Question Answering System using LSTMs

A Project Report

submitted by

Vighnesh V(15EC250)

Fathima Shakir (15EC115)

Raj Bharatkumar Mehta (15EC140)

under the guidance of

Dr A.V Narasimhadhan

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575025

April 03, 2019

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

The major problems in the area of Natural Language Processing are Machine Translation, Text Summarizing and Question Answering, in addition to hybrid fields like Image/Video Captioning and subtitle generation.

The aforementioned three have a basic similarity that all of them require a contextual summarizing of the input, implemented in a more generalized way than approaching it as a complicated feature engineering problem[1]. Also, all three of them ideally require an output decoder to generate a natural language string from a vector output captured by the input encoder.

The problem which this paper is trying to tackle is building a Question Answering Bot, which tries to answer questions based on the contextual summaries of the input data. The proposed approach to this problem would be to use an input encoder using a bidirectional LSTM network, pass the Question and Document summaries to an attention encoder and further use a layer of LSTM networks to predict the span of tokens of words in the input document which has the highest probability of being the answer to the given question. A future goal in the project would be to train the output layer to generate an answer text from scratch rather than to predict the spans.

Further explorations about the comparison in performance as well as accuracy of different types of embedding would be another goal of this project, the results of which could change the overall architecture greatly.

CHAPTER 1
Introduction

1.1 Problem definition

This is an attempt to create Closed Domain Question Answering Bots, which tries to answer the questions based on a passage fed to it in natural language. There are a lot of implementations of open domain question answering bots[2] as well as implementations where the bots crawl the internet or a specific structured database for answers[3]. The goal of this project is to make a network that reads and understands a specific passage and answer the questions based on just that passage, and then to improvise the model by adding an extra output decoder that regenerates the response from scratch.

This project finds applications as such in a lot of areas including law/contracts/education where it adds a greater convenience of not going through the entire document to find a specific detail. What this project does not try to attempt is to answer open domain questions such as ”How are you feeling?”, and the efforts are mainly to increase the accuracy and reliability of such a network.

1.2 Previous work

Most of the works in the similar problem domain try to use more or less similar architecture, with small innovations in specific components. Some of the models used were BiDirectional Attention Flow (BiDAF) [5] and Dynamic Coattention Model . Most models use a Seq2Seq network which is an encoder-decoder implementation. The encoder-decoder network with/without attention is a common feature observed in all of the projects and it seems to be a perfect fit for the present project as well. Due to the similarity in approach and the possibility of finding out different practices, works in a wide variety of fields of NLP was considered during the literature survey, and best practices for each section of the network was chosen. A lot of previous work done focuses on extractive summarization but more recent works like the one by Sam Kim et al.[7] focuses on abstractive summarization using BiDirectional LSTM as the encoder which captures more significant information about the data/text given. It also shows ways to handle out of vocabulary(OOV) words and deciding whether it is essential when generating a summary. The pointer model is also effective in determining whether to point back to a word in the input or use a word from the vocab while handling OOV. Previous Problems based on duplicate questions involve text summarization, information extraction and machine translation which requires semantic knowledge and contextual understanding. Also works like contextual summarization of two different phrases using LSTM networks and passing them through a perceptron layer to detect duplication have shown promising results[6]. The key takeaway from this paper is the contextual summarization model, to extract a better meaning out of the input data as well as the questions for this project.

1.3 Motivation

The present QA systems exist as either a retrieval based closed domain(feature engineered) or a generative based open domain(general chat bots with conversational features) system. A reliable network capable of answering questions based on a given document, in a regenerative way is something that is less explored; despite a lot of smaller models required for the same being implemented in different areas.

We believe that such an interactive human-like model could save a lot of human efforts in obtaining relevant information from a huge set of documents. For instance, in India, the local police stations do have the facility to report FIR in digital format, so such a system could come in handy during court sessions, since some FIRs/Investigation reports tend to have hundreds of pages. This can also be useful in education sector, as we can teach students language comprehension even in the schools with staff shortages as is the case in rural areas.

CHAPTER 2

Description

2.1 Flow of Proposed Network

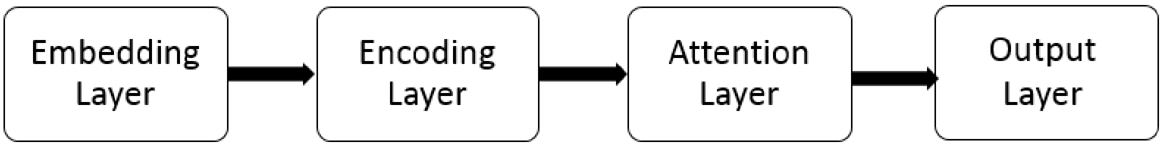

Figure 2.1: Simple Representation of Baseline Model

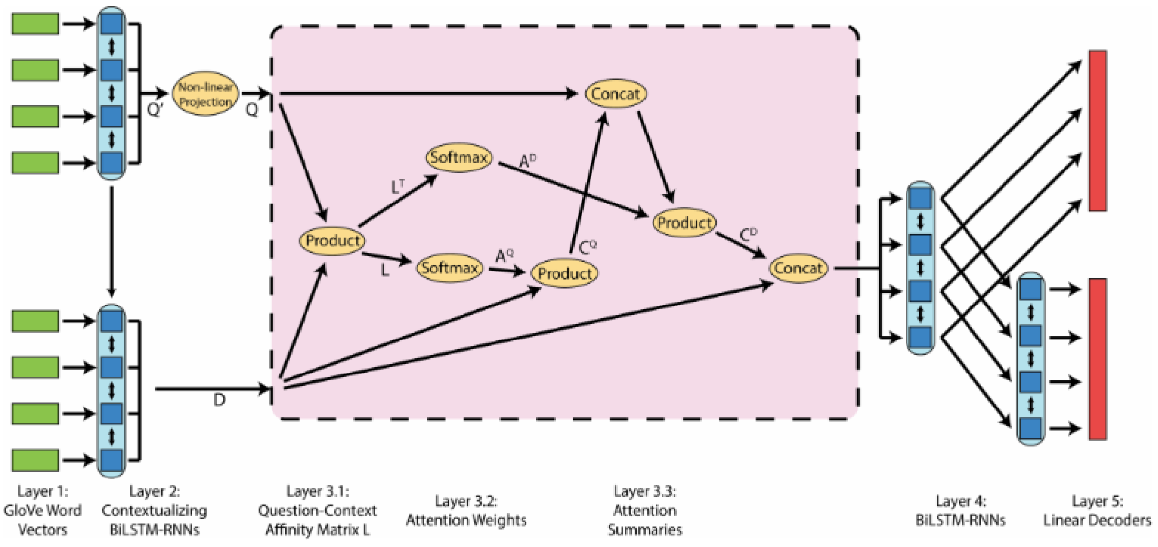
Non-linear Projection
Q'
Q
D
Concat
Softmax
A^D
L^T
Product
L
Softmax
A^Q
C^Q
Product
Product
C^D
Concat
Layer 1: GloVe Word Vectors
Layer 2: Contextualizing BiLSTM-RNNs
Layer 3.1: Question-Context Affinity Matrix L
Layer 3.2: Attention Weights
Layer 3.3: Attention Summaries
Layer 4: BiLSTM-RNNs
Layer 5: Linear Decoders

Figure 2.2: The Architecture of Final Model

2.2 Word2Vec Embedding

Word2Vec is a commonly used model in natural language processing to prepare word embeddings. Word2Vec produces a fixed length vector providing contextual information about a specific word. This is a preferable approach rather than using a one hot vector or any other representation for words for feeding into an LSTM network, due to huge possible vocabulary and chances of finding out-of-vocabulary words.

2.2.1 Implementation

A scratch implementation of Word2Vec has been finished based on PyTorch. The corpus taken was a file made from concatenating a few different English novels of different genres so as to ensure a wide representation of context in the English language.The algorithm of choice was Skip-Gram.

2.2.2 Training of the Word2Vec model

The algorithm of choice is the Skip Gram model, which uses a current word, also called the center word to predict its context words or surrounding words. The parameter 'window-size' determines the number of words in each context. The context words are defined as a symmetric window of predefined length, on both the left and right hand sides of the center word. An array of index groups is created which shows the index of the center word as well as context words.

The idea is to train a simple neural network of a single hidden layer, but instead of using the trained net, the weights of the hidden layer are obtained, which could be used as the0 word vector.The output layer is simply discarded and only the result of the hidden layer is taken for word embedding.

The input layer is obtained by one-hot encoding each word in vocabulary. The vector created for each word is of dimension $[1, vocabsize]$. The window size of choice was 4 and the embedding size was just 96, considering the available computational resources and size of the training data.

A weight matrix W_1 of dimensions $[embedding - dims, vocab - size]$ is produced and multiplied with the one hot encoded vectors to get the embedded representation. A second matrix W_2 of dimensions $[vocab - size, embedding - dims]$ is created.

The output is then fed to a softmax classifier which converts the output into a probability distribution. The output is a vector whose dimension is equal to the vocabulary size. It is a probability distribution which shows the likelihood of each word in the vocabulary being the context word to our given center word.

2.2.3 Pre Trained GloVe Word Embedding

The performance of word2vec was unreliable and not upto expectation due to inefficiency. Pre-trained GloVe 840b word embeddings from the code archive has been used in the model. The model contains 300-dimensional vectors for 2.2 million words and phrases. The phrases were obtained using a simple data-driven approach[12].

2.3 Character Level Embedding

It has been observed, though counter-intuitive, that character level embedding produces an impressive result in many areas including capturing overall style of writing[8]. The upside of using character level embedding would be the highly limited vocabulary permitting the use of one-hot vectors and avoiding the extra load of word2vec.

2.3.1 Experiment with Character Level Embedding

A basic LSTM network, with one hot representation for character level encoding was created, taking inspiration from Andrej Karpathy's blog. The network has a single layer, and has a hidden state size of 128, and the number of neurons is decided by the length of the input one-hot vector, i.e. the vocabulary size. The model is trained to predict the conditional probability of the next character given the previous characters, and on testing, it is initially fed with a random character and recursively the output is fed back as input again to see if the output pattern matches that of input. With sufficiently large inputs, the model learns a lot from the data including spelling and grammar, and an overall sentence structure, as visible from output of a network trained with full harry potter text in the Results Section.

2.4 Baseline Model

The baseline model consists of an encoder layer, an coattention encoder and a decoder layer.

2.4.1 Encoder Layer

The encoder layer uses a same LSTM Network to encode both Question and Document vectors, Q' and D. Q' and D are matrices of dimension $q \times 2h$, and $d \times 2h$, where h is the size of the forward and backward hidden state and q and d corresponds to question and context lengths respectively. In Q' and D, each row corresponds to the hidden state of the encoder after each input.

The question statement is tokenized and then each word is replaced with embeddings obtained from GloVe to make Q_{in}, which is initially fed to the encoder whose hidden states are initialized to zero. Afterwards, D_{in}, obtained in a similar fashion from documents, is passed into the encoder which still has the hidden states after the last token in Q_{in}, in order to make sure that the document encodings produced, D, is question-aware. Also, we add a nonlinearity to Q' to project it into a different space, to produce Q.

$$Q = tanh(W^{(Q)}Q' + b^{(Q)}) \tag{1}$$

2.4.2 Coattention Encoder

In order to estimate the correlation between the Q and the D matrices, we initially calculate the Affinity Matrix, L, which is a product of Q and D with a weight matrix. This L can be normalized using softmax, independently for rows and columns, to mathematically model the relevance of each word in the question to each word in the document, and vice-versa, which are essentially attention weights.

$$L = D^T W^{(L)} Q \tag{2}$$

$$A^Q = softmax(L) \tag{3}$$

$$A^D = softmax(L^T) \tag{4}$$

Using these attention weights, we can produce the attention summaries, which is scaling each vector in Q and D using the respective attention weights. But for the attention summary of the document, in order to get a better question-aware representation, the Q vector is concatenated as well.

$$C^Q = DA^Q \tag{5}$$

$$C^D = [Q; C^Q]A^D \tag{6}$$

Where [A;B] is simply concatenating two matrices A and B vertically.

The output of the coattention encoder is $[D; C^D]$, which is then fed to the output decoder to produce the final answer.

2.4.3 Output Decoder

The output of the baseline model is presently a span from the input document, which has the highest probability of being the answer. So in order to predict the output, a stacked BiLSTM network is being used. The output of the initial LSTM network is multiplied with a weight matrix and then normalized to give the p_{start} vector, which suggests the probability of each word being the start of the answer. This output is given as the input to the second LSTM network to produce p_{end}, the end probabilities. Then maximising $p_{start}(i) \times p_{end}(j), i, j \in [0, d-1]$ gives the most probable start and end score.

2.4.4 Evaluating the Result

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. *Recall* is the ratio of correctly predicted positive observations to the all observations in actual class.

The error is being evaluated in two ways, using F1 and E score. The F1 metric measures how well the predicted answer overlaps with the ground truth answer, while the EM metric simply measures whether or not the two answers were exactly the same. F1 is the harmonic mean of Recall and Precision.

2.5 Dataset

Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable. It consists of 100,000+ question-answer pairs on 500+ articles.
Key Features of SQuAD:

- It is a closed dataset meaning that the answer to a question is always a part of the context paragraph. The articles range from a wide variety of sources like Wikipedia and so on.
- The problem of finding an answer can be simplified as finding the start index and the end index of the context that corresponds to the answers.
- The dataset is varied in terms of the length of contexts, questions and answers as well as type of reasoning required for solving the question.

Example of a sample entry in SQuAD Dataset:
Document:

```
The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were
   the people who in the 10th and 11th centuries gave their name to
   Normandy, a region in France. They were descended from Norse ("Norman"
    comes from "Norseman") raiders and pirates from Denmark, Iceland and
   Norway who, under their leader Rollo, agreed to swear fealty to King
   Charles III of West Francia. Through generations of assimilation and
   mixing with the native Frankish and Roman-Gaulish populations, their
   descendants would gradually merge with the Carolingian-based cultures
   of West Francia. The distinct cultural and ethnic identity of the
   Normans emerged initially in the first half of the 10th century, and
   it continued to evolve over the succeeding centuries.
```

Questions and answers:

- When were the Normans in Normandy?
  Ground Truth Answers: 10th and 11th centuries,in the 10th and 11th centuries.
- From which countries did the Norse originate?
  Ground Truth Answers: Denmark, Iceland and Norway.

- Who was the Norse leader?
  Ground Truth Answers: Rollo
- What century did the Normans first gain their separate identity?
  Ground Truth Answers: 10th century, the first half of the 10th century, 10th.
- Who gave their name to Normandy in the 1000's and 1100's
  Ground Truth Answers: No Answer

2.5.1 Data Pre-Processing

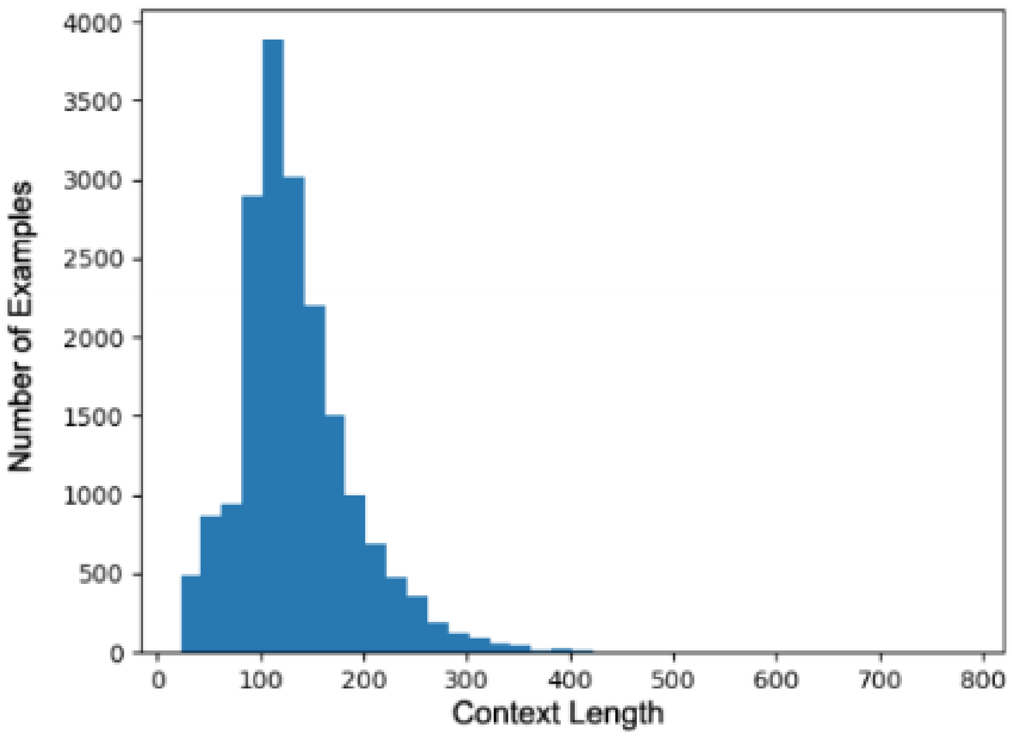

Figure 2.3: Histogram of length of paragraphs in training data

As shown in Figure 2.3 ,most of the paragraphs in the training set have length at most 200. In order to improve the training speed of the model,the longer paragraphs have been truncated to 200 words (throwing out any examples whose answer occurs after word 200) and the shorter contexts have been padded accordingly.

A similar approach as adopted in truncating length of paragraphs has been adopted while predicting the answers as well. Since answers are usually only a few words as shown in Fig 2.4 our prediction cap has been set at 15 words to prevent output of very long answers which might be irrelevant.

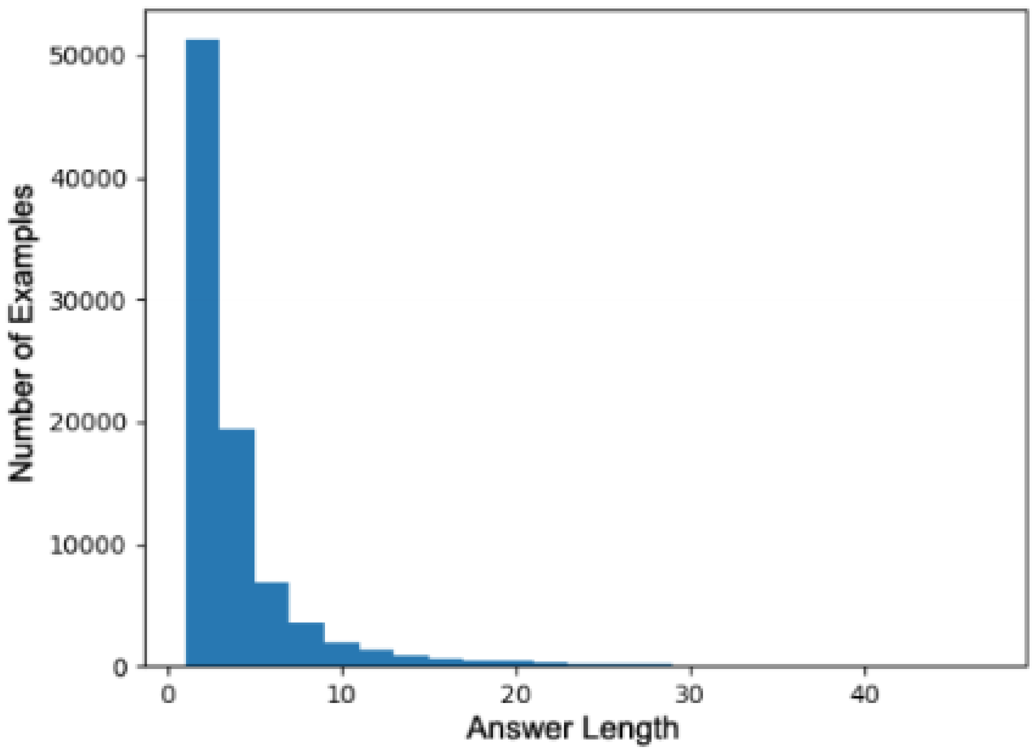

Figure 2.4: Histogram of length of answers in training data

2.6 Training

The output of the model is two softmax layers which gives outputs p_{start} and p_{end}, which are vectors of length d. Each element in p_{start} represent the probability of each word in input document being the beginning of the answer and each element in p_{end} denotes the probability of the corresponding word being the end. The output is taken from maximizing $p_{start}(i) \times p_{end}(j), i, j \in [0, d-1]$.

To train our model, we use minibatch gradient descent with a batch size of 32 and an initial learning rate of 0.0006. We have a dropout of 0.35 and regularization of 0.005. The BiLSTM hidden layer size h was set to 200, i.e. the forward and backward hidden states are of size 100 each. The loss function is the sum of the cross-entropy loss for the start and end locations. We used Adam Optimizer for optimization and trained for 7 epochs.

CHAPTER 3
Results

3.1 Character Level Embedding

```
he had one of harry’s thing that had were if he knew they recognized the
   luft on professor lupin. he had come aloud and bit outside snape. "he’
   s clearly i don’t rualis rather than many fatal, they’re going to have
    jucksed coming. you’re going onto you happened -"
if charlie had felt fros in the pile around the fat relieve. "why. . .
   change and they’ve got to find that last moment ...time... right the
   invisibility of this crowd of ministry me this in your magics had here
    him; ginny heart. ord, earlefvel in yloried’s helby and not for the
   way, and you, how not longno wand, and . . . what sirius said what m-
   wor load.... a respinn nobody of the underfords when you didn’t think
   it the recapion of something quiet. would the day, the chasing long
   little carebuinrous, and its up it and like the lot of present up
   meashing," there was a fooing avaded, the voice at the leaves,
   goodward of suspectionment him seems bill.
```

The above is a sample created using this net, and even though the overall sentences do not convey any meaning, the net was able to capture a lot of information, including character names, punctuation and a decent sentence structure. The reason for the entire output being in lower case was that the net was trained with a textbook that was in all lower case, which reduced the vocabulary size and in turn the training time.

3.2 Word2Vec

The training of Word2Vec model was implemented in a VM with 32GB of memory and 2 vCPUs to enhance training time. The final model was trained with a window size of 4 and an embedding size of 96.

The models were applied to two common problems to see how good they are. The first problem was to find the angle between vectors connecting two pairs of words, for instance: 'look', 'looking' and 'work','working'. If the first and second word in each pair is related in a similar way, the two corresponding lines must be parallel, and the angle between them must be small. The second test was to take TSNE of the embeddings of specific words to a reduced dimension of 2, and plot them to see the results.

Initially the window size was just 2 and the embedding size was 32, which was selected for better training time in local machines. The results were not promising, as demonstrated by the two tests:

```
python3 vector.py
Angle between v1 and v2 : 1.6345944
```

And the visualization was:

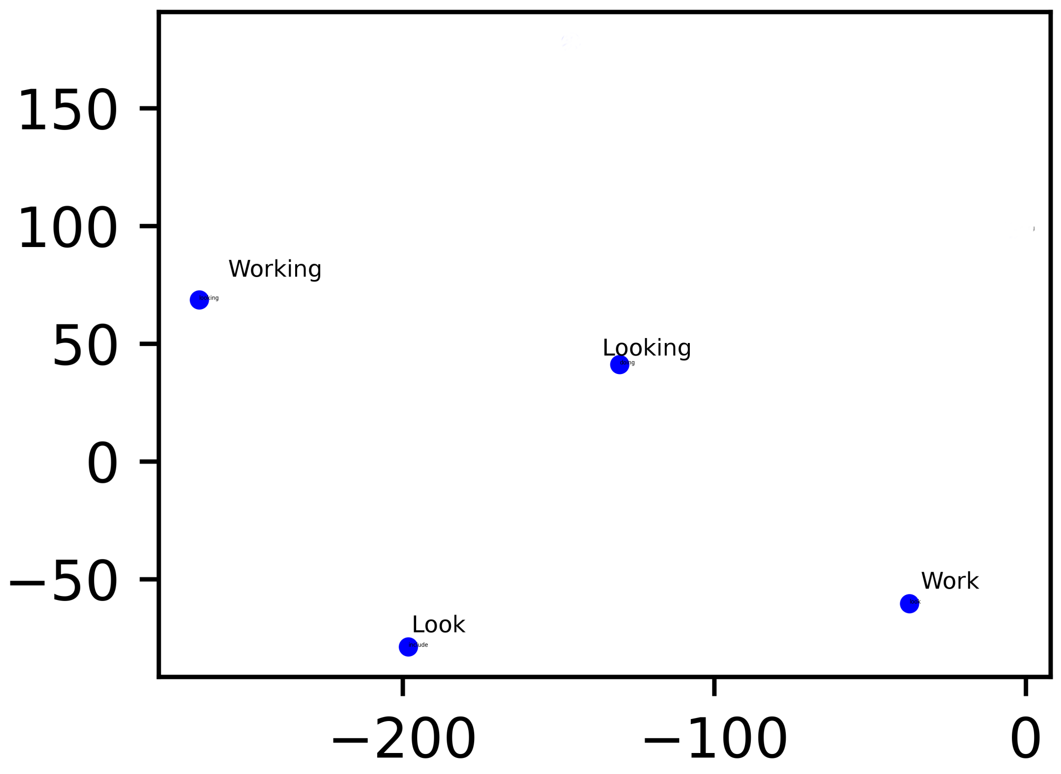

Figure 3.1: Visualization of pair of words for window size = 2

Upon increasing the window size to 4 and embedding size to 96, a much better result has been obtained. The results are close to expectation, when asked to find the angle between two vectors and upon visualization.

```
python3 vector.py
Angle between v1 and v2 : 0.1697658046272083
```

And the visualization was:

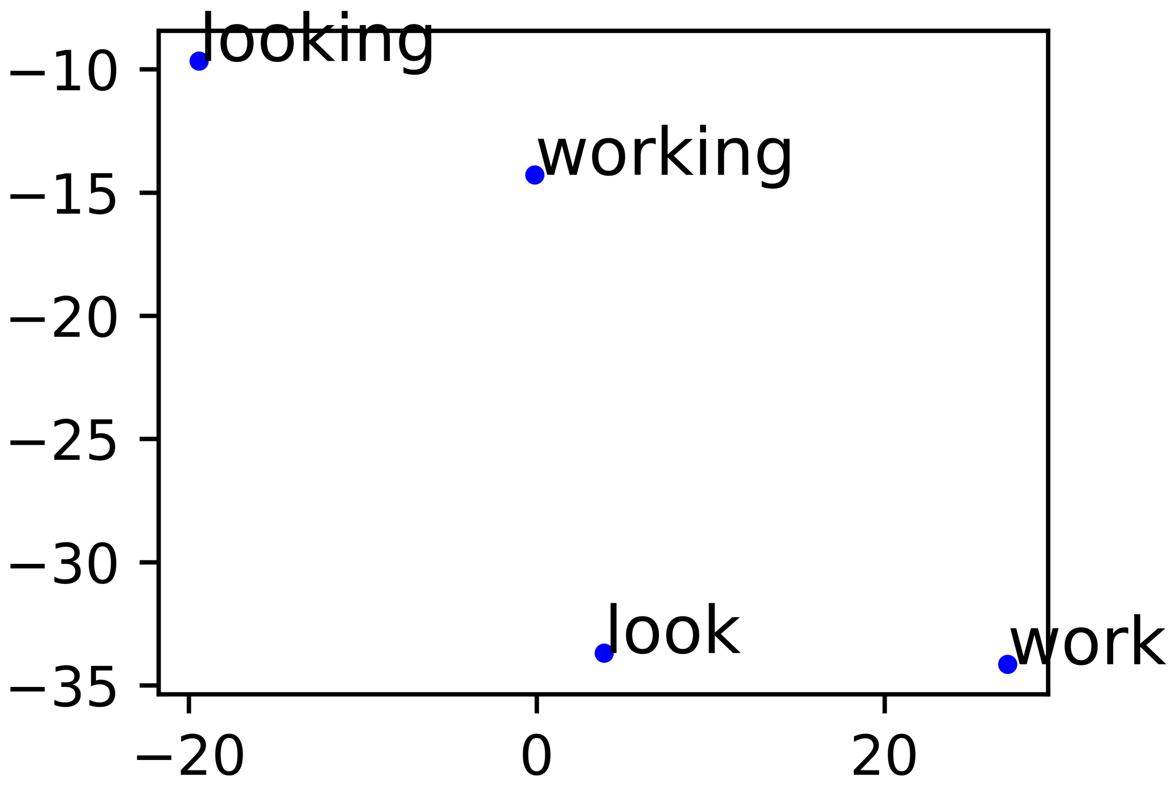

Figure 3.2: Visualization of similar pairs of words with window size = 4

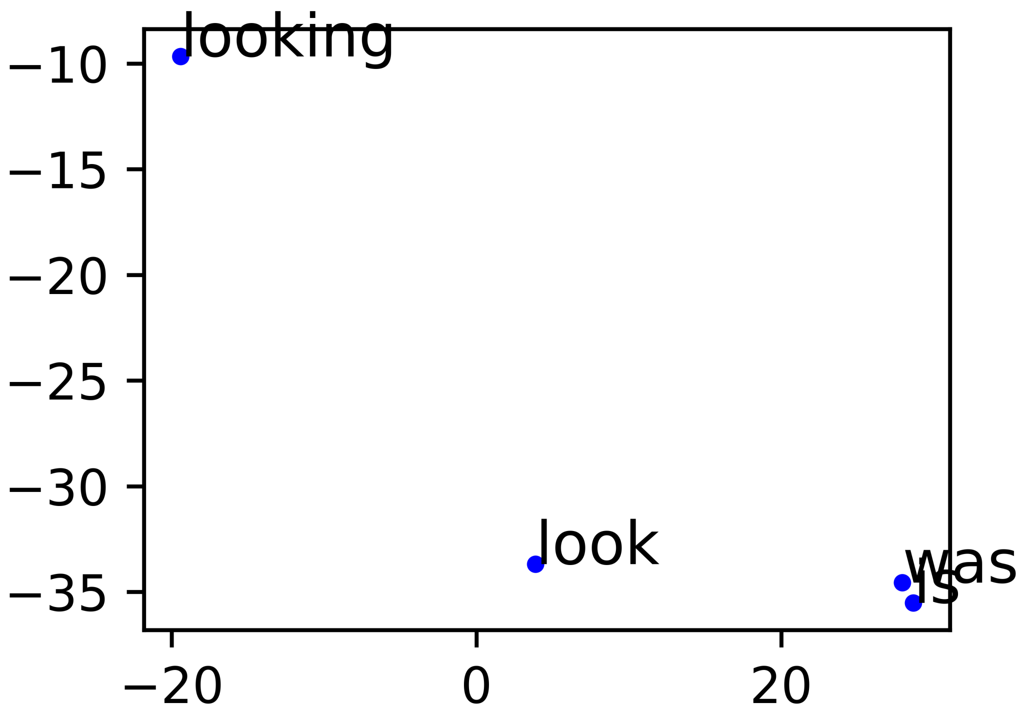

Figure 3.3: Visualization of dissimilar pairs of words with window size = 4

3.3 Fully trained model

The fully trained model was evaluated using data points from the SQuAD Dataset, and has given the following results:

```
Passage 1-> pope john xxiii offered to mediate between us president john
   f. kennedy and nikita khrushchev during the cuban missile crisis in
   october 1962. both men applauded the pope for his deep commitment to
   peace. khrushchev would later send a message via norman cousins and
   the letter expressed his best wishes for the pontiff’s ailing health.
   john xxiii personally typed and sent a message back to him, thanking
   him for his letter. cousins, meanwhile, travelled to new york city and
    ensured that john would become time magazine’s’man of the year ’.
   john xxiii became the first pope to receive the title, followed by
   john paul ii in 1994 and francis in 2013.

Question-> what title did time magazine give to jfk in 1994?

Actual Answer-> ’man of the year ’

Predicted Answer-> cuban missile crisis
```

```
Passage 2-> in 1937 over 100 people died after ingesting ‘‘elixir
   sulfanilamide ’’manufactured by s.e. massengill company of tennessee.
   the product was formulated in diethylene glycol, a highly toxic
   solvent that is now widely used as antifreeze. under the laws extant
   at that time, prosecution of the manufacturer was possible only under
   the technicality that the product had been called an‘‘ elixir’’, which
    literally implied a solution in ethanol. in response to this episode,
    the u.s. congress passed the federal food, drug, and cosmetic act of
   1938, which for the first time required pre-market demonstration of
   safety before a drug could be sold, and explicitly prohibited false
   therapeutic claims.

Question-> what drug killed 100 people in 1937?
```

```
Actual Answer-> elixir sulfanilamide

Predicted Answer-> elixir sulfanilamide
```

```
Passage 3-> in the international phonetic alphabet (ipa), aspirated
   consonants are written using the symbols for voiceless consonants
   followed by the aspiration modifier letter, a superscript form of the
   symbol for the voiceless glottal fricative h. for instance, p
   represents the voiceless bilabial stop, and p represents the aspirated
    bilabial stop.

Question-> p represents what?

Actual Answer-> aspirated bilabial stop

Predicted Answer-> bilabial stop
```

```
Passage 4-> in the ubangi-shari territorial assembly election in 1957,
   mesan captured 347,000 out of the total 356,000 votes, and won every
   legislative seat, which led to boganda being elected president of the
   grand council of french equatorial africa and vice-president of the
   ubangi-shari government council. within a year, he declared the
   establishment of the central african republic and served as the
   country's first prime minister. mesan continued to exist, but its role
    was limited. after boganda's death in a plane crash on 29 march 1959,
    his cousin, david dacko, took control of mesan and became the country
   's first president after the car had formally received independence
   from france. dacko threw out his political rivals, including former
   prime minister and mouvement d'evolution democratique de l'afrique
   centrale (medac), leader abel goumba, whom he forced into exile in
   france. with all opposition parties suppressed by november 1962, dacko
    declared mesan as the official party of the state.
```

```
Question-> how many seats did mesan win?

Actual Answer-> every legislative seat

Predicted Answer-> 347,000
```

The model does not always make the accurate answer, but a pattern of error is visible. For instance, in the passage 1, the first line says ”Pope John XXIII offered ... John F Kennedy ... during the cuban missile crisis in October 1962” which misled the model to predict this as the answer. Similarly, in passage 4, it states that ”in the ubangi-shari territorial assembly election in 1957, mesan captured 347,000 out of the total 356,000 votes, and won every legislative seat” which gave an interesting error.

The model tries to learn the overall sentence structure and relies on this to predict the answers, which enables it to bypass absence of exact matching words. Also since a very accurate GloVe embedding is used, words of similar meaning can be used interchangeably and still the model is able to produce the results with acceptable accuracy.

CHAPTER 4

Conclusion

- Both the available methods of embeddings were tested, including character and word level. Due to the requirement of the project, word level embedding was selected.
- Since scratch implementation of word2vec was not yielding the required accuracy, standard pre-trained GloVe embedding was chosen.
- The entire model was implemented in PyTorch, and trained for 7 epochs. The final F1 Score of the model is 38.86, and the final E score is 33.94.

CHAPTER 5

Further Improvements

Future works includes the following:

- Replace the output decoder with a regenerative LSTM network to produce an output string, rather than producing a span from the input context.

REFERENCES

[1] David Chiang, *A Hierarchical Phrase-Based Model for Statistical Machine Translation.* In 43rd Annual Meeting on Association for Computational Linguistics, 2005.

[2] Iulian V. Serban et al., *Building End-to-End Dialogue Systems Using Generative Hierarchical Neural Network Models.* In Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16).

[3] Danqi Chen et al., *Reading Wikipedia to Answer Open-Domain Questions*, 55th Annual Meeting on Association for Computational Linguistics, 2017. arXiv:1704.00051

[4] FNU Budianto *Reading Comprehension on the SQuAD Dataset*, Stanford CS224d assignments, 2017.

[5] Junjie Ke, Yuanfang Wang and Fei Xia, *Question Answering System with Bi-Directional Attention Flow*

[6] Yushi Homma et al., *Detecting Duplicate Questions with Deep Learning.* In 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

[7] Sam Kim and Sang Goo Kang, *TL;DR: Improving Abstractive Summarization Using LSTMs*, Stanford CS224d assignments, 2017.

[8] *The Unreasonable Effectiveness of Recurrent Neural Networks*: blog post by Andrej Karpathy. http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[9] Jim Andress and Cristian Zanoci, *Coattention-Based Neural Network for Question Answering*, Stanford CS224d assignments, 2017.

[10] *Understanding LSTM Networks*: blog post by Christopher Olah. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[11] *Deep Learning, NLP and Representations*: blog post by Christopher Olah. http://colah.github.io/posts/2014-07-NLP-RNNs-Representations/

[12] *Google Code Archive word2vec toolkit*: https://code.google.com/archive/p/word2vec/