Technische Universität Braunschweig

Institut für Informationssysteme

Master Thesis

Matching Paragraphs to answer entity based Questions

Submitted by:

Suma Kori

ITIS Matriculation Number: 10010744

TU Braunschweig Matriculation Number: 4850522

Supervised by:

Prof. Dr. Wolf-Tilo Balke

Braunschweig, January 2020

Declaration

I hereby declare that I have written this work myself and I have not used any tools and information sources other than the ones mentioned and all citations have been indicated.

Braunschweig, January 2020

Suma Kori

Abstract

In recent years, the mechanism of question answering has become one of the popular research topics. Question answering has applications in information retrieval, entity extraction, and so on. It also involves selecting an answer from the given context or to make a relevance judgment on the retrieved paragraphs (which contains the answer) so that most relevant answers to the questions can be provided to the user. The thesis aims to improve the mechanism of question-answering in the biomedical area by providing relevant answers to the questions which are being asked by real biologists.

The classification of the retrieved paragraphs based on their relevancy will be done by re-implementing the deep learning model “Siamese LSTM with attention” and by fine-tuning the pre-trained language model “Bidirectional Encoder Representation from Transformers (BERT)”. Both models are able to classify the new paragraphs but the pre-trained models boost the performance without taking much time to converge. The deep learning model works pretty well if the model is trained with a large amount of training data. And also the performance goes down for the longer sequence. In contrast, ‘BERT’ model is well suited for long distant dependencies. The self-attention mechanism of the transformer allows the model to read the paragraph in both directions and generates more accurate embeddings for each word. The thesis also aims to prove that the pre-trained BERT model could solve the classification problem in the biomedical domain more effectively.

Table of Contents

List of figures

List of tables

1. Introduction

The main goal of “Information retrieval” is to retrieve documents that are relevant to the user’s information needs. In contrast, the goal of information extraction is, to process document text to provide the user with one or more answers to a question or to aninformation need[1]. “Question-Answer” (QA) systems are useful in retrieving useful information from the web and providing insights. The mechanism of QA is a task in natural language processing that requires both natural language understanding and world knowledge. It has applications in many tasks including information retrieval, entity extraction, chatbots and so on. While question answering can be done in various ways, perhaps the most common characteristic of QA is selecting the answer from a given context. In other words, the system will pick a span of text from the context that correctly answers the question. If a correct answer cannot be found from the context, the system will merely return an empty string. Another way is to make a relevant judgment on the retrieved paragraphs (paragraphs containing the answer) so that the most relevant answers to their questions can be provided to the user.

In Information retrieval, we deal with queries and documents. In this case, queries are questions and documents are the list of paragraphs. Classifying the relevant paragraphs to the given question is challenging. The relevance judgment of the paragraph has been done by experts manually[1]. For the large collections of documents, the manual elicitation of relevance judgment is impractical. Making explicit relevance judgment is very expensive because it is necessary to cover a diverse set of queries in different contexts. This is practically an intensive task, often tedious and obviously error-prone. Manual elicitation is also a time-consuming and expensive process involving human beings. The thesis aims to solve this problem by re-implementing the deep learning model and by fine-tuning the pre-trained state of the art language model. Whenever the model analyses the new documents, it should be able to judge if the paragraph is relevant to the question or not.

The users are always interested in knowing the relevant answer to the question rather than looking at the document. The thesis mainly focuses on how to improve the QA problems and to provide relevant answers to the questions which are being asked by real biologists in the biomedical domain. The users of the biomedical literature expect relevant, short and specific answers to the question and to put them in context by providing supporting information. Meanwhile, manual judgment will be avoided sinceit is a time-consuming and expensive process involving human beings.

The QA problem will be solved in the thesis by applying deep learning model and a fine-tuning pre-trained language model. The results of both models will be compared. The task is to check If the fine-tuning pre-trained model is better in capturing the relation between question and answer and able to classify the new document correctly than a deep learning model?

Deep learning has achieved great success in image and speech recognition problems. Recently many deep learning-based methods have been proposed for the QA task as explained in the following Related work chapter. The proposed models have achieved the F1 score of 75.3% and the human performance is 91.2%. Because deep learning models have multi-layer structure and this structure is very helpful for the model to extract complicated information from input. In this project, we are adopting the idea of proposed deep learning models to build the "Siamese LSTM network with attention"[2]. Most of the proposed models have used the bidirectional recurrent neural network (RNN) with a self-attention mechanism in order to access the relationship between question and context. Although bidirectional RNN has access to both past and future context information, the range of context is limited due to the vanishing gradient problem. To avoid this problem the LSTM architecture with the attention mechanism has been used in this thesis work.

The idea of the Siamese LSTM network [2]will be used to build two identical LSTM networks. The attention mechanism on top of LSTM will be added to relate the current word in the paragraph with all other words in the paragraph. A classification layer will be created on top of this model to classify paragraphs.

But the disadvantage of using the deep learning model is, it takes a large amount of time for training the model from scratch. And it requires a large amount of data to train the model. With a small amount of data, the Deep Learning methods tend to overfit. In order to overcome these problems, we can make use of the Transfer learning (pre-trained model).

Transfer learning is the method of using the representation learned by one trained model for another model that needs to be trained on different data and for a similar/different task. Transfer learning uses pre-trained models i.e. models already trained on some larger benchmark datasets. Using transfer learning,deep learning applications can be built that solve NLP tasks much quicker. The pre-trained model can then be fine-tuned on small-data NLP tasks like question answering. This results in substantial improvement compared to training on these datasets from scratch.

The pre-trained model which will be used in this task is ‘‘Bidirectional Encoder Representation from Transformers (BERT)”[3]. BERT’s key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modeling. The pre-trained model can be fine-tuned with just one additional output layer to create state-of-the-art models for a QA task. With the help of Google BERT NLP, anyone can train their own state-of-the-art QA system in just 30 minutes on a single Cloud TPU and using a single GPU could be done in a few hours[3]. BERTmakes use of the Transformer, a self-attention mechanism that learns contextual relations between words in a text.

These two approaches will be experimented on ‘Text retrieval conference (TREC) Genomics dataset (2006)’ [4].

The transfer learning and pre-trained models can boost the performance without taking much time to converge, as compared to a model trained from scratch. Does this mean the fine-tuning the pre-trained paradigm is a clear winner against training from scratch?[5]. The thesis aims to examine whether the fine-tuning pre-trained model performs better in a QA task than deep learning models. Many approaches have been provided to solve QA problems as elaborated in the section 2. In this task, a model has been built to classify the paragraphs based on work that exists already. But the hypothesis is that pre-trained models are more powerful in order to capture the relationship between question and paragraphs and able to classify paragraphs more accurately due to the following reasons.

- The pre-trained model can either be context-free or contextual. The contextual representation can further be unidirectional or bidirectional[6].
    - In order to solve the QA tasks, the first step is to generate the embeddings of the question and paragraphs. In the deep learning model, PubMed pre-trained model will be used to get vectors that are induced from PubMed, PMC texts and their combination using the word2vec tool. Word2vec is a context-free model because it generates single word embeddings for each word in the vocabulary.
    - In contrast, BERT is the contextual bidirectional model. Bidirectional-because this model reads text from both directions (left and right) to gain a better understanding of the text.

  For example, given two sentences:

  The man was accused of robbing a bank

  The man went fishing by the bank of the river

  In the above two sentences, the 'bank' is a polysemy word. Word2vec would produce the same word embedding for the word "bank" in both sentences but BERT gives the different embeddings.

- The pre-trained models contain trained weights for the network. Instead of initializing the model with random weights, initializing it with the pre-trained weights reduces the training time and hence is more efficient.

Apart from the above reasons, less training data results in a poor approximation. In order to build a deep learning model, it has to be trained with a large amount of data. But pre-trained BERT model performs better even using the small amount of training data (since it has been already pre-trained on large data). All these points will be concluded in the section 6.

The thesis is organized as follows. In section 2, we briefly discuss open-domain question answering dataset, factoid vs non-factoid queries and someof the works carried out in the area of QA. In section 3, we discuss theTREC Genomics dataset. In section 4, we will discuss what are allpre-processing needs to be done before feeding the input to the model. Insection 5, we discuss the methodology of the thesis where we discuss ourapproaches to solve the QA problem. In section 6, we will have a look at theexperiments and analysis. And in section 7, we conclude the work along with future work.

2. Background Information and Related Work

This section includes three main parts as follows

1. Open-domain question answering dataset
2. Factoid vs Non-factoid queries
3. Models which are already implemented to solve QA problems

2.1. Open-domain question answering

There are two main varieties of QA datasets, those are open and closed datasets. In the open QA datasets, the answer depends on general world knowledge, in addition to any text provided in the dataset. The goal of open-domain QA is to answer a question from a large collection of documents[7]. In closed QA datasets, all information required for answering the question will be provided in the dataset itself[7].The open QA dataset will be used in this thesis work. There are several open-domain QA dataset as follows.

2.1.1. SQuAD

One of the most widely used datasets in research is the 'Stanford question answering dataset' (SQuAD)[8]. The SQuAD is a reading comprehension dataset, consisting of the question posed by crowd workers on a set of Wikipedia articles. Each question can be answered by finding the span of the text in the passage. So it is useful when building a QA model based on Information retrieval (IR) and Reading Comprehension (RC).

2.1.2. WikiQA

WikiQA dataset[9] is a publicly available set of question and sentence pairs, collected and annotated for research on open-domain question answering. WikiQA is constructed using a more natural process and is more than an order of magnitude larger than the previous dataset. WikiQA dataset includes questions for which there are no correct sentences, enabling researchers to work on answer triggering, a critical component in any QA system[9].

2.1.3. TREC QA

The Text Retrieval Conference (TREC) is co-sponsored by the National Institute of Standards and Technology(NIST) and the U.S. Department of Defense[10]. It started in 1992 to support the research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies.The annual evaluations at the Text Retrieval Conference (TREC) led to many advances in open-domain QA, many of which were used in IBM Watson for Jeopardy[10]. TREC has a question answering track since 1999, in each track the task was defined such that the systems were able to retrieve small snippets of text that contained an answer for the open-domain and the closed-class questions i.e., fact-based, short-answer questions that can be drawn from any domain[1].

2.2. Factoid vs Non-Factoid queries

Factoid queries are questions whose answers are short factual statements. The main feature of a factoid question is the presence of question words and the question starts with a question word, such as What, Where, Why, Who, Whose, When, Which and how. Most factoid questions are related to facts, current events, ideas and suggestions and could formulate an advice question, e.g. “How does p53 affect apoptosis?”. In addition, some factoid questions could contain two question words e.g. “**How** does Nurr-77 delete T cells before they migrate to the spleen or lymph nodes and **how** does this impact autoimmunity?”. Factoid questions could have any kind of information given as an answer or response.

The architecture of a factoid QA system can be reduced to a minimal schema that consists of three phases performed in a sequential pipeline. The following figure 1 shows the architecture and the information exchange between the three modules such as ‘Question Processing’, ‘Passage Retrieval’ and ‘Answer Extraction’.[11]

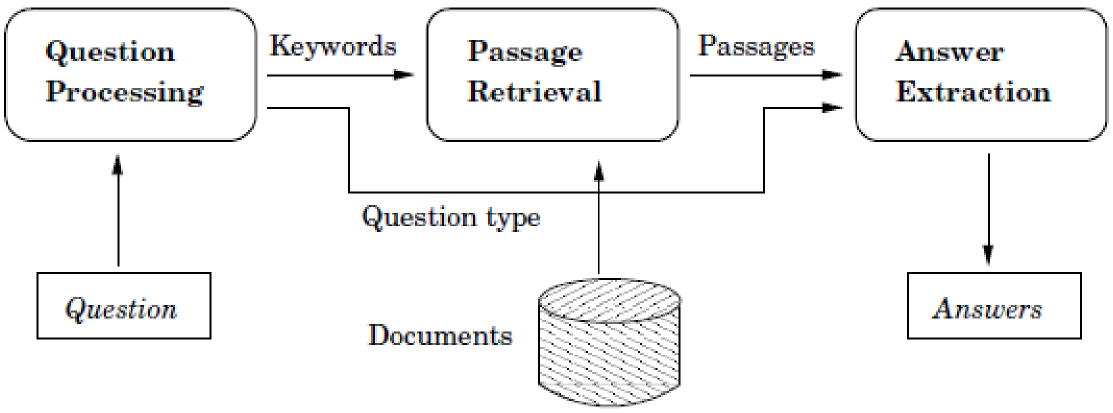


Figure 1: Overview of a generic QA architecture (6)

a. **Question Processing**: The Question Processing module analyses the question aiming to exploit some of the hyponyms relation(For example “What is the principal port in Ecuador?”, expects the name of a port, that’s a kind of location)[11].
b. **Passage Retrieval:** It is an information retrieval module. Documents or shorter passages will be retrieved from the documents collection using the keywords extracted by the Question Processing module.These passages should contain the answer to the question.The documents collection may be either a set of documents fixed for the evaluation or the Internet[11]. These collections are accessed by posting queries to local IR engines or to commercial web search engines.
c. **Answer Extraction:** This module extracts exact answers from the retrieved passages. This involves two different processes: first, a set of candidate answers are identified and then the answer is selected from the candidates[11].

Thequeries in TREC Genomics are with question words ‘what’ and ‘how’.

- What is the role of gene in disease?
- How do genes interact in organ function?
- How does a mutation in gene influence biological process?

In contrast, a non-factoid question covers all questions answering topics beyond factoid question answering. A non-factoid question can be about anything. It can be asked to provide an answer to a math problem or how to fix a specific model of an object. A non-factoid QA consists of finding already existing answers posted on community question answering sites.

2.3. Implemented models to solve QA problem

Several models are implemented for open-domain QA datasets. Most of the experiments are conducted using deep learning methods because deep learning performs much better for a large amount of the data. It would be really helpful for the QA dataset since most of the QA dataset contains a large amount of data.

2.3.1. Logistic Regression model

The "SQuAD:100,000+ Questions for Machine Comprehension of Text" [8] developed a strong "Logistic Regression"and compared the accuracy and F1 score with baseline methods. First, a human performance on SQuAD will be assessed. The human performance prediction will be evaluated and other answers will be kept as ground truth answers. To compare the performance of machines with the performance of humans following baselines are implemented.

Logistic Regression: Used to predict dependant variable given set of independant variable, such that dependant variable is categoric.

- The first baseline is a sliding window baseline. It could rank answer candidates by forming a bag-of-words vector for each answer paired with the question texts. First, a large number of possible answer candidates from the passage will be extracted. For each candidate answer, the unigram/bigram overlap between the sentence containing itand the questionwill be computed. Candidates that have maximal overlap will be kept.More concretely, this algorithm passes a sliding window over the whole texts and the size of such a window is equal to the number of words in the question-answerpair. The highest overlap score between a text window and the question-answer pair is taken as the corresponding score for the answer[8].
- To improve upon the sliding window baseline, a logistic regression baseline is implemented**[8]**. The logistic regression uses a range of features namely the lexicalized features and dependency tree path features as shown in table 1 with the example. Given the question"What causes precipitation to fall?", the sentence "In meteorology,

precipitation is any product of the condensation of atmospheric water vapor that falls under gravity." and answer "gravity". Q denotes question, A denotes candidate answer, and S denotes sentence containing the candidate answer as shown in table 1**[8]**.

The logistic regression model sits between the sliding window baseline and human performance as shown in figure 2.

Feature Groups	**Description**	**Examples**
Matching Word Frequencies	Sum of the TF-IDF of the words that occur in both the question and the sentence containing the candidate answer. Separate features are used for the words to the left, to the right, inside the span, and in the whole sentence	Span: [0 ≤ sum < 0.01] Left: [7.9 ≤ sum < 10.7]
Matching Bigram Frequencies	Same as above, but using bigrams. We use the generalization of the TF-IDF described in Shirakawa et al. (2015).	Span: [0 ≤ sum < 2.4] Left: [0 ≤ sum < 2.7]
Root Match	Whether the dependency parse tree roots of the question and sentence match, whether the sentence contains the root of the dependency parse tree of the question, and whether the question contains the root of the dependency parse tree of the sentence.	Root Match = False
Lengths	Number of words to the left, to the right, inside the span, and in the whole sentence.	Span: [1 <= num < 2] Left: [15 ≤ num < 19]
Span Word Frequencies	Sum of the TF-IDF of the words in the span, regardless of whether they appear in the question.	Span: [5.2 ≤ sum < 6.9]

Constituent Label	Constituency parse tree label of the span, optionally combined with the wh-word in the question.	Span: NP Span: NP, wh-word: “what”
Span POS Tags	Sequence of the part-of-speech tags in the span, optionally combined with the wh-word in the question.	Span: [NN] Span: [NN], wh-word: “what”
Lexicalized	Lemmas of question words combined with the lemmas of words within distance 2 to the span in the sentence based on the dependency parse trees. Separately, question word lemmas combined with answer word lemmas.	Q: “cause”, S: “under” case ←-- Q: “fall”, A: “gravity”
Dependency Tree Paths	For each word that occurs in both the question and sentence, the path in the dependency parse tree from that word in the sentence to the span, optionally combined with the path from the wh-word to the word in the question. POS tags are included in the paths.	VBZ nmod ---→ NN what nsubj ←-- VBZ advcl --→ + VBZ nmod ---→NN

Table 1: Features used in the logistic regression model

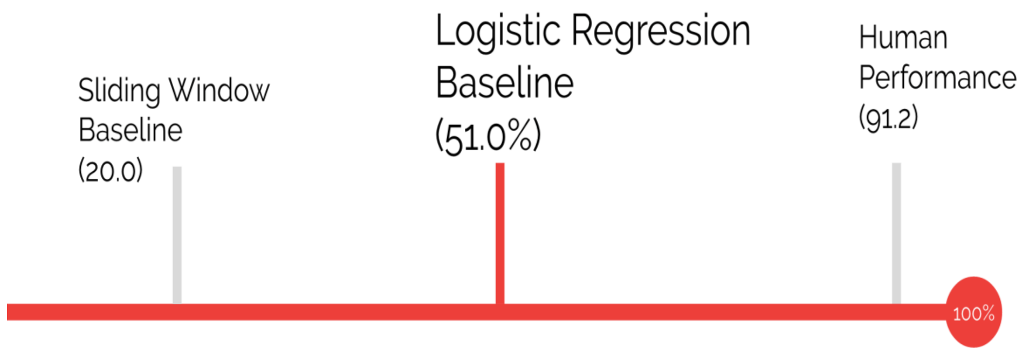


Figure 2: Comparing Logistic Regression with baseline**[8]**

The logistic regression achieves anF1 score of 51%, which is not a very good score and human performance is much higher.This concludes the simple classification algorithm works poor on QA problems.

2.3.2. RNN with Bidirectional attention flowlayer(BiDAF)

Another related work is the "NLP-Question Answering System using Deep Learning"[12] by creating a modified version of the bi-directional attention flow model. It consists of following building blocks

- Word Embedding layer: This layer converts the words into vectors using Glove or Word2Vec.
- RNN Encoder Layer: Bi-directional RNN allows each word in the context and question to make aware of other occurring words.
- BiDAF Attention + Self Attention: To get better performance a much more complex attention scheme is required. Attention flows from both contexts to answer and vice versa.
- Decoder Layer: Given a question-answer document there can be several answer spans corresponding to local maxima. The dynamic answer pointer decoder uses an iterative technique to predict the start point and endpoint. This allows the model to recover from the initial local maxima corresponding to the incorrect answer span.

Bi-Directional Attention Flow(BiDAF): It is a multi-stage hierarchial process that represents the context at different levels of granularity and uses bi-directional attention flow mechanism to obtain a query-aware context representation without early summarization

Model	Dev F1	Test F1
BiDAFAttention+BidaF decoder	74.4	75.14
BiDAFAttention+Self Attention	73.6	74.3
Coattention only	64.17	-
Coattention with BidaF decoder	69	-

Table 2: Performance of BiDAF

The above model achieves an F1 score of 75.142% on the test set with BiDAFwhich is a pretty good result.

2.3.3. Convolutional Neural Network(CNN)

Another work is based on Convolutional Neural Network (CNN)[13]. A QA corpus was created and released to setup a new QA task in the insurance domain.

- A general deep learning framework with several variants for the QA task is proposed and comparison experiments have been conducted.
- Novel techniques have been utilized to create multilayer and augmented CNN with discontinuous convolution and novel similarity metric that combine both L2-norm and inner product information[13].
  Figures 2, 3, 4 and 5 are different architectures of CNN which are used to solve QA[13].

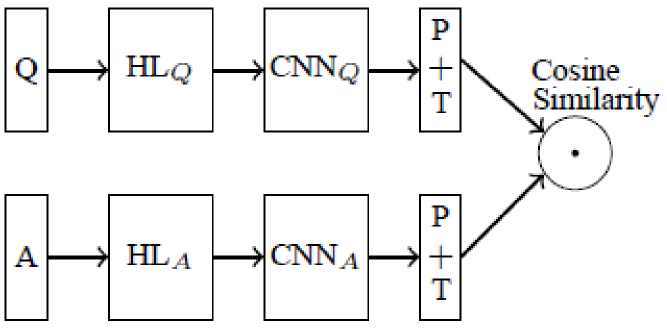


Figure 3: HL is a hidden layer, T is tanh layer and P is maxPooling

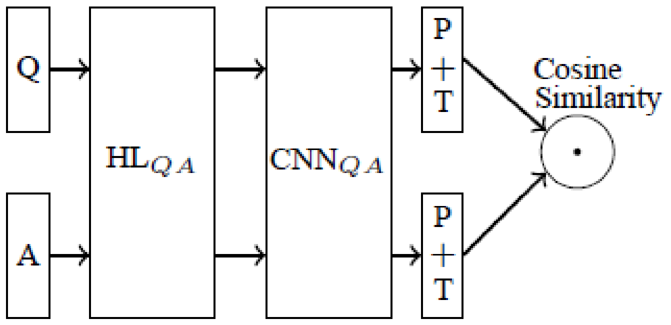


Figure 4: Weights are shared by Q and A

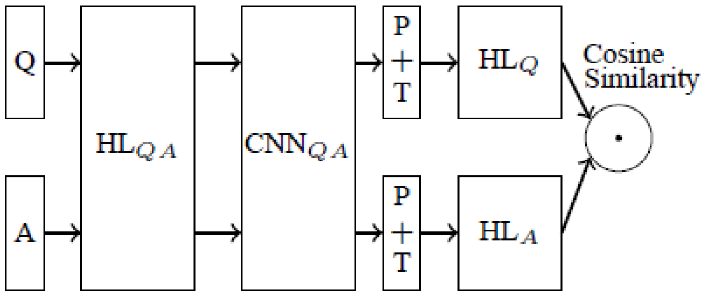


Figure 5: Added HLq and HLa after CNN

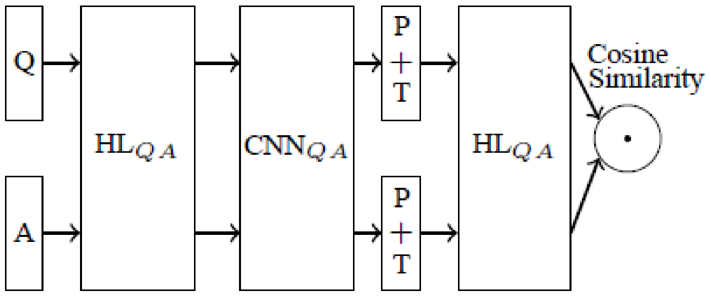


Figure 6: Added shared hidden layer HLqa after CNN

The QA dataset is experimented on above architectures by taking 200 hidden layers and 1000, 2000 and 3000 CNN filters. The results are shown in the below table. The top accuracy of the test corpus can reach up to 62.8%.

Idx	Dev	Test1	Test2	Description
1	31.9	32.1	32.2	Baseline: Bag-of-words
2	52.7	55.1	50.8	Baseline: metzler-bendersky IR model
3	44.2	41.7	39.5	Architecture 1: HL_Q (200) HL_A (200) CNN_Q (1000) CNN_A(1000)
4	58.2	57.8	53.6	Architecture 2: HL_{QA} (200) CNN_{QA} (1000)
5	36.1	33.6	32.7	Architecture 3: HL_{QA} (200)CNN_{QA} (1000)HL_Q (1000)HL_A (1000)
6	51.4	50.5	46.1	Architecture 4: HL_{QA} (200)CNN_{QA} (1000)HL_{QA} (1000)
7	47.0	46.7	43.0	Architecture 4: HL_{QA} (200)CNN_{QA} (1000)HL_{QA} (500)
8	60.6	59.2	55.1	Architecture 2: HL_{QA} (200) CNN_{QA} (2000)
9	61.5	61.3	57.8	Architecture 2: HL_{QA} (200) CNN_{QA} (3000)
10	**61.8**	**62.8**	**59.2**	Architecture 2: HL_{QA} (200) CNN_{QA} (4000)
11	59.7	59.3	55.6	Architecture 5: HL_{QA} (200) CNN_{QA} (1000)
12	59.9	60.6	55.9	Architecture 6: HL_{QA} (200) CNN_{QA} (1000)CNN_{QA} (1000)
13	59.9	58.7	53.8	Architecture 2: HL_{QA} (200) $Augmented - CNN_{QA}$ (1000)
14	60.0	60.3	54.3	Architecture 2: HL_{QA} (200) $Augmented - CNN_{QA}$ (2000)
15	61.7	62.2	56.3	Architecture 2: HL_{QA} (200) $Augmented - CNN_{QA}$ (3000)

Table 3: Accuracy score using CNN architectures (Severyn & Moschitti)

2.3.4. Question Answering on SQuAD with BERT

BERT[3], a language representation model is designed to pre-train deep bidirectional representation. The pre-trained model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference.

One of the most interesting related work is 'Question Answering on SQuAD with BERT'[14]. Most of the top 20 models on SQuAD 2.0 leaderboard are either BERT or BERT plus some other model (BERT+X). Prior to BERT, ELMO is a popular pre-trained deep contextualized word representation. ELMO stands for Embedding from Language Models. It is LSTM-based. It can be used as word embedding to improve model performance.'Question Answering on SQuAD with BERT'[14] explains the following three models for Question answering on SQuAD 2.0.The model will start with pre-trained BERT weights and fine-tune with SQuAD 2.0 training data[14].

ELMO: Embeddings from Language Models (ELMO) is pre-trained with a self-supervising task called a bidirectional language model and they show that the representation from this model is powerful and improves the state-of-the-art performance on many tasks.

- **BERT+BiDAF:** This model replaces BiDAF's embedding layers and attention flow layer with BERT. The input to the modelling layer is the final hidden state of BERT. It will produce probability distributions of start and end position for the answer span[15].

- **BERT+SAN:** SAN stands for a stochastic answer network. Its prediction generation module features multi-step reasoning, simulating the process of going through texts for multiple passes and refine predictions. Itpresents using SAN's answer module on top of BERT for natural language inference tasks. SAN for SQuAD 2.0, answer module will jointly train an answer span detector and an unanswerable question classifier[14].

- **Model3:**The last model uses a simpler output layer on top of BERT than the previous two. As it is the third model presented and is light-weight hence it is called Model3. It applies two dense layers to BERT's output, the activation function is Gaussian error linear units (gelu).

GELU: It is an activation function used in the most Transformers like BERT[27].

GELU(x) = $0.5x(1+\tanh(\sqrt{2/\pi}(x + 0.044715x^3)))$

Model	Dev (EM/F1)	Test (EM/F1)
BERT+BiDAF	74.0/76.9	-/-
BERT+SAN	74.9/77.6	-/-
BERT+Model3(single)	75.4/78.4	73.2/76.5
BERT + Model3(ensemble)	76.2/79.4	-/-

Table 4: BERT Performance on SQuAD 2.0 Dataset

The simpler model works better. This could be due to complex structures are hard to train with pre-trained BERT.

2.3.5. Learning to Rank with CNN

Most of the related works have experimented with SQuAD dataset. There are very few experiments conducted on the TREC QA dataset. The following related work is on the TREC dataset but the goal is to re-rank the question-answer pairs. The set of questions are collected from TREC QA tracks 8-13. The manual judgment of candidate answer sentences is provided for the entire TREC 13 set and for the first 100 questions from TREC 8-12. The motivation behind this annotation effort is that TREC provides only answer patterns to identify if a given passage contains a correct answer key or not. This results in many unrelated candidate answers marked as correct simply because regular expressions cannot always match the correct answer keys[16].

The paper "Learning to Rank Short Text Pairs with Convolutional Neural Networks"[16]presents a convolutional neural network architecture for re-ranking pairs of short texts by learning the optimal representation of text pairs and a similarity function to relate them in a supervised way from the available training data. The CNN model takes only words as input, thus requiring minimal pre-processing. The model has the following two main building blocks.

- Sentence model: The main building blocks are two distributional sentence models based on convolutional neural networks.The sentence model is used to learn good intermediate representations of the queriesand documents, which are later used for computing their semantic matching[16].
- Matching text pairs:The vectors generated by Sentence models are used to compute the query-document similarity scores, which together with the query and document vectors are joined in a single representation[16].

The results on TREC QA when augmenting deep learning models with word overlap features are shown in table 4. The performance is measured usingthe MAP and MRR.

Mean Reciprocal Rank (MRR): MRR is the mean of the reciprocal rank of the highest ranking relevant document. MRR only cares about the single highest-ranked relevant item[28]

Mean Average Precision (MAP): MAP is the mean of average precision, where the average precision of a single query is the mean of the precision scores at each relevant item returned in a search results list. MAP considers whether all of the relevant items tend to get ranked highly[28]

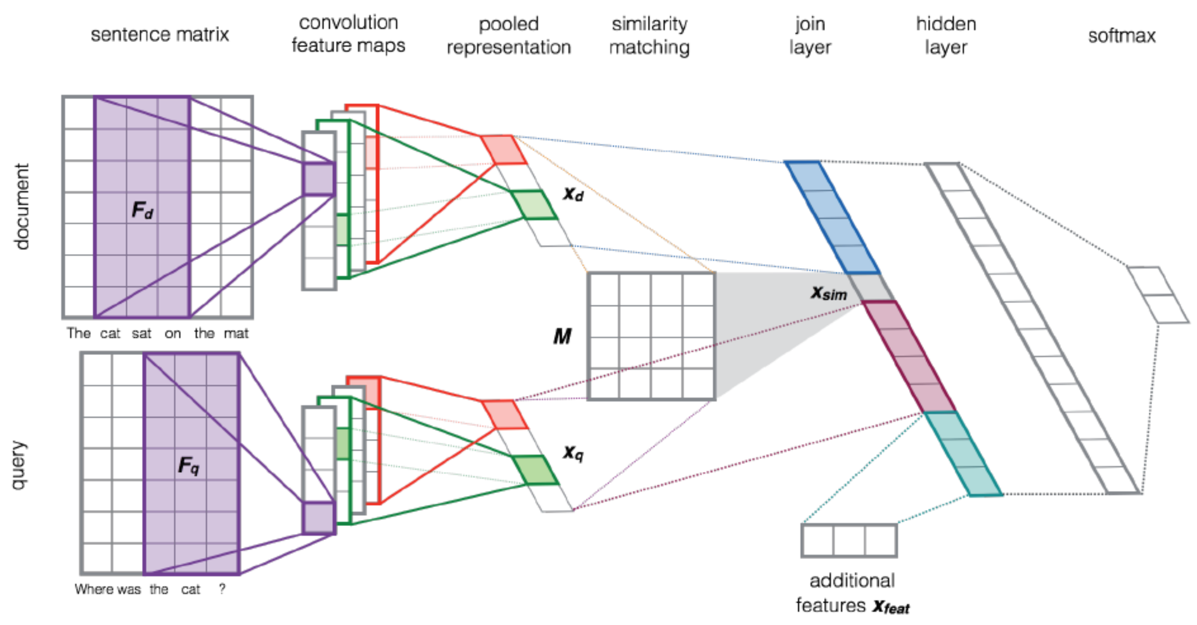


Figure 7: CNN architecture for re-ranking short text pairs[16]

An experiment is conducted by training a model with fewer data(Train) and a complete dataset(Train-all).

Model	MAP	MRR
CNN-Train	0.73	0.79
CNN-Train all	0.74	0.80

Table 5: Results on TREC QA using CNN

The overall performances of the discussed models are shown in the table. The single BERT model and BiDAF Attention model perform better than all othermodels. The notions of these models in this thesis to improve the QA problem for TREC Genomics dataset were used.

Model	Performance
Logistic Regression	51.0
CNN for Re-ranking	MAP 74 and MRR 80
BiDAFAttention+BiDAF decoder	75.14
BERT+Model3(single)	75.4

Table 6: Overall performance of an individual model

2.4. Summary of Related Work

The aim of the discussed models as above was to select the answer from a given context by picking a span of text from the context that correctly answers the question. But the idea of the thesis is to classify the relevant paragraphs to the question. Even though the idea is quite different from the discussed work, we can still adopt the notion of deep learning models to create vector representations and capture the meaning of questions and paragraphs for the TREC Genomics dataset. In order to classify paragraphs, the model should be aware of entities in the query and presence of the entity or synonym of the entity in the paragraph.This can be achieved by attention or self-attention layer which obtains a query-aware context representation without early summarization[12].

3. Dataset

The Text Retrieval Conference (TREC) is co-sponsored by the National Institute of Standards and Technology(NIST) and the U.S. Department of Defense[10]. It started in 1992 to support the research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies[10].

The main goals of TREC are[10]:

- to encourage research in information retrieval based on large test collections
- to increase communication among industry, academia, and government by creating an open forum for the exchange of research ideas
- to speed the transfer of technology from research labs into commercial products by demonstrating substantial improvements in retrieval methodologies on real-world problems
- to increase the availability of appropriate evaluation techniques for use by industry and academia, including the development of new evaluation techniques more applicable to current systems.

NIST provides a set of documents and questions for each TREC. Participants run their own retrieval system on the data and a list of retrieved top-ranked documents isreturned to NIST[10]. NIST pools the individual result, judges the retrieved documents for correctness and evaluates the results. The evaluation effort has grown in both the number of participating systems and the number of tasks each year. TREC has successfully achieved itsgoals of improving the state-of-the-art in information retrieval and of facilitating technology transfer[10]. There are various TRECs dataset available from TREC1 to TREC 12.

The dataset used in this project is the 'Text retrieval conference(TREC) Genomics dataset (2006)'[4] is an open domain question answering dataset. TREC has a question answering track since 1999, in each track the task was defined such that the systems were to retrieve small snippets of text that contained an answer for the open-domain and the closed-class questions (i.e., fact-based, short-answer questions that can be drawn from any domain)[1]. In TREC 2006 the questions are collected from real biologists and the research group submits their best of search results. The Experts determined the relevance of paragraphs and grouped them into aspects identified by one or more entities. The Experts were provided with guidelines and a one-hour training course to improve the judging process[1].

3.1. Document collection

There are three distinct parts in the dataset which are the Documents, the Questions (Topics) and the Relevance Judgments (Right answers). The documents come from the new full-text biomedical corpus. The full collection contains 162,259 documents from 49 journals. Thename of each document file is its PMID along with .html extension, which facilitates accessing the associated MEDLINE record[1].

3.2. Topics

The topics were expressed as questions. They are derived from the set of biologically relevant questions based on the General topic types(GTT). Each question had one or more aspects that were contained in the literature corpus.

3.2.1. General topic types(GTT)

TREC questions fall into three categories called GTTs (General topic types) as shown in table 6[1]. All questions had the general format of containing one or more biological objects and process the explicit relationship between them[1].

- **Biological objects:** Genes, proteins, gene mutations
- **Biological process**: Physiological process or disease.
- **Relationship**: Usually “contributes to”, ”affects”, “associated with” and“regulates”**.**

GTT	Question Pattern	Example
Find articles describing the role of gene involved in a given disease.	What is the role of gene in disease?	1. What is the role of PrnP in mad cow disease? 2.What is the role of IDE in Alzheimer’s disease?

Find articles describing interactions(e.g., promote, suppress, etc.) between two or more genes in the function of an organ or in a disease.	How do genesinteract in organ function?	1.How do Bop-Pesinteractions affect cell growth? 2. How do mutations in the Presenilin-1 gene affect Alzheimer's disease?
Find articles describing one or more mutations of a given gene and its biologicalimpact.	How does a mutation in gene influence biological process?	1.How does BARD1 regulate BRCA1 activity? 2.How does p53 affect apoptosis?

Table 7: General Topic Types and Examples [1]

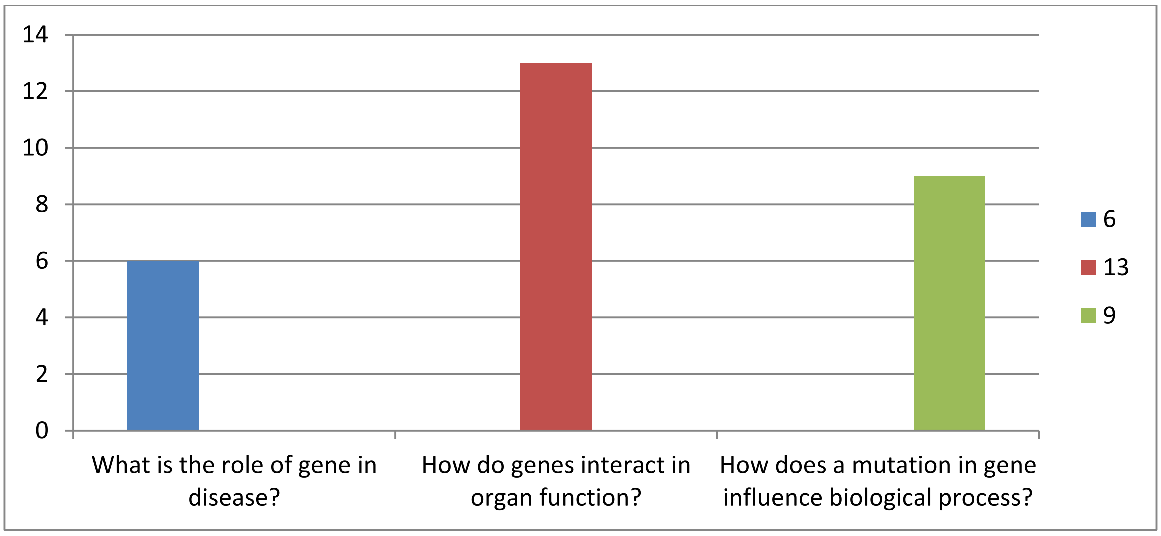


Figure 8: Number of a question in each pattern

3.3. Submitted Paragraphs

The submitted runs could contain up to 1000 passages per topic that were predicted to be relevant to answering the topic question. In the dataset, actual passages are not given but they are identified by the PMID, offset and lengthof the passages in character. The first character of each file was defined to be at offset zero[1].

3.4. Judging

Relevance judges were provided with guidelines and a one-hour training course to improve the judging process. To assess the relevance, the Judges were instructed to break down the question into required elements (e.g. the biological entities and processes that make up the GTT) and isolate the minimum contiguous substring that answered the question[1]. The judges also assigned one or more Medical Subject Headings(MeSH) terms along with subheadings to capture similarities and differences among retrieved passage aspects. Judges were instructed to use the most specific MeSH terms, with the option of adding subheadings. If one MeSH term was not sufficient to denote all aspects of the gold standard passage then judges assigned additional MeSH terms. The passages which are judged as definitely or possibly relevant were required to have a gold standard passage and atleast one MeSH terms.[1]

Gold standard passage: The judges evaluated the text of the maximum-length legal span for relevance and identified the portion of this text that contained an answer

MeSH: Medical Subject Heading(MeSH) is a comprehensive controlled vocabulary for the purpose of indexing journal articles and books in the life sciences. It serves as a thesaurus that facilitates searching

3.5. Occurances of passages per class

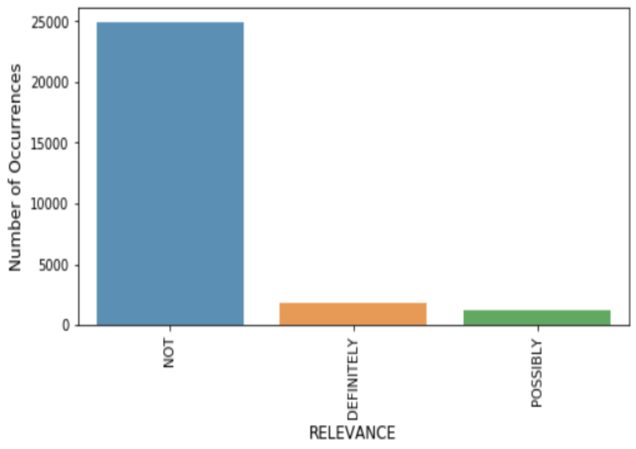


Figure 9: Number of passages per class

Relevance Judgement	Occurrences
Not	24934
Possibly	1237
Definitely	1828

Table 8: Number of Occurrences

The dataset is imbalanced. 89% of the submitted results are not relevant.

3.6. Instructions to judge

The following instructions are provided inorder to judge paragraphs. Since the questions are entity-based questions it is very important to review the query and paragraphs properly.

3.6.1. Evaluate the paragraphs for relevance

The questions will be reviewed if there is a gene or protein mentioned andthe synonym for it will be identified. For biological process or disease, familiarize with more general concepts, as well as sub-topics. For example, if there is an entity "mad cow disease" in the question, the paragraphs may or may not contain the same entity but paragraphs may contain sub-topics or synonyms of that entity.

Entity	**Subtopic/Synonym**
Mad cow disease	BSE or TSE
PrnP	PrPSc Polymorphism
IDE	Amyloid beta-Protein

Table 9: Synonym/Subtopic of Entities

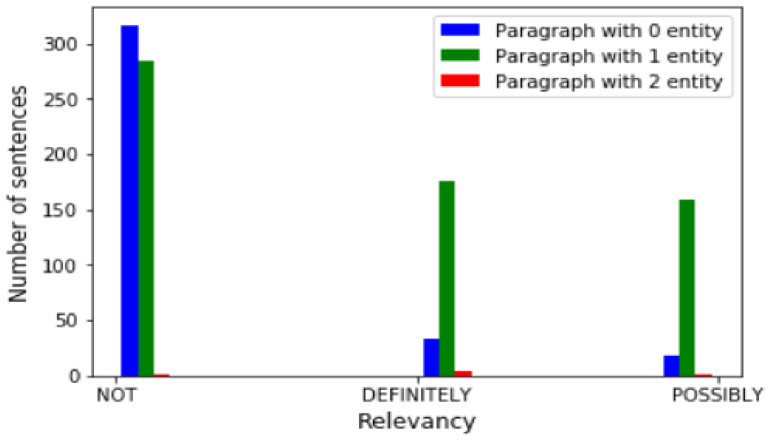


Figure 10: Entity occurrences in Paragraphs

Analyze the query “What is the role of PrnP in mad cow disease?”. The entities according to GTTs are ‘PrnP’ and ‘mad cow’. As shown in Figure 10, It is not necessary that Paragraphs should contain only these entities inorder to judgeparagraph as “Definitely” or “Possibly Relevant”. Most of the paragraphs contain the entity ‘PrnP’ but not ‘mad cow’.

Possibly Relevant Reason: mad cow is a member of the Transmissable Spongiform Encephalopathies (TSE) disease family	Transmissible spongiform encephalopathies (TSE) are fatal degenerativedisorders of the central nervous system that occur naturallyin man and sheep. Common to all TSE diseases is the accumulationof an abnormal form of the normal prion protein (PrP), primarilyin the brain and spinal cord. In its abnormal configurationthe PrP (PrPSc) is infectious and extremely resistant to

	proteolyticenzymes . The normal PrP protein (PrPC)is expressed in most tissues of the body, with the highest expression in the nervous tissues
Definitely Relevant Reason:mad cow disease is known as bovine spongiform encephalopathy, abbreviated as BSE.	The study shows that the antibody-labelling patterns of intracellular PrPd are the same in sheep of different genotypes infected with BSE by different routes and can be differentiated readily from those of all natural sheep scrapie sources so farexamined. Wide-scale testing for sheep BSE based on proteinase-digested PrPres fragment sizes may therefore be feasible. In addition,these findings provide support for earlier studies and furthersuggest that sheep homozygous for alanine at codon 136 and glutaminecodon 171 are more susceptible to BSE-agent infection than areother sheep genotypes
Not Relevant Reason:CJD is different disease	Human diseases include Creutzfeldt-Jakob disease (CJD), Gerstmann-Str&auml;ussler-Scheinkerdisease, kuru, and fatal insomnia . CJD, by far the mostcommon human TSE, may occur as a sporadic disease of unknownetiology (sCJD), a disease associated with mutations in theprion protein gene (PRNP), or a proven exogenous infection.The latter group includes variant CJD (vCJD) a distinct diseasephenotype that is believed to have been transmitted from cattleto humans through the consumption of contaminated meat

Table 10: Reason for Relevancy

3.6.2. Code results with mesh terms

Participants in TREC Genomics have been rewarded for the breadth of answers submitted[1]. To determine breadth, excerpts will be coded with standardized terms, allowing grouping of related results. The standardized terms are taken from the Medical Subject Headings (MeSH) thesaurus[1].MeSH has different types of terms, including headings, subheadings, and entry terms. Text extracts should be labeled with individual MeSH headings, without subheadings ifpossible. A combination of MeSH heading and subheading can be used if required. An extracted passage may be tagged with more than one MeSH term if necessary. Select MeSH terms that contribute to answering the question. Terms that merely replicate one of the required elements of the question aren't useful for distinguishing different types of answers. Sometimes the answer to a question is best represented by an action term, such as "peptide hydrolysis". Nouns describing objects (versus actions) are more prevalent in MeSH, and the MeSH term "peptide hydrolases" should be considered as an alternative.

Question	Excerpt	MeSH term	Reason
What is the role of PrnP in mad cow disease?	Bovine Prion Protein Gene (PRNP) Promoter Polymorphisms Modulate PRNP Expression and May Be Responsible for Differences in Bovine Spongiform Encephalopathy Susceptibility	Polymorphism, Genetic	PRNP polymorphisms influence mad cow disease.

What is the role of PrnP in mad cow disease?	Transmissible spongiform encephalopathies (TSEs), or prion diseases, are mammalian neurodegenerative disorders characterized by a posttranslational conversion and brain accumulation of an insoluble, protease-resistant isoform (PrPSc) of the host-encoded cellular prion protein (PrPC)	PrPSc Proteins	PrPSc is the general name of the diseaseforming isoform of prion proteins like PrnP.
How does BRCA1 ubiquitinating activity contribute to cancer?	The RING heterodimer BRCA1- BARD1 is a ubiquitin ligase inactivated by a breast cancer-derived mutation	UbiquitinProtein Ligase Complexes, Breast Neoplasms	Specifies role in ubiquitin pathway; specified type of cancer.
What is the role of IDE in Alzheimer's disease?	Taken together these results suggest that the use of insulysin to hydrolyze A peptides represents an alternative gene therapeutic approach to the treatment of Alzheimer's disease.	Amyloid betaProtein Precursor, Peptide Hydrolases	It is the peptidase activity of insulysin acting on A beta that is thought to impact Alzheimer's disease.

Table 11: MeSH terms applied to excerpts

4. Pre-Processing

It is necessary to find the embeddings of question-answers before feeding input into a neural network. These embeddings are useful for keyword/search expansion, semantic search and information retrieval. A Pre-trained neural network produces word embeddings which are then used as features in these models. The pre-trained models contain trained weights for the network. Instead of initializing the model with random weights, initializing it with the pre-trained weights reduces the training time and hence is more efficient.

In this project, PubMed pre-trained model is used to get vectors of the Question-Paragraph pair. Word vectors were induced from PubMed and PMC texts and their combination using the word2vec tool. The word vectors are provided in the word2vec binary format. Similar words are represented by numbers that are closer to each other and this provides the representation ofthe semantic value to it. Vocabulary dictionary stores word to index mapping and inverse Vocabulary dictionary stores index to word mapping.

4.1. Pre-processing in BERT

In BERT model, the data will be pre-processedin the same way it was trained. First, the tokens of Questions and Paragraphs will be generated. CLS at the beginning and SEPbetween Question and Paragraph pair will be inserted. In the end, segment ids will be generated to indicate whether a token belongs to the first sequence or the second sequence[3].

[CLS] [Tokens of question] [SEP] [Tokens of paragraph] [SEP]

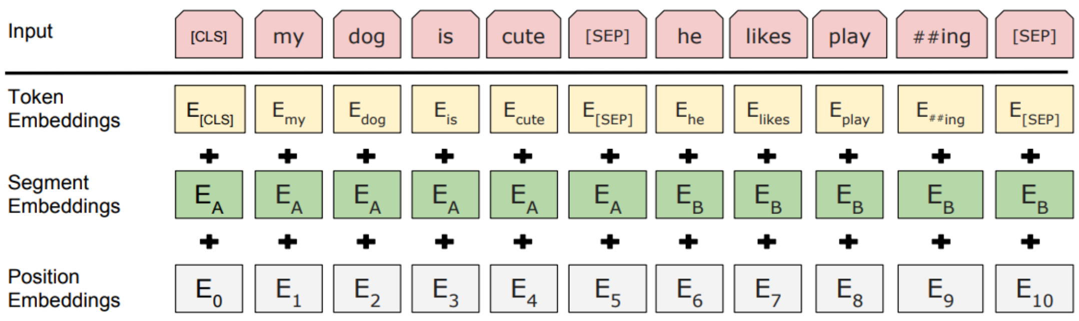
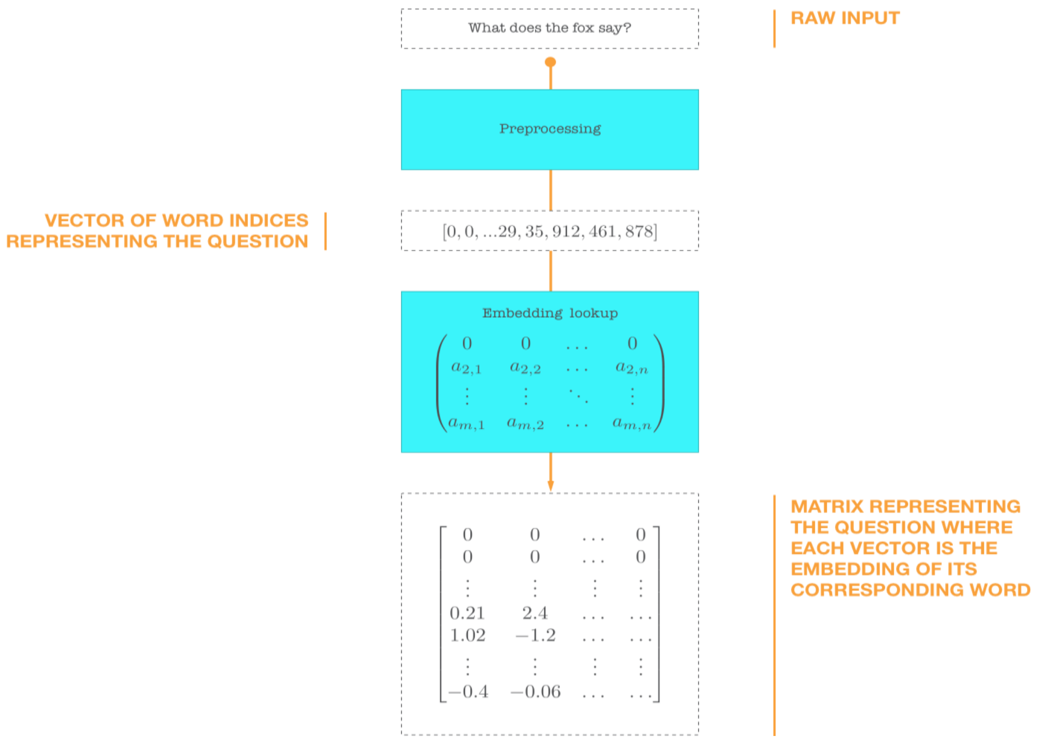

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 11: Input representation in BERT (Devlin, Chang, Lee, & Toutanova, 2019)

4.2. Pre-processing in LSTM

In LSTM with an attention model, the raw text needs to be converted to a list of word vector indices. A helper function takes a string as input and will output a list where each entry is a single word from the text and does some pre-processing like removing the specific sign. Word indices uniquely identify words from the text. These indices are fed into the embedding layer and the layer looks up the corresponding embedding for each word and encapsulates them into a matrix.



What does the fox say?
RAW INPUT
Preprocessing
VECTOR OF WORD INDICES REPRESENTING THE QUESTION
[0, 0, ...29, 35, 912, 461, 878]
Embedding lookup
MATRIX REPRESENTING THE QUESTION WHERE EACH VECTOR IS THE EMBEDDING OF ITS CORRESPONDING WORD

$$\begin{pmatrix} 0 & 0 & \cdots & 0 \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

$$\begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0.21 & 2.4 & \dots & \dots \\ 1.02 & -1.2 & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots \\ -0.4 & -0.06 & \dots & \dots \end{bmatrix}$$

Figure 12: Input representation in LSTM[13]

5. Methodology

The thesis aims to solve the QA problem using Deep learning and by the fine-tuning pre-trained language model.

1. A deep learning model:Siamese LSTM architecture with attention[2]
2. Fine-tuning Pre-trained model: Bidirectional encoder representation from Transformer(BERT)[3]

These models classify the new documents according to their degree of relevance.The model should be able to understand the semantic relation between questions and paragraphs.Paragraphs are classified based on the presence of entities or MeSH terms of the query. The submitted paragraphs will be judged as being "definitely relevant (2)", "possibly relevant (1)" or "not relevant (0)"[1]

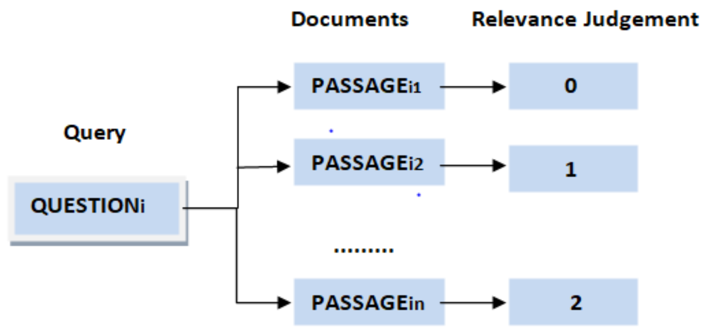


Figure 13: Relevance Judgement

Definitely relevant: The paragraph is definitely relevant if it contains all required elements of the question and answers the question.

Possibly relevant: The paragraph is possibly relevant if it contains the majority of required elements, missing elements are within the realm of possibility.

Not relevant: The paragraph is not relevant if it does not contain any required elements of the question.

The performance of the models will be compared with the baseline model i.e. "Manhattan LSTM model"

5.1. Baseline Model: Manhattan LSTM model

The baseline modelis the Siamese LSTM network[2]. Siamese networks are networks which have two or more identical sub-networks in them.Siamese LSTM Model has two identical LSTM networks.Embeddings of question and answer will be passed to thenetwork. The network will be trained to capture the meaning of the question and answers. The Output of the two LSTM will be merged using the Manhattan similarity function. In the end, the network will be trained using any optimizer.

Optimizer: Reducing the difference between the predicted output and actual output is the important task in machine learning. This is called as Cost function or Loss function. The optimizer update the weight parameters to minimize the loss function.

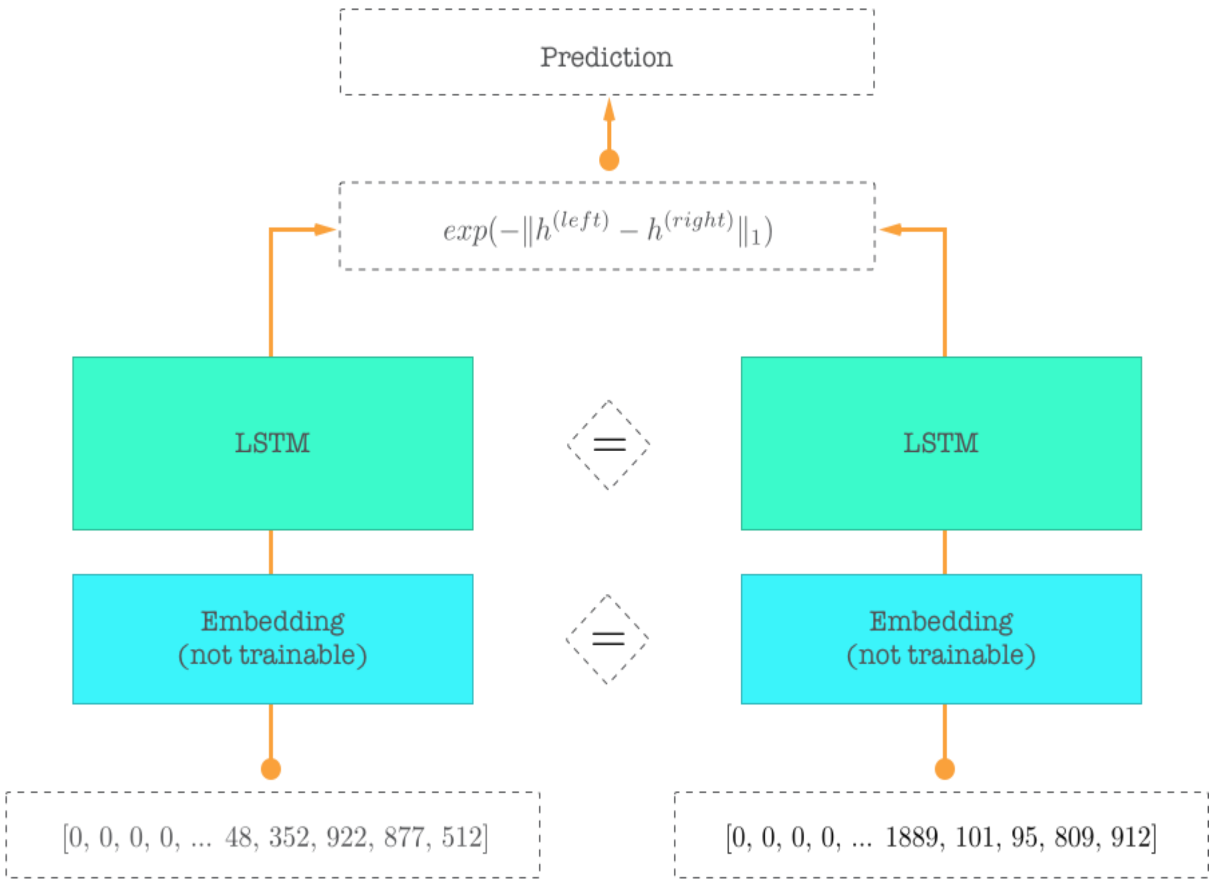


Figure 14: MaLSTM architecture

The model contains the following layers

1. **Input layer**: Input Question and Paragraph to the model
2. **Embedding layer**: Map each word into a low dimension vector
3. **LSTM layer**: Utilize LSTM to get high-level features from step 2
4. **Output layer**: Feature vector is finally used for classification.

5.1.1. Embedding layer

Given an input sentence consisting of n words denoted by S = $\{w_1\ , w_2 \ldots\ldots w_n\ \}$, every word w_i is converted into a real-valued vector e_i . For each word in S, look up the embedding matrix $W_{word} \in R^{dw*|v|}$, where V is a fixed-sized vocabulary, and d_w is the dimension of the vector. Then, the word representations X = $\{x_1\ , x \ldots\ldots x_n\}$ are obtained by mapping w_i to a column vector$xi \in R^{dw}$, which is then fed into the next layer(LSTM layer).

5.1.2. LSTM layer

Long short Term Memory networks[17] are a special kind of recurrent neural networks and are capable of learning long-term dependencies.RNN remembers things for small-time it may be reproducible, but once a lot of words are fed in, this information gets lost somewhere. This issue can be resolved by applying a slightly tweaked version of RNNs-the Long Short-Term Memory Networks. LSTMs make small modifications to the information by multiplications and additions. With LSTMs, the information flows through a mechanism known as cell states.

LSTM network comprised of different memory blocks called cells. There are two states that are being transferred to the next cell those are cell state and the hidden state. The memory blocks are responsible for remembering things and manipulations to this memory are done through three major mechanisms called gates, ‘Forget gate’, ‘Input gate’ and ‘Output gate’.

a. Forget Gate

The first step in the LSTM is to identify information that is not required and will be thrown away from the cell state. This decision is made by a sigmoid layer called a forget gate layer. The sigmoid layer looks at h_{t-1} (information from the previous timestamp) and x_t (new input) and outputs a number between zero and one. 1 represents keep the information and 0 represents remove the information. This is required for optimizing the performance of the LSTM network.

Sigmoid function: It is especially used for models where the probability of the output will be predicted. The probability exists only between the range 0 and 1.

$$\boldsymbol{f_t = \sigma(w_f[h_{t-1}, x_t] + b_f)}$$

$$w_f = Weight, h_{t-1} = Output\ from\ previous\ time\ stamp$$

$$x_t = New\ input, b_f = Bias$$

b. Input gate

The input gate is responsible for the addition of information to the cell state. This whole process comprises the following steps.

1. A sigmoid layer calls the “Input gate layer” decides which values will be updated.
2. Next, a tanh layer creates a vector of new candidate values, which could be added to the state.
3. Multiplying the value of the sigmoid layer to the created vector (the tanh function) and then adding this useful information to the cell state via addition operation.

Tanh function: The tanh function a.k.a. hyperbolic tangent function, is a rescaling of the logistic sigmoid. The range of the tanh function is from -1 to 1.

$$\boldsymbol{i_t = \sigma(w_f[h_{t-1}, x_t] + b_i)}$$

$$\boldsymbol{\tilde{c_t} = \tanh(w_c[h_{t-1}, x_t] + b_c)}$$

$$\boldsymbol{c_t = f_t * c_{t-1} + i_t * \tilde{c_t}}$$

$$c_t = new\ cell\ state(memory)at\ timestamp(t).$$

$$\tilde{c_t} = represents\ candidate\ for\ cell\ state\ at\ timestamp(t).$$

c. Output gate

A sigmoid layer which decides what parts of the cell state are going to output will be run. The cell state through tanh will be added and it will be multiplied by the output of the sigmoid gate.

$$\boldsymbol{o_t = \sigma(w_o[h_{t-1}, x_t] + b_o)}$$

$$\boldsymbol{h_t = o_t * \tanh(c_t)}$$

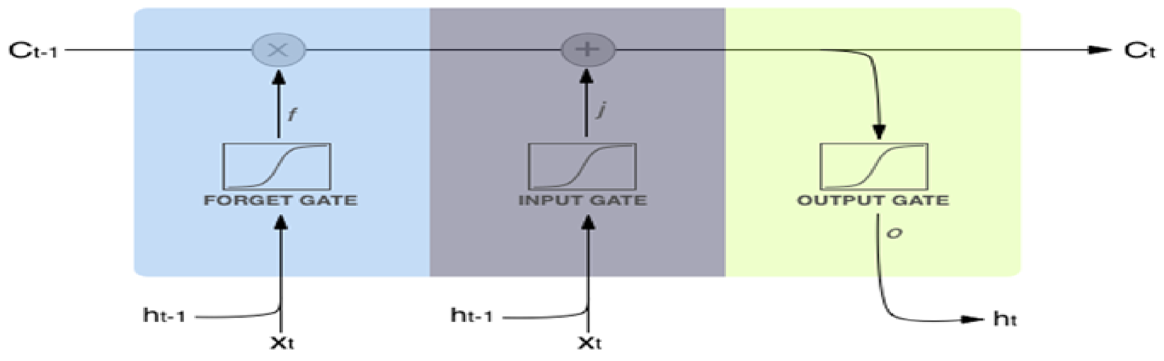


Figure 15: LSTM memory cells

5.1.3. Manhattan function

Similarity functions are used to measure the distance between two vectors or pairs. The two objects are similar if the distance between them is small and vice-versa. The general goal of Manhattan LSTM is to compare two sentences that can decide they are the same or not.

The two vectors that hold the semantic meaning of Question and answer will be merged using Manhattan similarity function

$$\exp(-|| h^{(left)} - h^{(left)} ||)$$

5.2. Siamese LSTM with attention

The architecture is similar to the baseline LSTM model.Additionally, the attention mechanism will be used.The model has two identical LSTM network and the final hidden state of both LSTM will be merged usingKeras merge layer. These features are fed into the attention layer which relates the current word in the paragraph with all the other words in the paragraphs. At the last network can be trained usingAdam optimizer and compute performance metric (Precision, Recall, and F1 score, etc.).

Adam Optimizer: Adam optimization involves using the first moments and second moments of the gradients. The first moments involves the exponentially decaying average of the previous gradients and the second moment involves exponentially decaying average of the previous squared gradients[26].

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \rightarrow (\text{mean}),$$
$$v_t = \beta_1 v_{t-1} + (1 - \beta_2) g_t^2 \rightarrow (\text{uncentered variance}),$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t},$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t},$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t,$$

Adam optimizer uses the combined averages of previous gradients at different moments to give it more expressive power to better adaptively update the parameters.

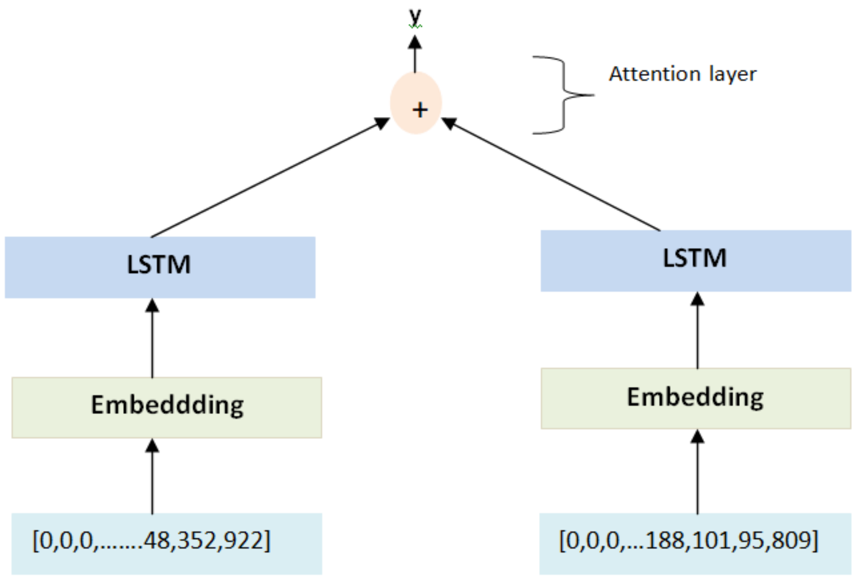


Figure 16: Siamese LSTM with Attention

5.2.1. Attention layer

Attention is the way to look back more into the past and be able to influence the future[18]. Important information can appear at any position in the sentence. The main idea of attention mechanism proposed at the word level is to consider the importance distribution of each word in the contextual sentence.Attention neural network is the most widely used network for machine translation, speech recognition, to image captioning.

Let ‘H’ be a matrix consisting of output vectors that are produced by LSTM output, where ‘T’ is the sentence length. The representation ‘r’ of the sentence is formed by a weighted sum of these output vectors[19]:

$$M = \tanh(H)$$

$$\alpha = \mathrm{softmax}(w^{T} M)$$

$$r = H\alpha^{T}$$

where H $\epsilon R^{d^w * T}$, d^w is the dimension of the word vectors, w is a trained parameter vector and w^T is a transpose. The dimension of w, α, r is d^w, T, d^w separately.

The final sentence-pair representation used for classification will be obtained from:

$$h^* = \tanh(\mathrm{r})$$

5.2.2. Output layer

Softmax classifier will be used to predict the class label. Softmax activation produces the probability of each class. The number of hidden units in the output layer is equal to the number of classes. In this case, the number of hidden units is 3 since there are three classes. Each activation unit calculates the probability of its respective class and the sum of all elements shall be one[20].

The standard softmax function is defined by the following formula

$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$ for i = 1,……….,k and z = ($\boldsymbol{z_1}$,…,$\boldsymbol{z_k}$)

Apply the standard exponential function to each element z_i of the input vector z and normalize these values by dividing by the sum of all these exponentials.

5.3. BERT

BERT (Bidirectional Encoder Representations from Transformers) is released in late 2018. BERT has been developed by Google research team which has proved to be the best of the model which are available into the market. BERT has been designed to pre-train deep bidirectional representations from the unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.BERT's key technical innovation is to apply the attention model called Transformer[21].

BERT has the following two steps:

- **Pre-training**: During pre-training, the model is trained on unlabeled data over different pre-training tasks. Pre-training is very expensive, but it is a one-time procedure for each language. Google released a number of pre-trained models. GluonNLP has a list of pre-trained BERT models. We can load the pre-trained BERT using the model API in GluonNLP, which returns the vocabulary along with the model.
- **Fine-tuning**:BERT model is first initialized with pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters.

5.3.1. Architecture of BERT

BERT is a multi-layer bidirectional Transformer encoder. There are two models introduced in the paper[3].

- BERT base: 12 layers(transformer blocks), 12 attention heads, 768 hidden units,and 110 million parameters

- BERT Large: 24 layers, 16 attention heads, 1024 hidden units and 340 million parameters

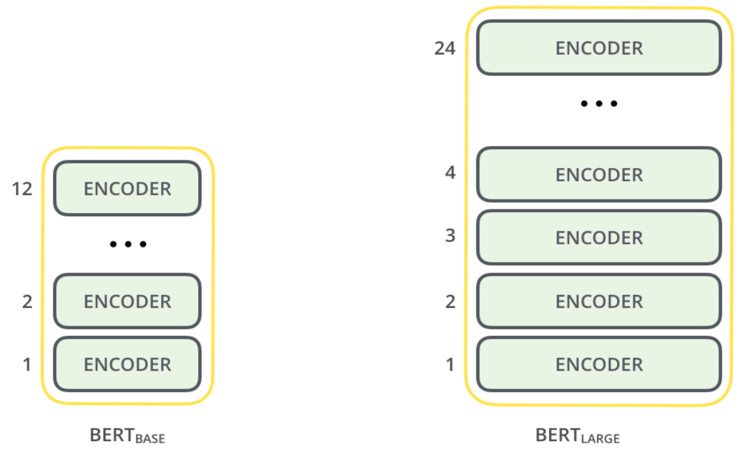


Figure 17: BERT Architecture**[22]**

5.3.2. Pre-training BERT

Defining a prediction goal when training a language model is challenging.Many models predict the next word in the sequence, a directional approach that inherently limits context learning. To avoid this challenge, BERT uses two training strategies: "Masked LM" and "Next Sentence Prediction"[3].

5.3.2.1. Masked LM (MLM)

Language Modeling is the task of predicting the next word given a sequence of words. In masked language modeling instead of predicting every next token, a percentage of input tokens is masked at random and only those masked tokens are predicted.Bi-directional models are more powerful than uni-directional language models. But in a multi-layered model bi-directional models do not work because the lower layers leak information and allow a token to see itself in later layers. This is the reason to use a 'masked language model' over standard language modeling[3].

The masked words are not always replaced with the masked token – **[MASK]** because then the masked tokens would never be seen before fine-tuning. Therefore, 15% of the tokens are chosen at random and

- 80% of the time tokens are actually replaced with the token [MASK].
- 10% of the time tokens are replaced with a random token.
- 10% of the time tokens are left unchanged.

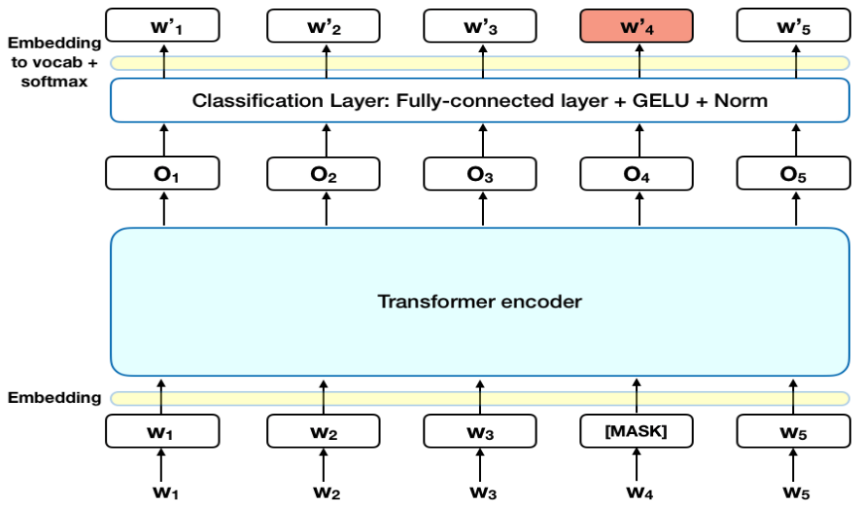

Embedding to vocab + softmax
w'1
w'2
w'3
w'4
w'5
Classification Layer: Fully-connected layer + GELU + Norm
O1
O2
O3
O4
O5
Transformer encoder
Embedding
w1
w2
w3
[MASK]
w5
w1
w2
w3
w4
w5

Figure 18: Masked LM**[21]**

5.3.2.2. Next Sentence Prediction (NSP)

Next sentence prediction is a binary classification task in which, given a pair of sentences, it is predicted if the second sentence is the actual next sentence of the first sentence.It is helpful because many downstream tasks such as Question and Answering and Natural Language Inference require an understanding of the relationship between two sentences[3].

5.3.3. Tokenization strategy in BERT

BERT uses WordPiece tokenization[6]. The vocabulary is initialized with all the individual characters in the language. For example given the text “Here is the sentence I want embeddings for”. The word “embedding” is represented as

['em', '##bed', '##ding', '##s']

The word ‘Embedding’ has been split into smaller subwords and characters. The two hash sign preceding some of these subwords denotes that this subword or character is part of a larger word and preceded by another subword.BERT model creates a fixed-size vocabulary of individual characters, subwords, and words that best fits the language data. The vocabulary contains four things[6]:

1. Whole words
2. Subwords occurring at the front of a word or in isolation("em" as in "embeddings" is assigned the same vector as the standalone sequence of characters "em" as in "go get em"
3. Subowords not at the front of a word, which is preceded by '##' to denote this case
4. Individual characters

To tokenize a word under this model, the tokenizer first checks if the whole word is in the vocabulary. If not, it tries to break the word into the largest possible subwords contained in the vocabulary, and as a last resort will decompose the word into individual characters. Note that because of this, we can always represent a word as, at the very least, the collection of its individual characters.As a result, rather than assigning out of vocabulary words to a catch-all token like 'out of vocabulary (OOV)' or 'UNK,' words that are not in the vocabulary are decomposed into subword and character tokens that we can then generate embeddings for[6].

5.3.4. Fine-Tuning procedure

There are different fine-tuning procedures for each downstream tasks. The fine-tuning for the classification task will be done in the following way. The final hidden state of the CLS token is taken as the fixed-dimensional pooled representation of the input sequence.This will be fed to the classification layer. The classification layer is the only new parameter added and has the dimension of K*H, where K is the number of classifier labels and H is the size of the hidden state. The label probabilities are computed with a standard softmax.

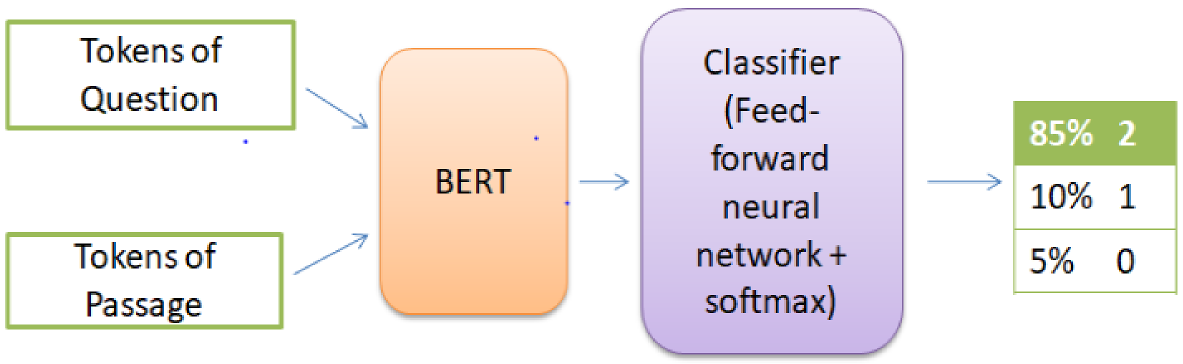


Figure 19: Fine-tuning procedure for classification

5.3.5. Transformer

Before discussing the input and output of the model, it is necessary to understand the Transformer. The explanation about the transformer is taken from[23] blog. The transformer is introduced by ‘Attention is all you need’ paper by google[24]. The transformer is an evolution of encoder and decoder architecture. BERT is basically a trained Transformer Encoder stack(only encoder layer of the Transformer is used in the BERT).

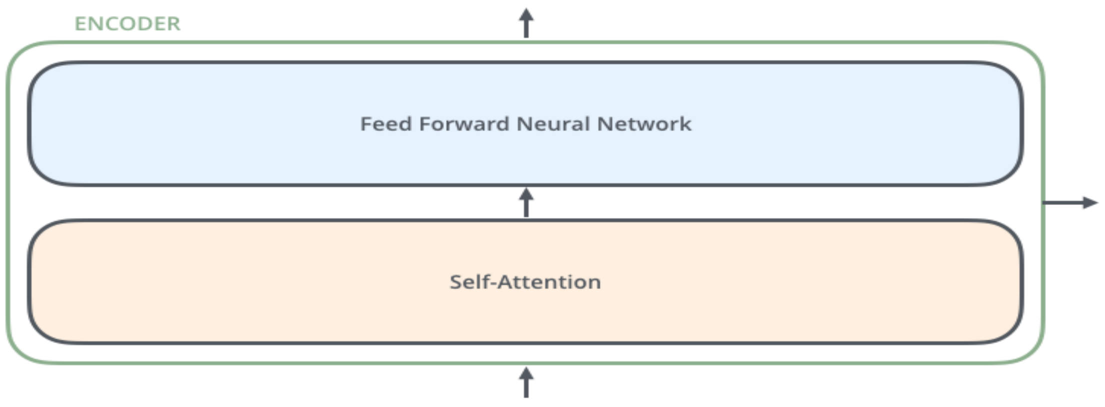


Figure 20: Encoder architecture**[22]**

According to the paper[24], the transformer has six layers of encoders(one can also experiment with other arrangements). All encoders are identical in structure. Each encoder is broken down into two sub-layer those are feed-forward neural network and Self-Attention. The encoder receives a list of vectors as input. It processes this list by passing these vectors into a ‘self-attention’ layer, then into a feed-forward neural network, then sends out the output upwards to the next encoder.

The Output of the self-attention layer is fed to a feed-forward neural network. The bottom-most encoder is responsible for creating the embeddings. Other encoders receive a list of vectors each of the size 512[22]. The key property of the transformer is that the word in each position flows through its own path in the encoder. In the self-attention layer, there will be dependencies between the paths. But these dependencies are not there in the feed-forward layer and thus the various path can be executed in parallel while flowing through the feed-forward layer.

5.3.5.1. Self-Attention layer

This layer helps the encoder look at other words in the input sentence as it encodes a specific word. For example the sentence “The animal didn’t cross the street because it was too tired”. How does the algorithm understand the subject in the sentencei.e, ‘it’ refers to the ‘animal’? When the model is processing the word “it”, self-attention allows it to associate “it” with “animal”[24]. As the model processes each word, self-attention allows it to look at other positions in the input sequence for clues that can help lead to a better encoding for this word.

α

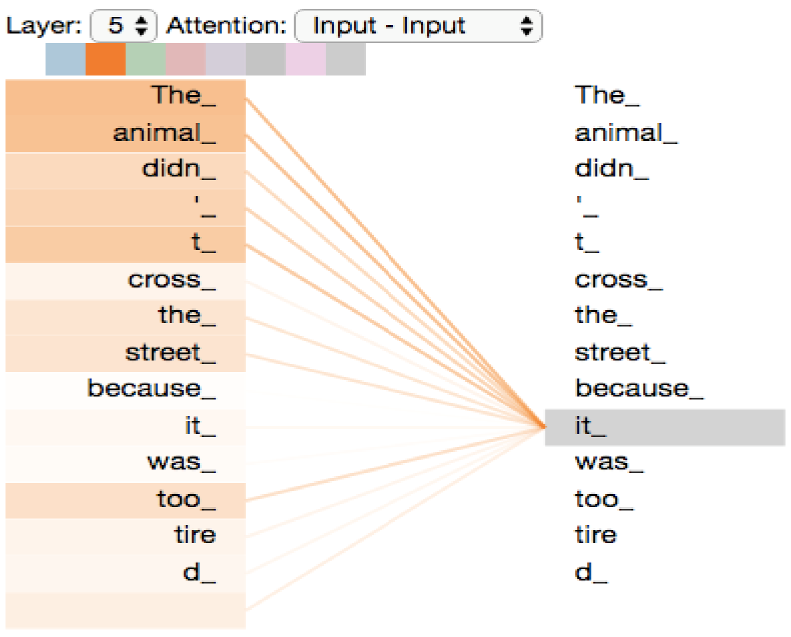


Figure 21: Part of the attention mechanism was focusing on "The Animal", and baked a part of its representation into the encoding of "it"**[22]**

The following are the steps to calculate self-attention using vectors.

1. Three vectors from each encoder's input vector will be created. For each word a Query vector, Key vector and Value vector will be created. The vectors will be created by multiplying the embedding by three matrices that are optimized during the training process.

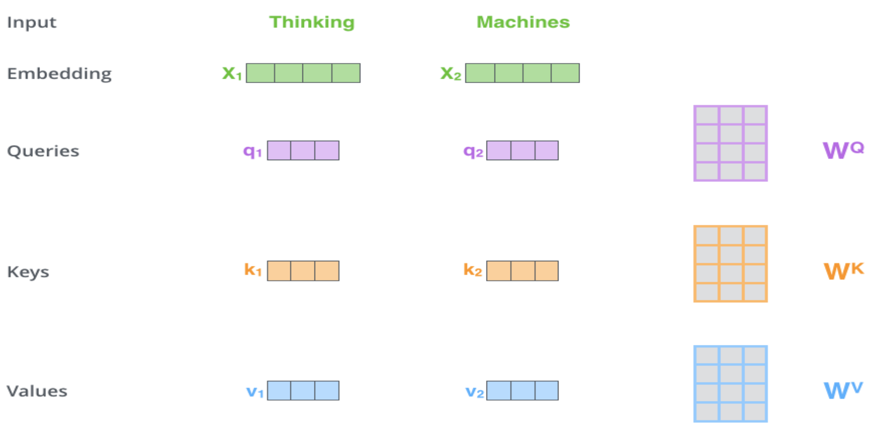


Figure 22: Creating a "query", a "key", and a "value" projection of each word in the input sentence**[22]**

2. The score will be calculated by the dot product of the query and the transpose of the key. The score determines how much focus to place on other parts of the input sentence.

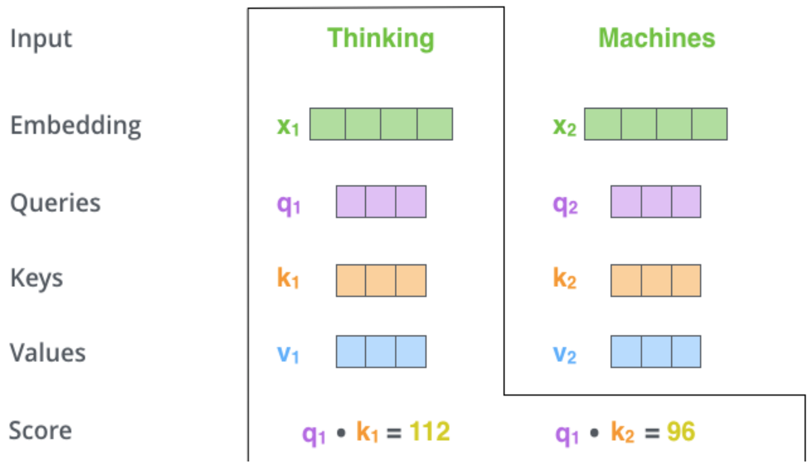


Figure 23: Calculating self-attention for the word 'Thinking'[22]

3. The scores will be divided by 8 (the square root of the dimension of the key vectors used in the paper – 64), then the result through a softmax operation will be passed. Softmax normalizes the scores so they're all positive and add up to 1.

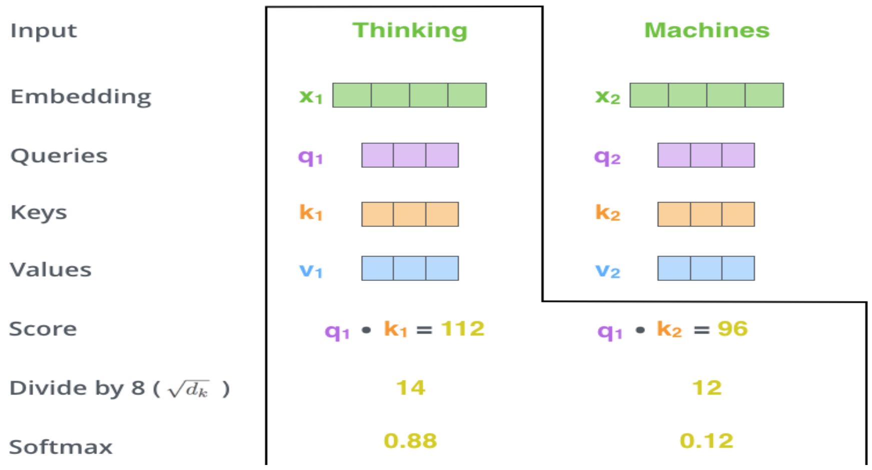


Figure 24: Softmax score determines how much the word 'Thinking will be expressed at this position. [22]

4. Each value vector by the softmax score will be multiplied. The intuition here is to keep intact the values of the word we want to focus on and remove the irrelevant words (by multiplying them by tiny numbers like 0.001, for example).
5. The weighted value vectors will be summed up. This produces the output of the self-attention layer at this position (for the first word).

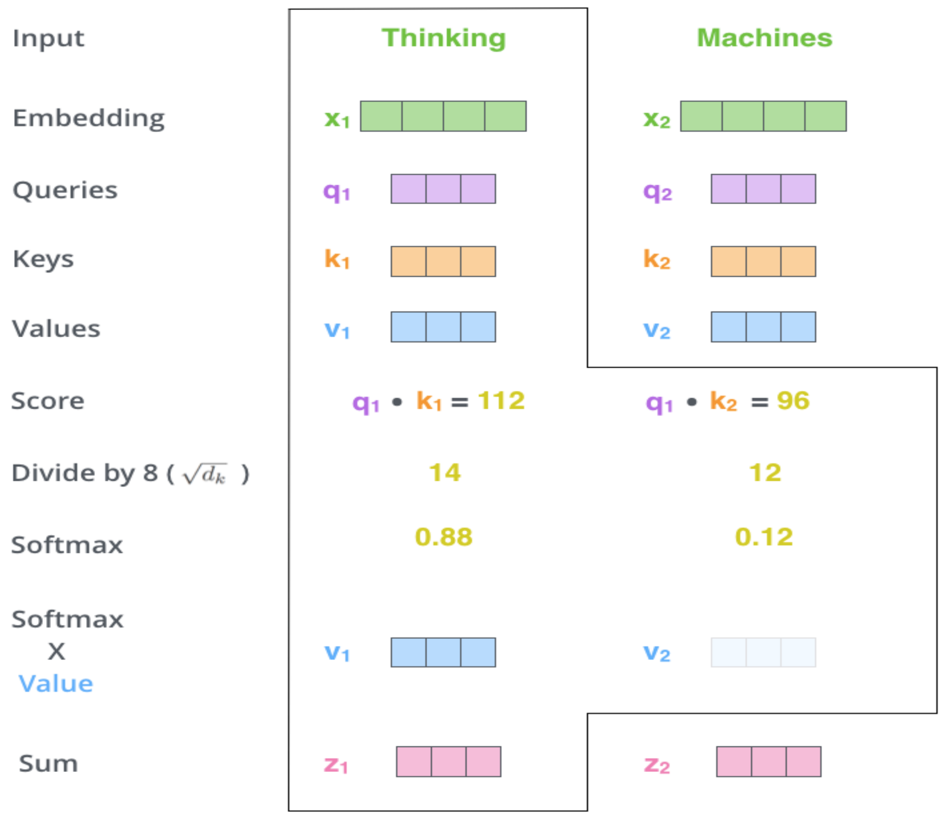


Figure 25: The output of the self-attention layer for the word 'Thinking'

5.3.6. BERT model input

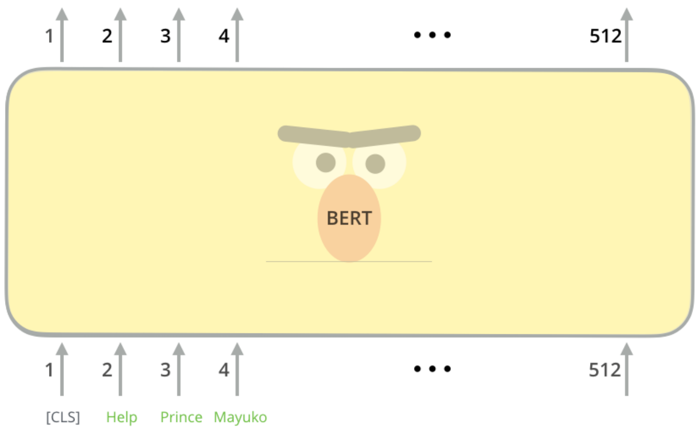


Figure 26: BERT input[22]

The first input token is supplied with a special CLS token which stands for classification. The model is similar to the encoder of the transformer, BERT takes a sequence of words as input which keeps flowing up the stack. Each layer applies self-attention and passes its results through a feed-forward network and then hands it off to the next encoder[22].

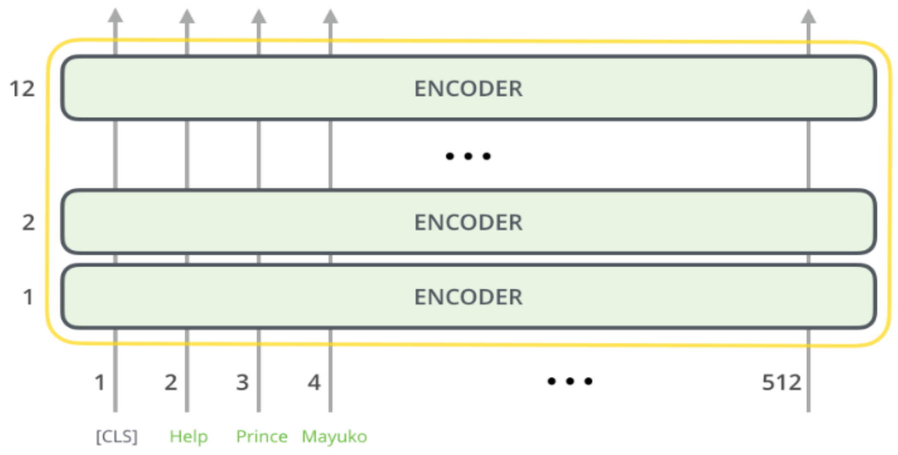


Figure 27: Stack of encoder layer[22]

5.3.7. Model Output

Each position outputs a vector of hidden_size. In this task, we will only focus on the first position(the [CLS] token). That vector will be used as the input for a classifier(i.e. Softmax classifier).

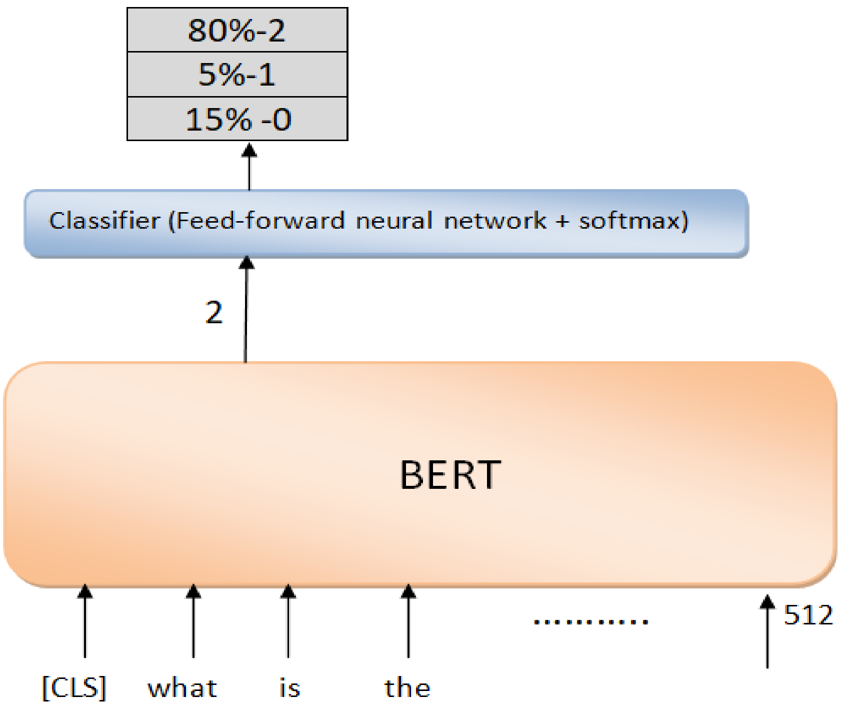


Figure 28: Model Output

5.4. Evaluation Metrics

The evaluation metrics are used to measure the performance of the model. The metrics used in this task are the Precision, Recall and F1 score.In TREC Genomics dataset classes are not distributed evenly.In such a case,the F1 measure works better than accuracy. Before understanding themetrics following definitions needs to be known.

1. **True Positives**: The total number of accurate predictions that were "Positive". In this case, this is the total number of correctly predicted paragraphs which are "Definitely Relevant" and "Possibly Relevant"
2. **False Positives:** The total number of inaccurate predictions that were "Positive". In this case, this is the total number of incorrectly predicted paragraphs which are "Definitely Relevant" and "Possibly Relevant"
3. **True Negative**: The total number of accurate predictions that were "Negative". In this case, this is the total number of correctly predicted paragraphs which are "Not Relevant"
4. **False Negative:** The total number of inaccurate predictions that were "Negative". In this case, this is the total number of incorrectly predicted paragraphs which are "Not Relevant"
5. **Confusion Metrics:**A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known.

		Predicted Class		
		Class=0	Class=1	Class=2
Actual class	Class=0	TP_0	E_{10}	E_{20}
	Class=1	E_{01}	TP_1	E_{21}
	Class=2	E_{02}	E_{12}	TP_2

Table 12: Confusion Metrics

As shown in the confusion matrixTP_0 is the number of true positive samples in class 0, i.e. the number of classes that are correctly classified from class 0 and $E_{10}$are the samples from class 1 that were incorrectly classified as class 0.

Precision:Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answers is, of all paragraphs that labeled as relevant, how many are actually relevant? High precision relates to the low false-positive rate.

Precision = TP/TP+FP

Recall: Recall is the ratio of correctly predicted positive observations to all observations in an actual class. The Recall answers, of all the paragraphs that are truly relevant, how many did we label?

Recall = TP/TP+FN

F1 score: F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into the account. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

6. Experiments

In this section, experiments will be conducted on the TREC Genomics dataset. The dataset is imbalanced and contains three classes as explained in the section 3. There are 25,764 passages and the dataset is divided into 60 % training and 40% testset.

Data	Not	Possibly	Definitely
Training data	13638	749	1071
Test data	9081	471	754
Total	**22719**	**1220**	**1825**

Table 13: Statistics of training and test data

The evaluation metric will be adopted to evaluate the model, which is based on a weighted F1-score for three classes. Experiments will be conducted using the baseline Manhattan LSTM model, LSTM with attention and BERT model. The performance of each model will be demonstrated and results are compared. The Performance of the model on various range of training data and different paragraph lengths will be discussed. Parameter selection for each model will be done by cross-validation.

Weighted F1-score: Weight the F1-score of each class by the number of samples from that class.

6.1. Baseline Model: Manhattan LSTM model

In the Manhattan LSTM network, Pubmed pre-trained embeddings will be used to initialize the embedding layer(200-dimensional). The input to this model is embeddings of Questions and answers without any contextual information. The following table presents a hyperparameter choice for themodel. Dropout will be applied to the word embeddings. The ‘Adam’ optimizer will be used to train the model.

Hyperparameter	Choice	Experiment values
Learning rate	0.01	0.1, 0.01, 0.001
Epochs	5	3,5,10
Batch size	16	8,16,32

Table 14: Parameter choice for baseline model

Learning rate: The amount that weights are updated is controlled by a configuration parameter called the learning rate.

Epochs: One epoch is when an entire dataset is passed forward and backward through the neural network only once.

Batch size: The batch size defines the number of samples that will be propagated through the network. If we have 1000 training examples, and the batch size is 500, then it will take 2 iterations to complete 1 epoch.

	Precision	Recall	F1-score	Support
Not	0.83	0.89	0.94	9081
Possibly	0.00	0.00	0.00	468
Definitely	0.00	0.00	0.00	757
Weighted avg	0.78	0.88	0.83	10306

Table 15: Performance of the baseline model

Class	0	1	2
0	9080	0	1
1	466	2	0
2	756	0	1

Table 16: Confusion metrics usingthe baseline model

Table 15 shows the results when no context information and no attention are used. Precision and Recall for minority classes (Possibly and Definitely relevant) are zero. The model predicts relevant classes as ‘NOT relevant’ which means relevant examples become false negative. This is due to the imbalanced classes. Though the overall weighted f1 score is 83 %, the individual precision and recall for relevant classes are zero. The performance is really poor on relevant classes and the model classifies everything as not relevant. This concludes that the baseline LSTM model without contextual information is not suited for long distant dependencies because LSTM can remember sequences of 100s, not 1000s or more[18].

6.2. Siamese LSTM with attention

As described in section 5.2, the method to incorporate context information is to use the attention mechanism to act on the preceding and current word. For this model, the same hyperparameters will be used as given in table 14. Table 17 summarizes the performance of the model using contextual information and attention mechanism. Following observations are made

1. The results of the QA classification using contextual information is quite better. There is performance degradation of 5% and this model outperforms the baseline model.
2. The precision and recall for an individual class are not impressive but better than the baseline model. Even though the attention mechanism has been applied to the LSTM, the model still gives more false-negative and fails to classify most of the relevant passages and to capture long distant dependencies.

	Precision	Recall	F1-score	Support
Not	0.92	0.99	0.95	9081
Possibly	0.46	0.13	0.21	468
Definitely	0.56	0.31	0.40	757
Weighted avg	0.85	0.88	0.86	10306

Table 17: Performance of the Siamese LSTM with attention

Class	0	1	2
0	**8962**	55	68
1	336	**61**	65
2	469	48	**234**

Table 18: Confusion metrics using LSTM attention

6.3. BERT

The list of pre-trained BERT models is available in GluonNLP. The BioBERT pre-trained model will be used here. The model was fine-tuned on a Cloud TPU, which has 64GB of RAM. The out of memory is the main issue if we use the same hyperparameters. Hence it is necessary to adjust a few parameters inorder to solve memory issues.

Hyperparameter	Choice	Experiment values
Learning rate	3e-5,	5e-5, 3e-5, 2e-5
Epochs	3	3, 4
Batch size	16	16, 32
Dropout	0.1	0.1, 0.01

Table 19: Hyperparameters choice for BERT

The model will be trained with Adam optimizer and trained for 3 epochs because more training will cause the model to overfit on the training data. BERT has a constraint on the maximum length of a sequence after tokenizing. For any BERT model, the maximum sequence length after tokenization is 512. But we can set any sequence length equal to or below this value. A bigger number gives better results for longer sequences. Due to GPU memory constraints, the max input sequence length is reduced to 256.

	Precision	Recall	F1-score	Support
Not	0.96	0.98	0.97	9968
Possibly	0.45	0.37	0.40	481
Definitely	0.75	0.67	0.71	751
Weighted avg	0.91	0.92	0.91	11200

Table 20: Performance of the BERT model

Class	0	1	2
0	**9739**	159	70
1	198	**178**	105
2	151	94	**506**

Table 21: Confusion metrics using BERT

The table 20 and 21 summarizes the results of the pre-trained BERT model. The following observations are made.

1. The pre-trained BERT for TREC QA is more effective and outperforms the previous two models. There is performance degradation of 4%.
2. Precision and Recall for each class have been improved compared to the other two models. The BERT model is able to classify the relevant passages correctly thus reduces the false-negative as shown in Table 21. This proves that the BERT is better in capturing long distant dependencies in texts due to the transformer's self-attention mechanism.

6.4. Analysis

To further understand the models, we break down their performance with respect to the length of passages, various training data, and type of questions.

6.4.1. Various training data

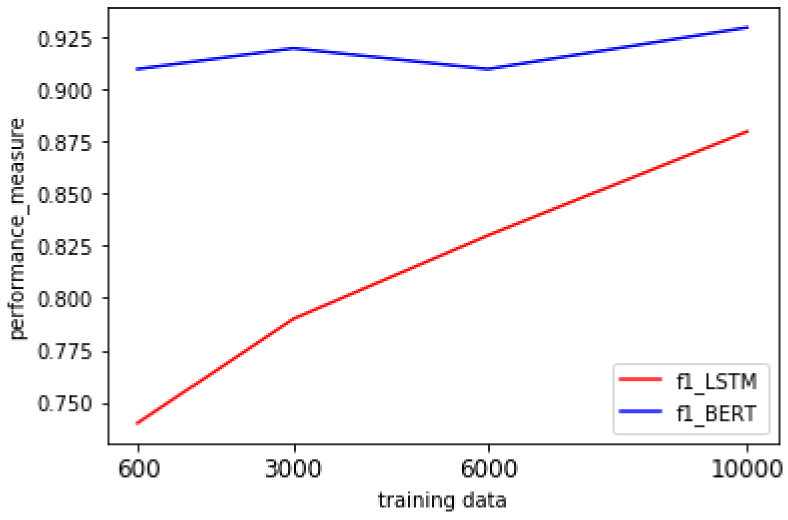


Figure 29: Performance of the model on various training data

The figure summarizes that the deep learning model should be trained with a large amount of data inorder to improve the performance. Increasing training data always add information and should improve performance. But in the real world, we also deal with less amount of data, the model should perform well on any size of data. In this case, the pre-trained model is more effective than a deep learning model. The pre-trained BERT model performs better even using the small amount of training data and the F1 score is close for different data sizes.

6.4.2. Paragraph length

Since the LSTM fails to capture long distant dependencies, the performance of the LSTM attention model goes down when the length of passage becomes longer. But the performance is effective for the smaller length as shown in figure 28.

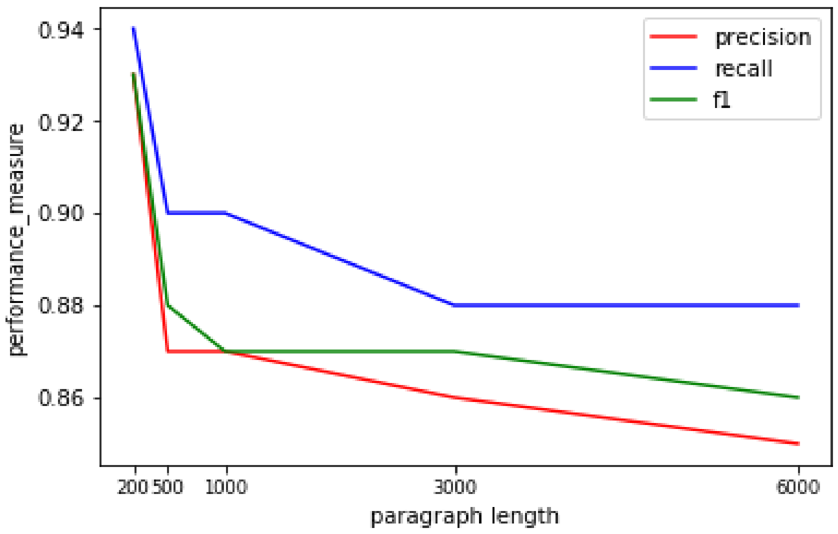


Figure 30: Performance of LSTM-attention model on different paragraph length

The performance of the BERT model on different paragraph length is shown in figure 29. Both models work well for smaller lengths. Infact, the performance of both models is closer for small sequence length. The BERT model's performance goes down when the length of the correct answers becomes longer. For longer answers, it can still maintain F1 scores at around 90%. This shows BERT's ability to capture long-distance dependencies with its deep and large-scale self-attention layers. This is one of the major reasons why BERT out-performs the model which has been trained from scratch.

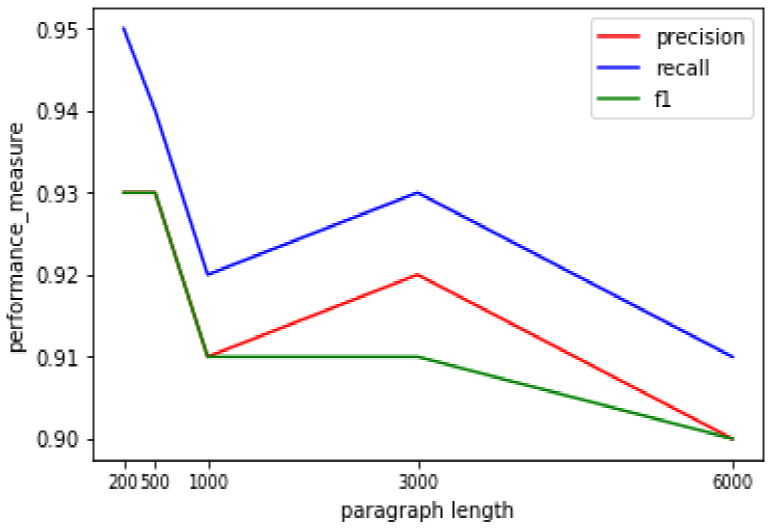


Figure 31: Performance of BERT model on different paragraph length

6.4.3. Question patterns

In this section, we provide the results of the models in terms of the different question patterns.The TREC Genomics data contains only three types of questions.

- What is the role of gene in disease?
- How do genesinteract in organ function?
- How does a mutation in gene influence biological process?

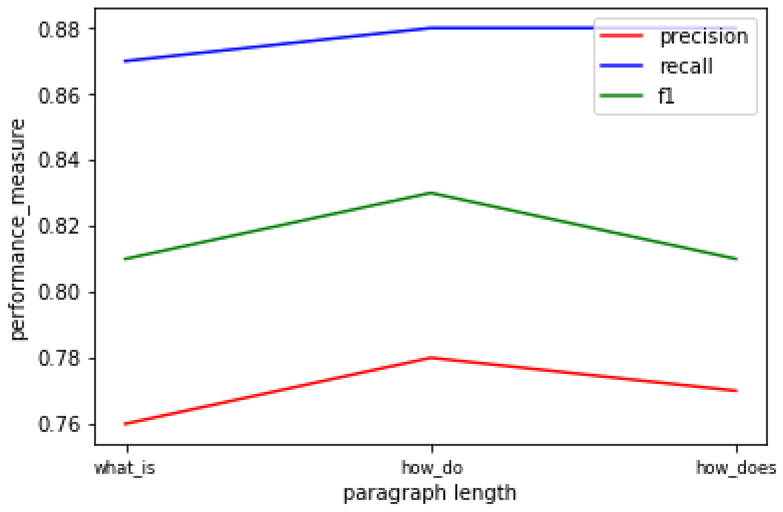


Figure 32: Result of 'LSTM with attention' model on different question patterns

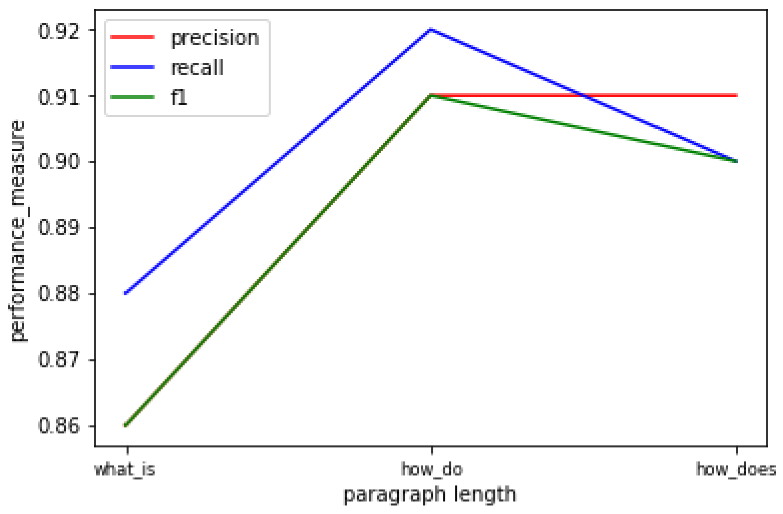


Figure 33: Result of BERT model on different question patterns

We expected the ‘how’ type questions are hardest since they involve more reasoning and have a longer answer. The performance of the LSTM model on all type of questions is almost closer. But for the BERT model ‘what’ question patterns are quite difficult to classify compare to ‘how’.Still, we can not judge by taking only two types of factoid questions. Experimenting these models on different factoid questions like ‘when’, ‘why’ so onwould give a better understanding. According to [14] the BERT models are better at ‘when’ questions as time-related patterns are easier to spot but ‘why’ type of questions are hardest for the model since they have a longer answer.

7. Conclusion and Future Work

The goal of the thesis is to apply a deep learning model and a fine-tuning pre-trained language model to find the relationship between questions and paragraphs. The goal also covers the usage of these models to classify the new passages according to their degree of relevance by just adding an additional classification layer.

The work started with the basic LSTM model to classify the new passages. Although LSTM has access to both past and future context information,the model fails to capture long distant dependencies due to vanishing gradient problem. The relevant passages are not classified correctly thus it led to more false-negative as shown in table 16.

To avoid this issue wehave built a deep learning model 'Siamese LSTM with attention', where themodel has been trained from scratch.The attention layer helps to relate the current word in the paragraph with all other words. The performance of the model is better than the basic LSTM model. The attention layer tried to read the longer sequence of passages and classified thepassages correctly and reduced the number of false-negative.The f1 score of this model is pretty good. The model performs well on a smaller length of passages as shown in figure 28. As the length of the passages increases the performance goes down. Still, the model maintains the f1 score between 0.80 to 0.85 for a longer sequence.But the model needs to be trained with a large amount of data inorder to capture the information and classify the new passages correctly.With a small amount of data, the model tends to overfit.

In order to solve this issue,we have applied the pre-trained BERT model to classify the passages. The analysis of the results shows that BERT is better at capturing long distant dependencies. The F1 score obtained on TREC Genomics shows an impressive improvement over the state of the art. The major reason why BERT out-performs the deep learning model is due to theself-attention mechanism of the transformer. In order to classify the passages correctly, the first thing is to generate accurate embeddings of each word in the text. Self-attention provides embedding by reading text from both directions. And also the size of the training data will not affect the performance of the BERT model and it maintains the F1score around 0.90 for longer sequences passages.More importantly, we can save time by not training the model from scratch.

The drawback could be we can only use the pre-trained model whenthere is a relation between our dataset and the pre-trained model. In this work, we have used BioBERT which has been trained on Pubmedwhich is suitable for our TREC Genomics dataset. But we can not use the pre-trained model if there is no relation between our data and the pre-trained model, so in such a case, we should train the model from scratch.

The other drawback of BERT is, It is restricted to the maximum tokens 512.The BERT would handle a paragraph length, how about the length more that 10,000 or more?

One of the most important areas to focus on as part of future work would be to handle the sequence length for large data.One way can be to split the text into multiple subtexts and apply a classifier on each of them and combine the results back together. The class which was predicted for most of the subtexts will be chosen.

Another important area of future work could be to use the ‘GRU layer’ instead of LSTM and create an attention layer on top of it.The performance of the LSTM with attention is also impressive but we can still improve the result by replacing LSTM with ‘Gated recurrent units’(GRU). GRU is simpler and thus easier to modify and does not need memory units therefore it is faster to train than LSTM as explained in [25].

Bibliography

[1] W. Hersh, A. M. Cohen, P. Roberts and H. K. Rekapalli, TREC 2006 Genomics Track Overview, vol. 2003, 01, pp. 14-23.

[2] J. Mueller and A. Thyagarajan, "Siamese Recurrent Architectures for Learning Sentences Similarity," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, 2018.

[3] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT:Pre-training of Deep Bidirectional Transformers for Language Understanding," vol. 2, pp. 1-16, 24 May 2019.

[4] T. Voorhees, "Building a Question Answering Test Collection," *National Institute of Standards and Technology,* pp. 1-9, 2000.

[5] Avinash, "Pre-Trained machine learning models vs models trained from scratch," 14 June 2019. [Online].

[6] C. McCormick, "BERT Word Embeddings Tutorial," 14 May 2019. [Online].

[7] P. Qi, "Answering Complex Open-domain Questions at Scale," 21 October 2019. [Online].

[8] P. Rajpurkar, J. Zhang, K. Lopyrev and P. Liang, "SQuAD:100,000+ Questions for Machine Comprehension of Text," vol. 1, pp. 1-10, 16 January 2016.

[9] Y. Yang, W.-t. Yih and C. Meek, "WikiQA: A challenge Dataset for Open-Domain Question Answering," vol. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, p. 2013–2018, 2015.

[10] Wikipedia contributors, "Text REtrieval Conference(TREC) Data," 9 April 2019. [Online]. Available: https://en.wikipedia.org/w/index.php?. [Accessed 2 January 2020].

[11] P. R. C. i. Umbert, "Factoid Question Answering for Spoken Documents," 2012.

[12] A. navalakha, "NLP-Question Answering System using Deep Learning," 15 May 2019. [Online].

[13] M. Feng, B. Xiang, M. Glass, L. wang and B. Zhou, "Applying deep learning to answer selection: A study and an open task," *IBM watson,* vol. 2, pp. 1508-1585, 2 October 2015.

[14] Z. Hu, "Question Answering on SQuAD with BERT".

[15] Shao and Zhou, "Question Answering on the SQuAD Dataset".

[16] A. Severyn and A. Moschitti, "Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks," pp. 373-382, August 2015.

[17] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," pp. 1735-1780, 1997.

[18] E. Culurciello, "The fall of RNN/LSTM," 13 April 2018. [Online]. [Accessed December 2019].

[19] P. Zhou, W. Shi, J. Tian, Z. Qi and B. Li, "Attention-based bidirectional long short-term memory networks for relation classification," pp. 207-212, 7 August 2016.

[20] T. Sterbak, "Guide To Multi-Class Multi-Label Classification With Neural Networks In Python," 8 November 2017. [Online].

[21] R. Horev, "BERT-State of the Art Language Model for NLP," 7 November 2018. [Online].

[22] J. Alammar, "The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)," 3 December 2018. [Online].

[23] J. Alammar, "The Illustrated Transformer," 27 June 2018. [Online].

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, "Attention is all you need," vol. 5, pp. 1-15, 6 December 2017.

[25] M. Nguyen, "Illustrated Guide to LSTM's and GRU's: A step by step explaination," 24 September 2018. [Online].

[26] Firiuza, "Optimizers for Training Neural Networks," Data Driven Investor, 28 July 2018. [Online].

[27] C. Hansen, "Activation Functions Explained- GELU, SELU, ELU, ReLU and more," 22 August 2019. [Online].

[28] J. A. Thom and F. Scholer, "A Comparison of Evaluation Measures Given How Users Perform on Search Tasks," pp. 1-4, 10 December 2007.

[29] R. Cai, X. Zhang and H. Wang, "Bidirectional Recurrent Convolutional neural network for Relation Classification," *Association for computational linguistics,* pp. 756-765, 12 August 2016.