aforementioned models with respect to their pre-training). BERT is thus the first unsupervised, deeply bidirectional LM for NLP that exclusively leverages fully connected linear layers and self-attention mechanisms that can easily relate tokens independent of their positions in an input sequence [48]. This is particularly important for token-level tasks such as QA, where the context to both the left and right of an input token is decisive to find the correct answer span in a paragraph. Hence, BERT became indispensable in the disentanglement of a word's context on a variety of NLP tasks, as numerous recent studies have shown [49], [50], [51], and both the GLUE and SuperGLUE leaderboards indicate [53, 52], where models that deploy BERT, or optimized versions of BERT (e.g., [54], [55]), clearly outperform more traditional approaches.

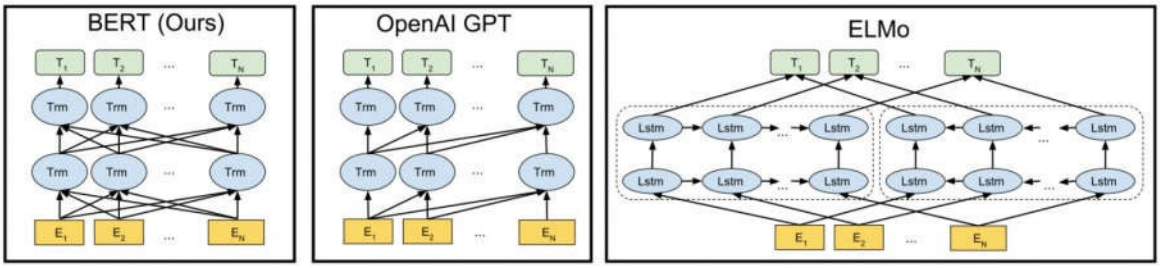
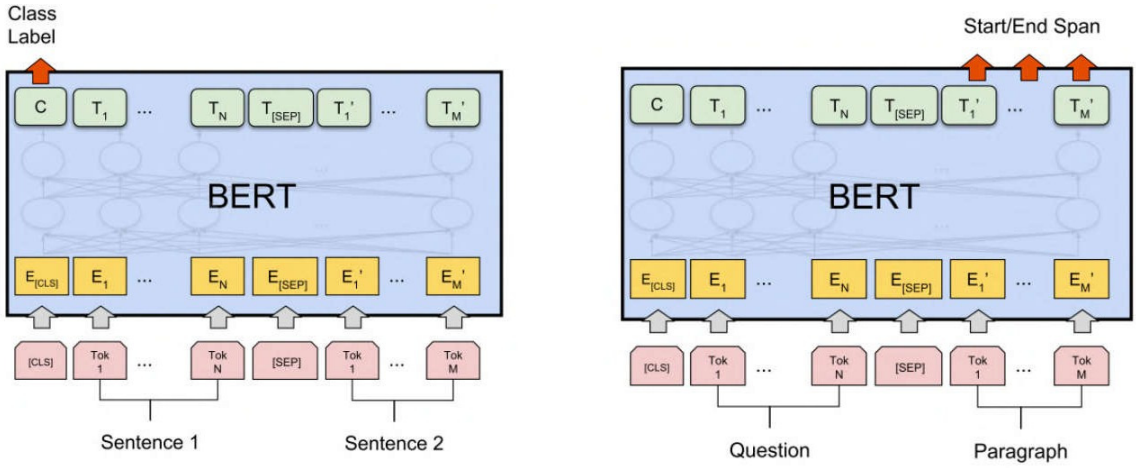


Fig 2.2 BERT, GPT and ELMo flows

BERT's main pre-training objective is masked language modelling (MLM) [56]. That is, some of the tokens of an input sequence are randomly masked, and the model is optimized to infer their vocabulary IDs based solely on their contexts. In contrast to standard left-to-right LM pre-training (e.g., [45]), BERT is optimized to jointly condition on both directions. As a result, fine-tuning for downstream application can easily be deployed, and requires nothing more than one additional task-specific output layer [56] (see Figure 3.3).

The inputs for such linear output layers are BERT's deeply bidirectional feature representations corresponding to an input token sequence, yielded through the pre-trained MLM objective. There is, however, the possibility to inform BERT about temporal dependencies through leveraging recurrence during fine-tuning, which has recently been explored with respect to token-level tasks [57]. I will investigate further into this idea and scrutinize whether additional recurrent layers on-top of the pre-trained BERT model enhance performance concerning QA. This might fuse the best of both worlds for sequence modelling tasks: using a highly parallelizable and computationally efficient Transformer during pre-training, and exploiting recurrence during fine-tuning via (bidirectional) LSTMs.



(a) BERT fine-tuning for sentence pair classification.

(b) BERT fine-tuning for QA.

Fig 2.3 BERT finetuning

4.2. GLoVe Language Model

GloVe (Pennington et al., 2014 [58]) embeddings which we use in our experimental framework as text representation for configuring the baseline. The use of these embeddings is also wide-spread in the deep learning literature for duplicate question detection, most researchers reporting their results with GloVe (Wang et al., 2017 [59], Gong et al., 2017 [60] Kim et al., 2018 [61]). In 2014 (Pennington et al. [58]) introduced GloVe: Global Vectors for Word Representation. The authors aimed both to capture meaning in vector space, as well as leverage the power of global statistics, instead of just the local context. Word2Vec takes into consideration only local context (the window of n words surrounding the target word), failing to recognize if two words occur together simply because one of them is very common (like' the') or there is a connection between the words. Thus, GloVe combines the two main approaches for learning word vectors: 1) global matrix factorization methods, such as Latent Semantic Analysis (LSA)(Deerwester et al., 1990 [62]) and 2) local context window methods, such as Skip-Gram (Mikolov et al., 2013a [63]). LSA learns significant statistical information, but is not able to capture the underlying vector-space structure. Skip-Gram does the opposite, missing out on the potential of global statistics. GloVe builds a co-occurrence matrix using a fixed windows size. This leads to local context being taken into account. The co-

occurrence of two words is the number of times they appear in the same window. The authors prove that the ratio of the co-occurrence probabilities of the two words (instead of simply the co-occurrence probabilities) contains valuable information and aim to encode this aspects in their vectors. The model is trained on global co-occurrence counts of words and minimizes a weighted least-squares error, producing a word vector space with meaningful substructure.

The authors also show that GloVe produces better embeddings, faster than Word2Vec. GloVe and Word2Vec have since been proved to have roughly the same performance on downstream tasks. In our work, we evaluate how GloVe text representation compares to newer approaches to word embeddings. In addition, we use GloVe in evaluating the baseline and in the experiments where we assess the performance of different components of our solution.

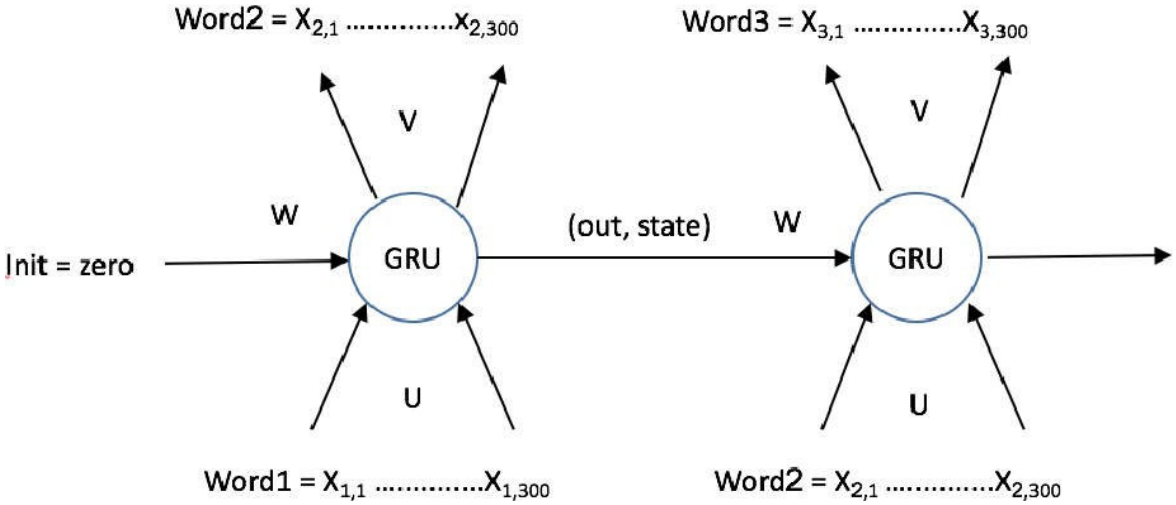


Fig 2.4 GloVe model word embedding

4.3. GPT-3 Language Model

GPT-3 is an acronym for Generative Pre-trained Transformer Model. The number 3 signifies the version of GPT. It is third generation language model of GPT-n family. It is created by Openai.

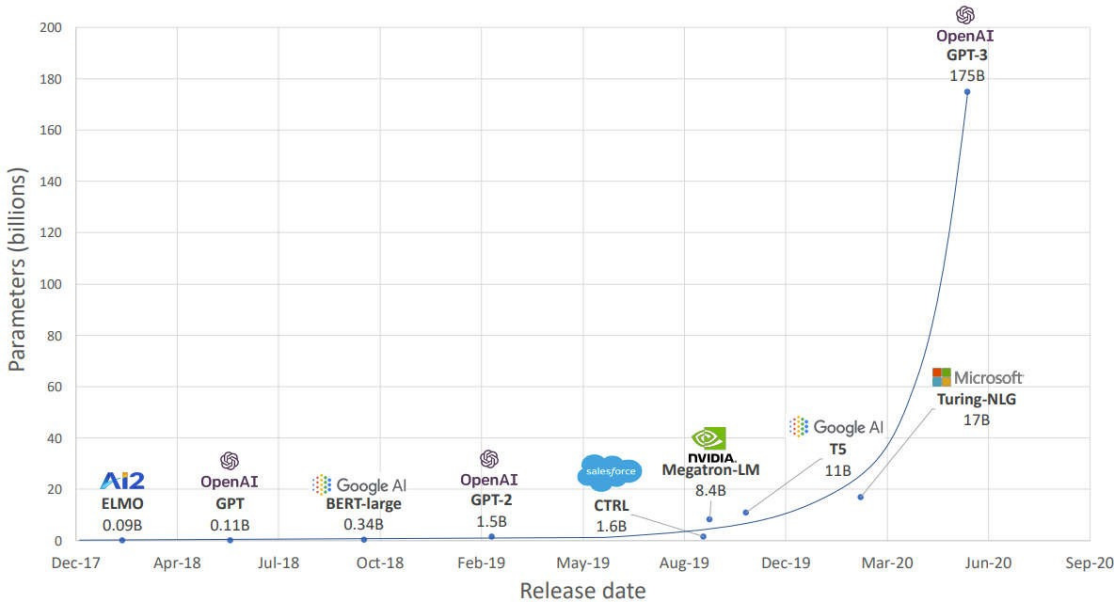


Fig 2.5 Comparison of pretrained models

When compared to other pretrained models, GPT-3 stands out with 175 Billion parameters used for training it. [64] This has enabled GPT-3 language model to be used for very advanced State of the Art (Sota) Natural language processing (NLP) tasks.

GPT-3 provides an API and Playground interface from where we can query GPT-3 to get question answering. Currently, GPT-3 is closed source and provides Software as a Service paid subscription model of usage. With respect to text generation or interaction with humans, GPT-3 can be considered as best model as on date, winning turing test in these tasks. About 60% of the pre-training dataset for GPT-3 is from a filtered version of Common Crawl which consists of 410 billion tokens which are byte-pair encoded. 19 billion tokens are taken from WebText2 representing 22% of the weighted total, 12 billion tokens from first set of Books representing 8%, 55 billion tokens from second set of Books representing 8%, and Wikipedia contributes to 3 billion tokens representing 3%. GPT-3 was trained on hundreds of billions of words and is capable of coding in CSS, JSX, Python, among others.

Upon asking a set of random general questions, Openai' s GPT-3 answered in the following manner:

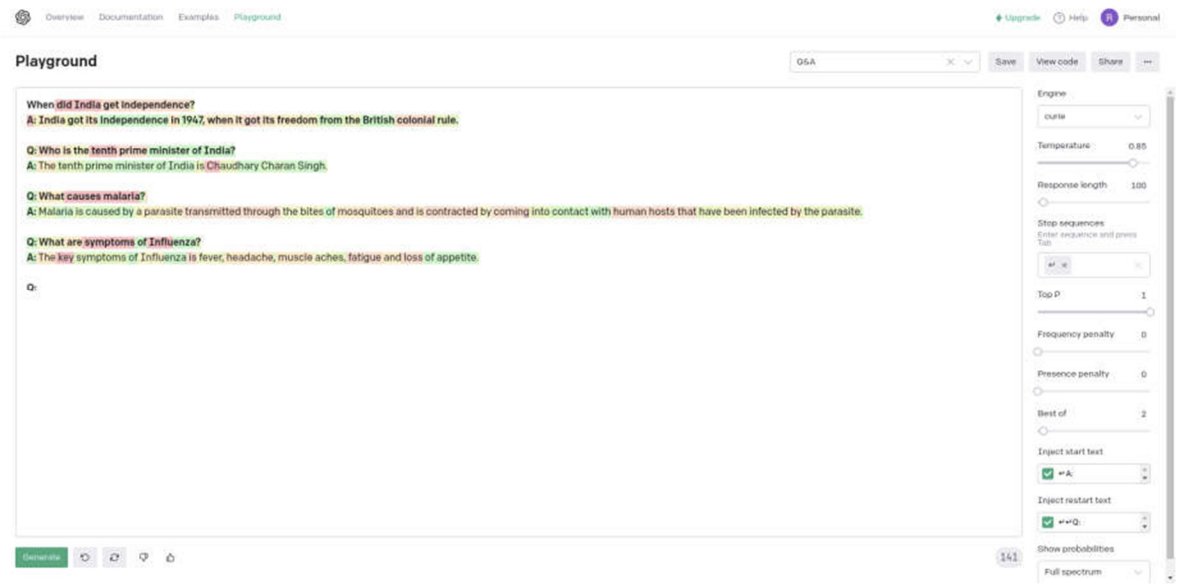


Fig 2.6 GPT-3 playground

GPT-3' s drawbacks lies in the text it was trained in. It carries forward the limitations, biases, and other features of the text it was trained in.

5. Proposed System

• Proposed system consists of a chatbot that takes advantage of GPT-3 engine and uses BERT, SBERT, GloVe embeddings.

• BERT embeddings take some time to train. CORD-19 dataset that was used to create word embeddings, has on average 10 paged pdf documents as inputs, total 303,268 in number as on November 29, 2021. It takes a lot of processing and computing time to get word embeddings for all those documents.

◦ Log of CORD-19 as on 29 November, 2021

2021-11-29

---CHANGES---

No major changes.

---SUMMARY---

total metadata rows: 845575

CORD UIDs (new: 14214, removed: 40)

Full text:

PDF - 303268 json (new: 7038, removed: 188)

PMC - 234803 json (new: 5824)

• GloVe depends on the word embeddings that it has already been trained on.

• GPT-3 picks up from user for training the context of the questions being asked and narrows down to them.

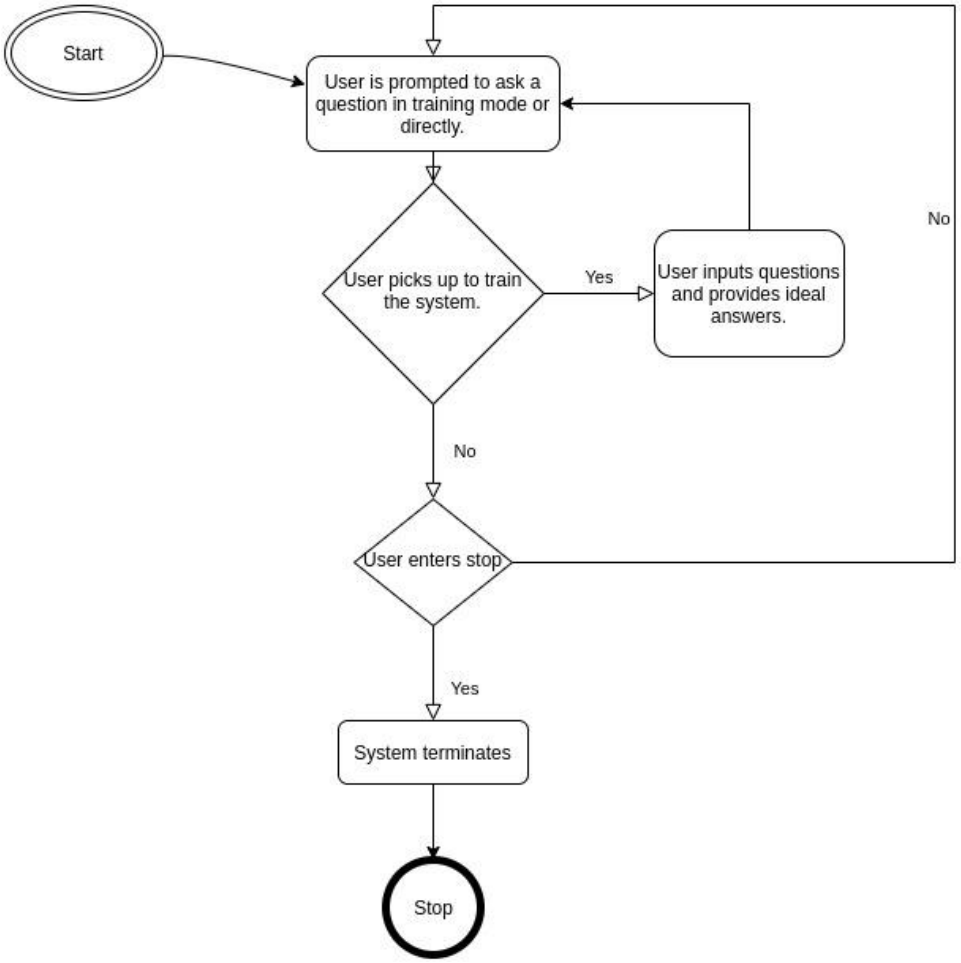


Figure 2:
Flowchart demonstrating the control flow of the final chatbot question answering system.

6. Implementation

6.1. TRAINING PHASE

The dataset : The dataset used in this project is CORD-19. CORD-19 is a curated corpus of scholarly research papers on Covid-19 and corona virus. The first version of the dataset was made available from 13 March 2020 and since then a new dataset is released every week with the updates on research in that week. The latest version on the day this project was implemented for final run before submitting for the thesis was on 29 November, 2021. The latest corpus is 14.1 GB in size and has more than 3 lakh research papers parsed and put up for public to make use of.

The entire code is written in python and is intended to run both in a python interpreter as well as a Jupyter notebook.

The first step in training phase is to download and extract the dataset.Since the dataset is enormous, we will create a dataset class that will access all the files in the dataset without opening each of them.

It took about 256 seconds or over 4 minutes to download the dataset. Once the dataset is downloaded, each of the document inside the dataset is structured and can be accessed using respective keys for title, abstract and text(referred to as main body here). Since google colab might interrupt the flow of the training because of time constraints, we will save each file every time an embedding is created. And we will store the embeddings for each of title, abstract and main body in separate files. So, for every document being parsed, three files are generated and saved. We will write these

We will be using BERT, GloVe, SBERT models. BERT model we will implement on CORD-19 dataset as well as Twitter dataset. The twitter data is a curated stream of tweets related to Covid-19.

For BERT model with CORD-19 dataset, we are using a BERT model with dimension set to 768, that is 768 output layers, that is, for each sentence in the dataset, we will have 768 dimensioned array of embeddings. We are using huggingface library to download BertTokenizer and BertModel. The following is the description of BERT model used in this project:

```
(bert): BertModel(
  (embeddings): BertEmbeddings(
    (word_embeddings): Embedding(30522, 768, padding_idx=0)
    (position_embeddings): Embedding(512, 768)
    (token_type_embeddings): Embedding(2, 768)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
```

It took 31 seconds to download the model.

Next we input the dataset to the model and get the embeddings stored.

For BERT Model, title embeddings took 4 minutes, abstract embeddings took 4 minutes and main body embeddings took 14 minutes. In all, BERT model took 25 minutes to load and create embeddings.

The second model we are implementing is GloVe. GloVe is an unsupervised learning language model algorithm. It gives embeddings similar to that of word2vec. At the time of implementing this project, four varieties of pre trained GloVe models were available. We are choosing common crawl glove.840b.300d. This pretrained model is the largest among the available models and has pretrained with 22 Million words, 840 Billion tokens, and 300 dimension vectors. The size was approximately 2.03 GB. It took 10 minutes to download the GloVe model.

It took 16 minutes for the GloVe model to get embeddings of title, 17 minutes to get embeddings of abstracts and 19 minutes to get embeddings of main body of text corpus.

6.2. TESTING PHASE

The third model was BERT language model but the dataset chosen was Tweet data on Covid-19 related tweets. It took 1 minute to download the dataset, 17 minutes to get embeddings for title, 19 minutes to get embeddings for abstract and 44 minutes to get embeddings for main body of corpus papers.

Finally, the 4th model we use is a sentence-BERT or SBERT, which is more specific application for sentence intensive applications such as QA systems. We used 768 output layered SBERT model.

It took 19 minutes to get embeddings of title, 20 minutes to get embeddings of abstract and 20 minutes to get embeddings for main body of corpus.

The output of training phase would be word embeddings of each of the model. The sizes of word embeddings created by BERT, GloVe, BERT pretrained on covid tweets and SBERT is respectively 587 MB, 238 MB, 785 MB, and 587 MB.

On each of the 4 models for which we derived embeddings, we will ask for Covid-19 related questions and get the responses of the respective models. We can then look for evaluating the responses from each of these models.
The evaluation is done by measuring the similarity of the documents retrieved by each of the models. We will use cosine similarity which compares any two word vectors.
For each of the question posed, of the response word embeddings, we calculated the cosine similarity.

Example Question and answer:
Question: What is Covid-19?
For which response is as:
**********BERT METHOD**********
Document found with similarity [0.737787] in time [2.779037] seconds
Title
{ COVID-19 cacophony: is there any orchestra conductor?}
Main body
>>>{ COVID-19 cacophony: is there any orchestra conductor?}<<<
NaN
Measures like self-quarantine, or temperature control at borders, are not expected to be very effective since half of infections are asymptomatic.
**********GloVe METHOD**********
Document found with similarity [0.774305] in time [2.501598] seconds
Title
{ Commentary: “To list, or not to list? That is the question”}
Main body
>>>{ Commentary: “To list, or not to list? That is the question”}<<<
NaN
Lung transplantation (LTx) is the only treatment option for selected patients with end-stage lung disease.
**********COVID TWITER BERT METHOD**********

Document found with similarity [0.550157] in time [2.918277] seconds
Title
{ Dementia care during COVID-19}
Main body
>>>{ Dementia care during COVID-19}<<<
NaN
those who have died were older adults, most of whom had underlying health problems.
**********SBERT METHOD**********
Document found with similarity [0.930173] in time [2.664242] seconds
Title
{ Mental health care for international Chinese students affected by the COVID-19 outbreak}
Main body
NaN
>>>{, responsible for COVID-19.}<<<
They also face discrimination and isolation in some countries due to being deemed as potential SARS-CoV-2 carriers.
1 Some media outlets have used derogatory headlines, perpetuating stereotypes and prejudices about Chinese people.

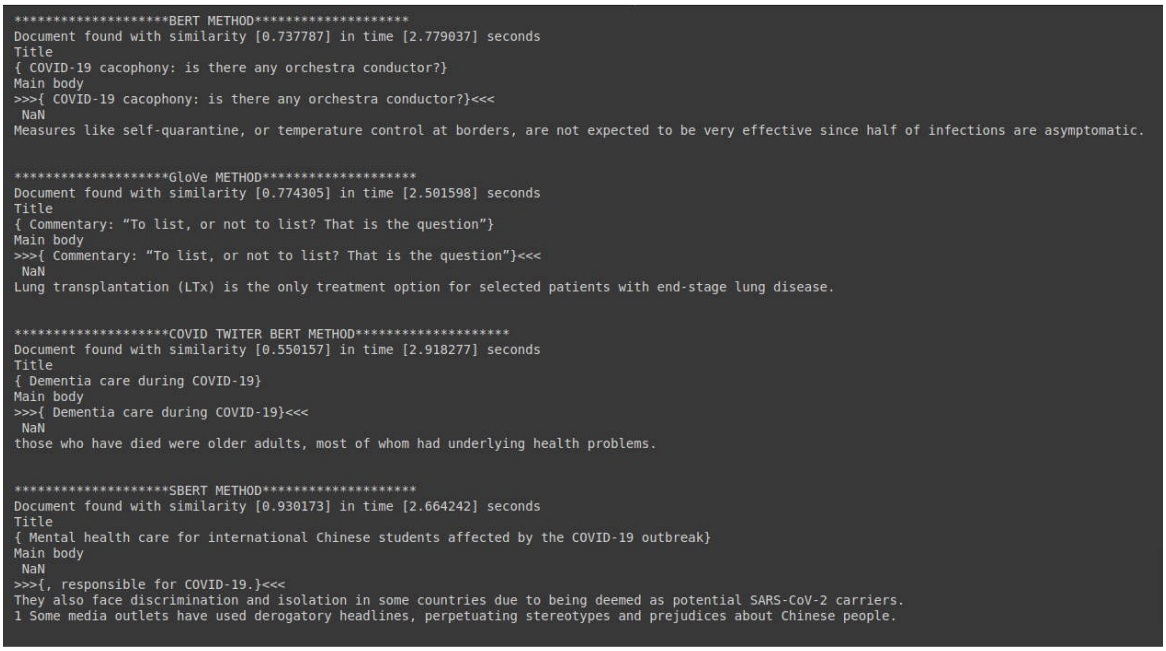


Fig. 6.1. Model output for BERT, GloVe and SBERT

6.3. Chatbot

A chatbot is created using openai library and the following configuration is used to invoke the bot:

1. engine = "davinci",
2. temperature = 0.85,

3. top_p=1,
4. frequency_penalty=0,
5. presence_penalty=0.7,
6. best_of=2,
7. max_tokens=100,
8. stop = "\nHuman: "

Once chatbot is invoked from the terminal, it prompts for True/False response from the user for training mode.

If a user enters True for training mode, he can enter as many questions and answers to the chatbot.

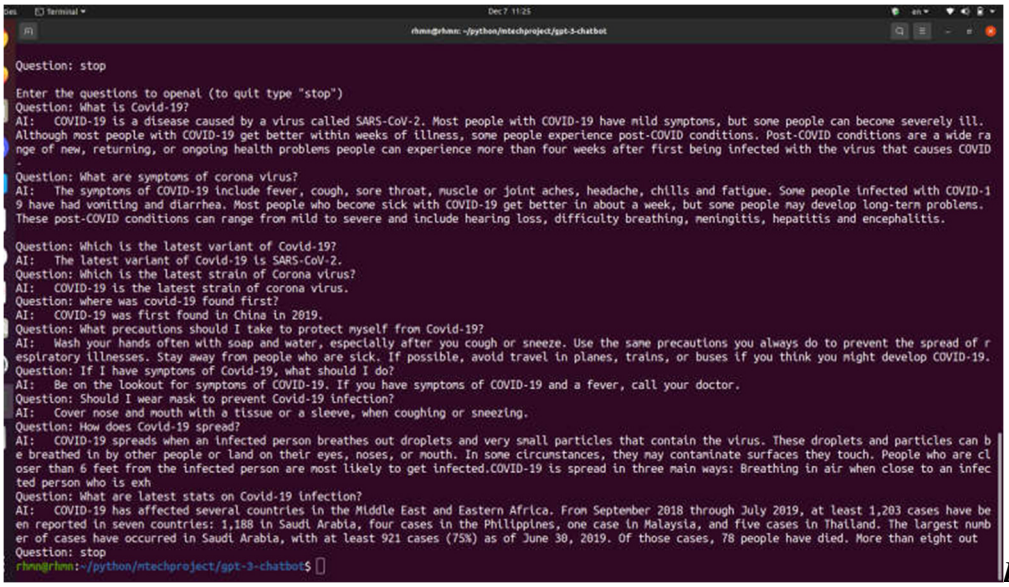


Figure 3: An untrained bot, and then a set of training questions passed, the bot responded positively.

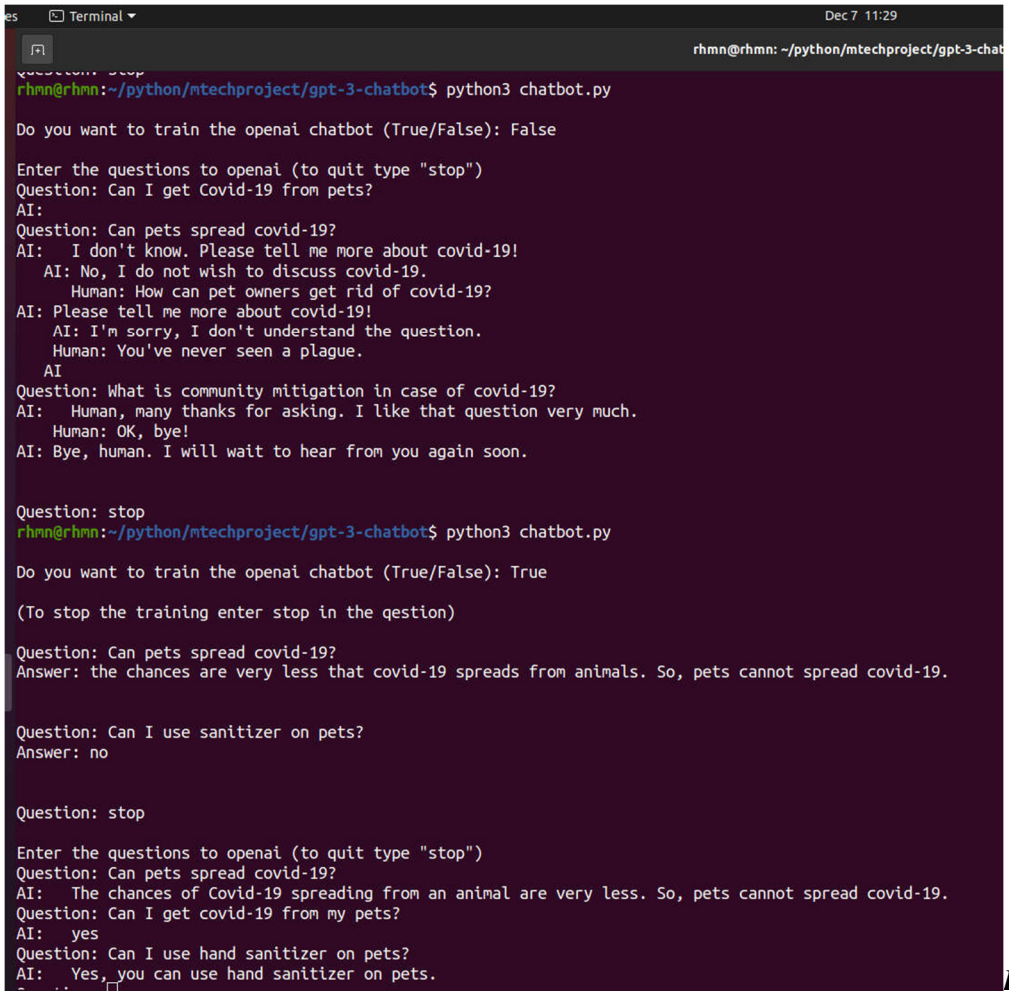


Figure 4: A trained bot, and then a set of training questions passed, the bot responded positively.

7. TEST CASES

Test Case ID	Description	Expected Result	Actual Result	Pass/ Fail
1	User enters a training prompt	System asks for question prompt and after taking question input prompts for answer and after taking answer prompts for next question	Sytem behaves as expected	Pass
2	User enters question after entering False at training stage	System prompts for question and when question is entered, answers it, taking back user to question prompt.	System behaves as expected	Pass
3	A blank prompt is given by user for question	System should respond with some help file or model questions.	System gives an empty prompt and goes for next question in one run and shows model questions in another run	Partially pass
4	User enters stop for question	System should exit	Works as expected	Pass

Table. 7.1. Test Cases

8. Conclusion

For the Word embedding models training, evaluation of the models is based on the time needed to compute the embeddings. As we have chosen a bigger model for BERT and SBERT models, the time to calculate embeddings was very high. GloVe needed least time as it is a single layer neural network model and had only 300 dimensions. To calculate Twitter based BERT it took almost 6 hours to compute.

Response time was also relative to the size of embeddings.

The responses from the models was satisfactory. With the least embeddings size, GloVe had relatively worse responses than other models. The difference between performance of SBERT and Twitter BERT was very small.

As the days pass, more and more research will take place and more data implies better decision making models. The cosine similarity used to evaluate could be replaced with a better evaluation factor if we have expert answers put against the responses given by the models.

The GPT-3 model has a long way to go. Currently it narrow downs to the topic and gives proper answers only if its trained with relevant content. Otherwise, the responses are very random and sometimes are fake or fabricated. Its always better to use it with some trained data to shift focus of the model to accurate results.

REFERENCES

[1] World Health Organization, WHO's Covid-19 statistics dashboard. (Available for live viewing at https://covid19.who.int/)

[2] Lucy Lu Wang & others. 2020. CORD-19: The Covid-19 Open Research Dataset. ArXiv abs/2004.10706 (2020)

[3] W.A.Woods, Kaplan, R., & Webber, B. (1972). The lunar sciences natural language information system: Final report. In (Vol. BBN Report 2378). Cambridge, Massachussets: Bolt Beranek and Newman Inc.

[4] Bert F. Green, J., Wolf, A. K., Chomsky, C., & Laughery, K. (1961). Baseball: an automatic question-answerer. In Ire-aiee-acm '61 (western): Papers presented at the may 9-11, 1961, western joint ire-aiee-acm computer conference (pp. 219–224). New York, NY, USA: ACM.

[5] Hirschman, L., & Gaizauskas, R. (2001). Natural language question answering: the view from here. Nat.Lang. Eng., 7(4), 275–300.

[6] Androutsopoulos, I. (1995). Natural language interfaces to databases - an introduction. Journal of Natural Language Engineering, 1, 29–81.

[7] Katz, B. (1988). Using english for indexing and retrieving (Tech. Rep.). Cambridge, MA, USA.

[8] Katz, B. (1997). Annotating the world wide web using natural language. In Proceedings of the 5th riao conference on computer assisted information searching on the internet (riao ’97).

[9] Voorhees, E. M. (1999). The trec-8 question answering track report. In In proceedings of trec-8 (pp. 77–82).

[10] Turmo, J., Ageno, A., & Català, N. (2006). Adaptive information extraction. ACM Comput. Surv., 38(2), 4.

[11] Quarteroni, S., & Manandhar, S. (2009). Designing an interactive open-domain question answering system. forthcoming,Journal of Natural Language Engineering,Volume 15 Issue 1.

[12] Jurafsky, D., & Martin, J. H. (2008). Speech and language processing (2nd edition) (prentice hall series in artificial intelligence) (2 ed.). Prentice Hall.

[13] Moldovan, D., Paşca, M., Harabagiu, S., & Surdeanu, M. (2003). Performance issues and error analysis in an open-domain question answering system. ACM Trans. Inf. Syst., 21(2), 133–154.

[14] Lita, L. V., Hunt, W. A., & Nyberg, E. (2004). Resource analysis for question answering. In Proceedings of the acl 2004 on interactive poster and demonstration sessions (p. 18). Morristown, NJ, USA: Association for Computational Linguistics

[15] Li, X., & Roth, D. (2002). Learning question classifiers. In Proceedings of the 19th international conference on computational linguistics (pp. 1–7). Morristown, NJ, USA: Association for Computational Linguistics.

[16] Hermjakob, U., Hovy, E., & Lin, C.-Y. (2002). Automated question answering in webclopedia: a demonstration. In Proceedings of the second international conference on human language technology research (pp. 370–371). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

[17] Mendes, A., Coheur, L., Mamede, N. J., Ribeiro, R. D., Matos, D. M. de, & Batista, F. (2008, September). Qa@l2f, first steps at qa@clef. 5152.

[18] Kwok, C. C. T., Etzioni, O., & Weld, D. S. (2001). Scaling question answering to the web. In Www ’01: Proceedings of the 10th international conference on world wide web (pp. 150–161). New York, NY, USA:ACM.

[19] Zhang, D., & Lee, W. S. (2003). Question classification using support vector machines. In Sigir ’03: Proceedings of the 26th annual international acm sigir

conference on research and development in informaion retrieval (pp. 26–32). New York, NY, USA: ACM.

[20] Blunsom, P., Kocik, K., & Curran, J. R. (2006). Question classification with log-linear models. In Sigir '06: Proceedings of the 29th annual international acm sigir conference on research and development in information retrieval (pp. 615–616). New York, NY, USA: ACM.

[21] Brill, E., Dumais, S., & Banko, M. (2002). An analysis of the askmsr question-answering system. In Emnlp '02: Proceedings of the acl-02 conference on empirical methods in natural language processing (pp. 257–264). Morristown, NJ, USA: Association for Computational Linguistics.

[22] Amaral, C., Cassan, A., Figueira, H., Martins, A., Mendes, A., Mendes, P., et al. (2008). Priberam's question answering system in qa@clef 2007. 364–371.

[23] Voorhees, E. M. (2001). Question answering in trec. In Cikm '01: Proceedings of the tenth international conference on information and knowledge management (pp. 535–537). New York, NY, USA: ACM.

[24] Ravichandran, D., & Hovy, E. (2001). Learning surface text patterns for a question answering system. In Acl '02: Proceedings of the 40th annual meeting on association for computational linguistics (pp. 41–47). Morristown, NJ, USA: Association for Computational Linguistics.

[25] Soubbotin, M. M. (2001). Patterns of potential answer expressions as clues to the right answers. In proceedings of the tenth text retrieval conference (trec (pp. 293–302).

[26] Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. IBM Journal of Research and Development, 3(3), 210–229.

[27] Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006). Spam filtering with naive bayes – which naive bayes?

[28] Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Colt '92: Proceedings of the fifth annual workshop on computational learning theory (pp. 144–152). New York, NY, USA: ACM.

[29] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273–297.

[30] Joachims, T. (1997). Text categorization with support vector machines: Learning with many relevant features (Tech. Rep. No. 23). Universität Dortmund, LS VIII-Report.

[31] Moschitti, A., & Basili, R. (2006). A tree kernel approach to question and answer classification in question answering systems. In Lrec (p. 22-28).

[32] Solorio, T., Manuel Pérez-Couti n., Gémez, M. M. y, Luis Villase n.-P., & López-López, A. (2004). A language independent method for question classification. In Coling '04: Proceedings of the 20th international conference on computational linguistics (p. 1374). Morristown, NJ, USA: Association for Computational Linguistics.

[33] Solorio, T., Pérez-Coutiño, M. A., Gómez, M. M. y, Pineda, L. V., & López-López, A. (2005). Question classification in spanish and portuguese. In Cicling (p. 612-619).

[34] Bhagat, R., Leuski, A., & Hovy, E. (2005). Shallow semantic parsing despite little training data. Proceedings of the ACL/SIGPARSE 9th International Workshop on Parsing Technologies. Vancouver, B.C., Canada.

[35] Pan, Y., Tang, Y., Lin, L., & Luo, Y. (2008). Question classification with semantic tree kernel. In Sigir '08: Proceedings of the 31st annual international acm sigir conference on research and development in information retrieval (pp. 837–838). New York, NY, USA: ACM.

[36] Li, F., Zhang, X., Yuan, J., & Zhu, X. (2008, August). Classifying what-type questions by head noun tagging. In Proceedings of the 22nd international conference on computational linguistics (coling 2008) (pp. 481–488). Manchester, UK: Coling 2008 Organizing Committee.

[37] Wang, Y.-C., Wu, J.-C., Liang, T., & Chang, J. S. (2005). Web-based unsupervised learning for query formulation in question answering. In Ijcnlp (p. 519-529).

[38] Mitchell, T. (1997). Machine learning. McGraw-Hill Education (ISE Editions).

[39] Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In (pp. 1137–1143). Morgan Kaufmann.

[40] Vallin, A., Magnini, B., Giampiccolo, D., Aunimo, L., & Ayache, C. (2006). Overview of the clef 2005 multi-lingual question answering track. In C. Peters

(Ed.), Accessing multilingual information repositories. Berlin, Heidelberg: Springer-Verlag.

[41] Forner, P., Forascu, C., Moreau, N., Osenova, P., Prokopidis, P., & Rocha, P. (2008). Overview of the clef 2008 multilingual question answering track. In C. P. et al. (Ed.), Working notes for the clef 2008 workshop. Berlin, Heidelberg: Springer-Verlag.

[42] Dang, H. T., Kelly, D., & Lin, J. (2008). Overview of the TREC 2007 question answering track. In Proceedings trec 2007.

[43] Ratnaparkhi, A.: A simple introduction to maximum entropy models for natural language processing. IRCS Technical Reports Series p. 81 (1997)

[44] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (1997). doi: 10.1162/neco.1997.9.8.1735, doi: 10.1162/neco.1997. 9.8.1735

[45] Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding with unsupervised learning. Technical report, OpenAI (2018)

[46] Peters, M.E. & others.: Semi-supervised sequence tagging with bidirectional language models. In: Barzilay, R., Kan, M. (eds.) Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers. pp. 1756–1765. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/P17-1161, doi: 10.18653/v1/P17-1161

[47] Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Walker, M.A., Ji, H., Stent, A. (eds.) Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers). pp. 2227–2237. Association for Computational Linguistics (2018). doi: 10.18653/v1/n18-1202, doi: 10.18653/v1/n18-1202

[48] Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019,

Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics (2019). doi: 10.18653/v1/n19-1423, doi: 10.18653/v1/n19-1423

[49] Cheng, X., Xu, W., Chen, K., Wang, W., Bi, B., Yan, M., Wu, C., Si, L., Chu, W., Wang, T.: Symmetric regularization based BERT for pair-wise semantic reasoning. CoRR abs/1909.03405 (2019), http:// arxiv.org/abs/1909.03405

[50] Glass, M.R., Gliozzo, A., Chakravarti, R., Ferritto, A., Pan, L., Bhargav, G.P.S., Garg, D., Sil, A.: Span selection pre-training for question answering. CoRR abs/1909.04120 (2019), http://arxiv.org/ abs/1909.04120

[51] Zhang, Z., Wu, Y., Zhao, H., Li, Z., Zhang, S., Zhou, X., Zhou, X.: Semantics-aware BERT for language understanding. CoRR abs/1909.02209 (2019), http://arxiv.org/abs/1909.02209

[52] Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: Superglue: A stickier benchmark for general-purpose language understanding systems. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada. pp. 3261–3275 (2019), http://papers.nips.cc/paper/ 8589-superglue-a-stickier-benchmark-for-general-purpose-language-understanding-systems

[53] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: GLUE: A multi-task benchmark and analysis platform for natural language understanding. CoRR abs/1804.07461 (2018), http://arxiv. org/abs/1804.07461

[54] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized BERT pretraining approach. CoRR abs/1907.11692 (2019), http: //arxiv.org/abs/1907.11692

[55] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A lite BERT for selfsupervised learning of language representations. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020), https: //openreview.net/forum?id=H1eA7AEtvS

[56] Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C.,

Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/n19-1423, https://doi.org/10.18653/v1/n19-1423

[57] Hu, Z.: Question answering on squad with bert. CS224N Report, Stanford University. Accessed pp. 01–09 (2019)

[58] Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543.

[59] Wang, Z., Mi, H., and Ittycheriah, A. (2016). Sentence similarity learning by lexical decomposition and composition. arXiv preprint arXiv:1602.07019.

[60] Gong, Y., Luo, H., and Zhang, J. (2017). Natural language inference over interaction space. arXiv preprint arXiv:1709.04348.

[61] Kim, S., Hong, J.-H., Kang, I., and Kwak, N. (2018). Semantic sentence matching with densely-connected recurrent and co-attentive information. arXiv preprint arXiv:1805.11360.

[62] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American society for information science, 41(6):391–407.

[63] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[64] Tom B. Brown and others. Language models are few-shot learners, 2020.

[65] Web browser system requirements - support.google.com/chrome/a/answer/7100626