Machine Comprehension Using BiDAF on SQuAD

Rahul Baid, Pietro Dominietto, Ruhani Suri
College of Computing, Georgia Institute of Technology
rbaid3@gatech.edu , pdominietto3@gatech.edu , rsuri8@gatech.edu

Abstract

The aim of the project is to build and train a model to perform contextual question answering, i.e. to be able to learn a text and then answer simple questions based on it. We have chosen to train a BiDAF model, using BERT and ALBERT as baselines. We are using the SQuAD 1.1 dataset for our training, using the approach of fine-tuning pre-trained models for the baselines. The model could have various practical applications in the field of large text comprehension. The code is hosted at the following link: cs4650-team13

1 Introduction & Related Work

The goal of the project is to train a model on the Standford Question Answering Dataset [SQuAD] v2.0 (2016). The project demonstrates the ability for a machine to comprehend an input text, model it in a way it can extract information from and then be able to answer questions about it. The comprehension of the question itself should be possible from the way the text is modeled. An acceptable goal will to be build and train a model with at least 2 baselines for comparison.

The original goal of the project was to train a model on our course textbook, Speech and Language Processing (Jurafsky and Martin, 2009) and then use the model to answer stack-overflow questions of similar topics. However, as we began collecting our training data we discovered the problems we had to face in cleaning and modeling it.

1. We discovered that the textbook itself it tough to model, both by its size and complexity. See Appendix A.

2. Secondly, we faced issues in preparing training Question Answer data. We tried scrapping data off from Stack Overflow, or using textbook exercises, but we found both to be too complex to model, and often not requiring straightforward comprehension but subjective opinions. See more details in the Appendix B and C.

It's interesting to note how most of the issues discussed above also highlight the most common problems with data collection in training Machine Learning models. This is where we chose to retain the essence of the task and do away with data collection problems. Thus, we decided to use an already well-prepared dataset, the Stanford Question Answering Dataset v2.0.

The SQuAD has been popularly used in the field of machine question answering training, and hence serves as an important benchmark. The best performance on SQuAD 2.0 as of writing of this paper was by (Rajpurkar et al., 2016), which achieved a EM of 90.386 and F1 of 92.777, made using an ensemble of ALBERT, DAAF and verifier. Other of the top models that before above the human performance of 86.831 EM and 89.452 F1 often use the ALBERT model, sometimes ensembled with DAAF and transformer (Zhuosheng Zhang, 2020), or XLNET (Zhuosheng Zhang, 2019). Our models take inspiration from these in fine-tuning the hyperparameters as well as using pre-trained models for better performance.

2 Methods

2.1 Data

The SQuAD dataset v1.1 (Rajpurkar et al., 2016) is made up of 87600 contexts for training, plus question and answers related to that context, and 34727 similar entries for validation. The context corresponds to a paragraph containing information regarding many subjects, while the questions are simple factoid questions, whose answer is known and is taken from the context. There are multiple questions per each context, and there are more than

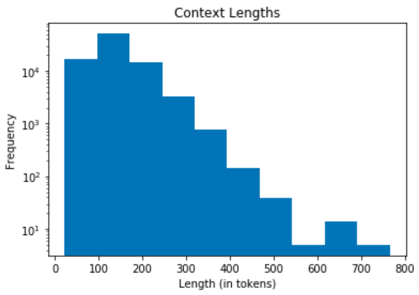

Figure 1: Context Lengths

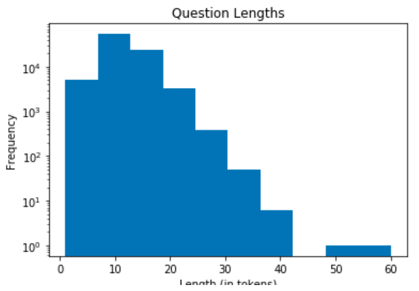

Figure 2: Question Lengths

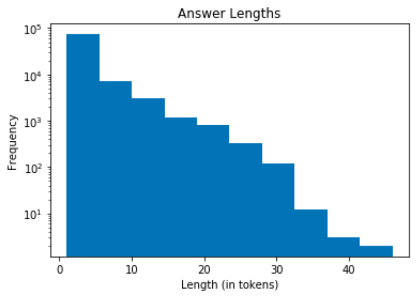

Figure 3: Answer Lengths

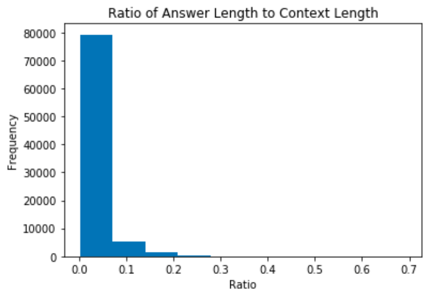

Figure 4: Ratio Answer : Context

500 contexts. The dataset has been widely used in previous works, and it's an optimal framework to build the models on, being it high-quality and sufficiently large. Hence, it is a popular benchmark in the machine question answering task.

The graphs above denote some important characteristics of the dataset. An example of context-question-answer tuple is proposed below:

Context ”The normal force is due to repulsive forces of interaction between atoms at close contact. When their electron clouds overlap, Pauli repulsion (due to fermionic nature of electrons) follows resulting in the force that acts in a direction normal to the surface interface between two objects. The normal force, for example, is responsible for the structural integrity of tables and floors as well as being the force that responds whenever an external force pushes on a solid object. An example of the normal force in action is the impact force on an object crashing into an immobile surface.”

Question ”What is the repulsive force of close range atom interaction?”

Answer ”normal force”

2.2 Models/Analysis

After researching about the methods used in the task of Reading comprehension and Question Answering, we chose to implement a single neural model which would then be perfected until the end of the project. We learnt that there are currently two approaches to solve this aforementioned task: Non pre-trained contextual embedding (Non-PCE) and Pre-trained contextual embedding (PCE). The difference between these two approaches is that in Non-PCE, a model is built from scratch specifically for the purpose of Question Answering, and this generally involves attention layers for the question to attend to context and vice versa. While in PCE, a pre-trained model with a predefined architecture and pre-trained parameters is used with slight modifications (or fine-tuning) to solve the required task. As such, PCE model can be utilized for various NLP tasks like sentence prediction, sequence classification among others. In this paper, we will discuss three models, two using PCE, and other using Non-PCE approach.

The model that we chose is called Bi-Directional Attention Flow, BiDAF in short, and was introduced by Seo et al. in their research paper in 2016. It was built originally using a non-PCE approach, although in our implementation we'll use PCE, specifically GloVe - Global Vectors for Word Representation (Pennington et al., 2014). The model's core is based on multiple deep neural network implementations of bi-directional LSTMs. An LSTM is a neural network architecture that can memorize long-term dependencies. BiDAF employs a bidirectional-LSTM (bi-LSTM), which is

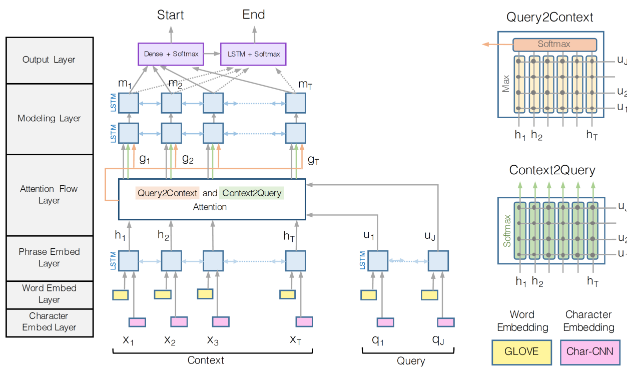

Figure 5: BiDAF Architecture

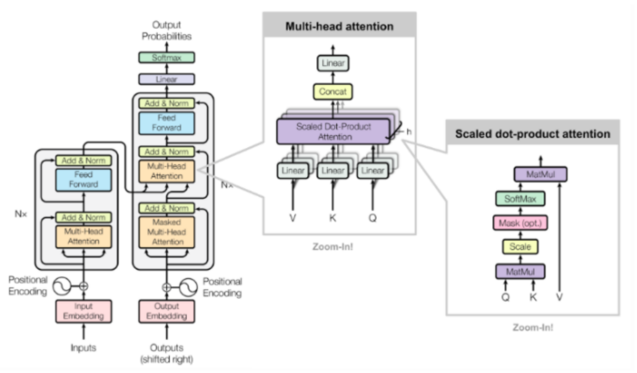

Figure 6: Transformer Design

composed of forward- as well as backward-LSTM sequences. The combined output embeddings from the forward- and backward-LSTM simultaneously encode information from both past (backwards) and future (forward) states.

BiDAF is composed of 6 layers which can be segregated into 3 main parts as below:

1. Embedding and Contextual Layers: 3 plus 1 in number, they work on changing the representation of the words in the both the context and the question from strings to vectors, and in capturing the relationship between the context and the given question.

2. Attention and Modelling Layers: The take as input the vectors from the embedding layers and fuses them so as to obtain the context portion relevant to the question

3. Output Layer: This layer takes the context representation from the above and transfers it into meaningful phrases and sentences for the answer, based on simple probabilities.

The context and the question are embedded separately following the same steps both at word- and character-level, then the encoding is merged. However, in our implementation, we just took care of embedding the characters, since for the word embeddings we used GloVe. The character-level embedding happened first through a vectorial transformation, then we applied a 2D-convolution on the character vectors, in order to capture dependencies among near characters. Then the output is merged and given to the next layer.
Between the embedding layer and the attentional layer there is the contextual layer, which is the first LSTM we encounter, and it halves the size of the context and question embedding. Next, the attentional layer is placed: it identifies which words of the context and question deserve more attention as being more important in relation to the sense of the question. After this, context and question are given together to the modeling layer: it is a bi-LSTM which learns what are the important parts of the context related to the question, therefore it identifies the position of the answer in the context. Finally, an output layer translates this importance values into probability distribution of the first and last word of the answer, taken from the context. The model is now able to select a piece of the context that corresponds to the answer to the question.

2.3 Baseline Models

We did a thorough literature review of all the models that expertise in machine question answering, especially pertaining to performance on the SQuAD dataset. We selected a model which became a popular choice in this domain after it was introduced not so long ago in late September 2019. The model is known as ALBERT Lan et al. and it has gives start-of-art results on the SQuAD benchmark (EM=89.731, F1=92.215) (Rajpurkar, 2020).

To have some baseline models against which to compare our BiDAF, ALBERT is similar to famous Bidirectional Encoder Representations From Transformers (BERT) model (Devlin et al., 2018) that was introduced by the Google AI team. When BERT was introduced, it was a breakthrough in language representational learning. ALBERT is a improvement over BERT, as it has fewer parameters, takes lesser time to train, occupies lesser GPU/TPU memory and performs better than the BERT model (EM=76.70, F1=73.85) on the SQuAD dataset. Therefore, for the purpose of this project, we trained an ALBERT model on the SQuAD dataset

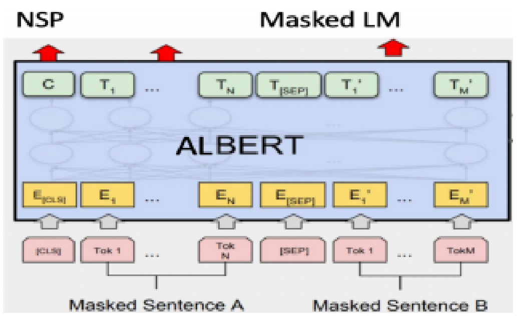

Figure 7: ALBERT Architecture

to be able to answer questions from given a context. However, to give more comparative results between different models, we have also provided experimental results from pre-trained BERT model which was adapted from Chris McCormick (McCormick, 2020). We have attached an architecture diagram of ALBERT and BERT in the next page for a comparison. The model was trained with 12 transformer blocks and self-attention heads and 768 hidden units (H). Transformers rely solely on attention resulting in increased parallelisation and reduced training time. See the design of a transformer in the figure above.

We used a pre-trained ALBERT model and fine-tuned its weights on SQuAD dataset using similar structure to BERT paper (Devlin et al., 2018). We concatenated the question and the context together and used them as one single input and separated the two inputs by the '[SEP]' token. Then we have vectors, $S \in R^H$ and $E \in R^H$, which are basically the probability of a word being start or end of the answer span respectively. For a word X_i the probability that it is the start of an answer is :

$$p(.) = \frac{e^{S.X_i}}{\Sigma_j e^{S.X_j}} \qquad (1)$$

And similarly for probability of X_i being at the end of an answer. The final answer prediction is equal to the text span where the sum of the two probabilities (start and end probability) is the highest. Being a deeply bidirectional, unsupervised language representation, without any substantial task-specific architecture modifications, training in ALBERT is performed in 2 ways:

1. Masked Language Modeling (MLM): by masking some words in the input, we try to predict the masked word by bidirectionally conditioning all the words in the context. As many as 15% of the input words may be replaced by a [MASK] token.

2. Next Sentence Prediction (NSP): with the input as pair of sentences, the model learns to predict if the 2 sentences appear subsequently in the trained context.

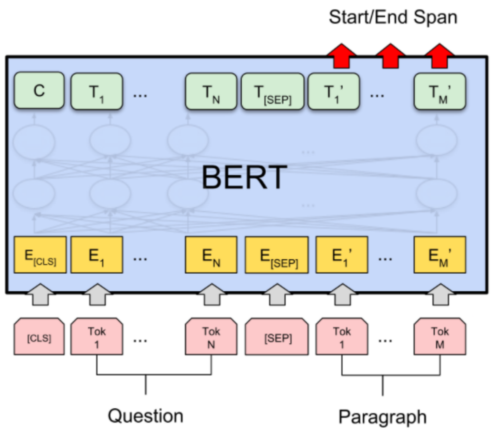

Figure 8: BERT Architecture

Often, both techniques are used together the minimize their individual losses. As for the loss function, we used cross-entropy loss with mean reduction to take the average inside each batch.

$$L = -\log p_{start}(i) - \log p_{end}(i) \qquad (2)$$

where $p_{start}(i)$ and $p_{end}(i)$ are start and end probabilities. The ALBERT model used for experiments in this project was adapted with the help of Hugging Face Transformers. (HuggingFace)

A more advanced ALBERT model addresses foregoing scalability problem of BERT from two major parameter reduction techniques: factorized parameter reduction and cross-layer parameter sharing. Additionally, it also proposes a self-supervised loss for sentence-order prediction (SOP), which improves upon the BERT version of determining inter-sentence coherence. Overall, the new ALBERT model improves BERT performance while using much fewer parameters.

3 Results

3.1 Experiment Setup

The SQuAD dataset is composed of (context, question, answer) tuples, as shown in the example earlier. The contexts are from Wikipedia, while the questions are strictly based on the context and the answers are a span from the context. Below are some of our findings of the 3 models on different batch sizes, learning rates and epochs.

Model	batch size	learning rate	epochs
BERT	12	3e-5	1
BERT	6	3e-5	3
BERT	12	5e-5	3
ALBERT	12	3e-5	1
ALBERT	12	1e-5	1
ALBERT	12	3e-5	3

Table 1: Model configuration details

BERT and ALBERT had a max sequence length of 384 for question and context combined, and doc stride of 128. They were trained on a Tesla P100 GPU provided by Google Colaboratory. (Google)

Regarding BiDAF model instead, we ran the model using the following hyper-parameter configurations:

- epochs: 3; context threshold: 100, 200, 400; hidden size: 50, 100, 150; char. convolution size: 100, 200;
- epochs: 10; context threshold: 100, 200, 400; hidden size: 50, 100, 150; char. convolution size: 100, 200;
- epochs: 12; context threshold: 100, 200, 400; hidden size: 50, 100, 150; char. convolution size: 100, 200;

This model too was trained on a Tesla P100 GPU provided by Google Colaboratory (Google). We chose not to go futher that 12 epochs since we observed that performance (EM and F1-score) did not increase anymore. Regarding the other hyper parameters, we observed that a low threshold improves the efficiency of the model in terms of training time, but if too low makes the model perform poorly. For the hidden sizes, we weren't able to use parameters higher than the ones here due to computational constraints by the use of just one free GPU.

As evaluation metrics, we have chosen Exact Match (EM) and F1 score. EM is the binary measure of whether the model's output matches exactly with the truth. This metric gives no ”partial credit”. F1 score is the harmonic mean of precision and recall. Precision is the percentage of correct predictions among all predictions. Recall is the recovery rate of the truth in the prediction. This metric is less strict than EM.

3.2 Result Comparison

The table below summarises the (best) results of training the models BiDAF, BERT and ALBERT on the SQuAD dataset. Regarding our implemented BiDAF model, the final hyperparameters that we chose as best performing are: epochs = 10, context threshold = 200, hidden size = 150, convolution size = 200.

Model	EM	F1
BiDAF	63.6	74.5
SOTA BiDAF	˜81	˜87
BERT	71.3	78.4
ALBERT	78.3	83.3

Table 2: Performance of BiDAF and ALBERT

We can easily observe that our implemented model fails to reach the levels of performance of the baseline models in these evaluation metrics, and it's even further from the actual State-Of-The-Art BiDAF model applied to the same task. There may be many reason why this is the case, an we'll try to expose them here next, and in the conclusion section.

On baseline models, we tested hyperparameters like batch size, learning rate and number of epochs. Some of the parameter values were taken from the original papers. Then, for the purpose of experimentation (and hope to obtain better results), we changed some values to see what difference it makes with respect to performance. Our tuning mainly used the approach of trial and error and by observing statistical patterns in the result.

Given our project objective, we observe negative results: in fact, we weren't able to reach the same level of performance of other models. That is, BiDAF is the lowest out of the 3 different models, and does not surpass or even match our two baseline models ALBERT and BERT. We can attribute these results to the fact that the baseline models were later developed in time, while the concept of BiDAF model goes back to 2016. At the same time, due to computational efficiency, our own BiDAF was only trained on a small subset of the SQuAD dataset, cut by the context threshold, and thus wasn't able to learn about all the contexts available to the other models.

There are 2 types of analysis which we said we will be performing in this project:

1. First, as a part of doing an comparative experimental study, to determine how good different

models perform, we will be comparing their EM and F1 scores on the training (SQuAD) dataset and justify why is that so.

- In this we conclude that ALBERT and BERT, our baselines, perform better than our own BiDAF model because of computational resources constraints and further developments in neural networks applied to text comprehension.

2. Second, we will also determine which model is the best for our goal, by taking into account different factors like time needed to train the model, how easily can the model be fooled, how the model performs on a large corpus of unstructured text and the accuracy score of the models.

   - In this too we conclude that perhaps our choice of baseline of ALBERT is a stronger model than the BiDAF we were able to train, both because of its architecture and training nature. However, we see that the SOTA BiDAF is able to outperform our baseline models, so that gives some credit to such neural network if fine-tuned properly.

Due to time and resource constraints, we were unable to test this model further on other real world data. But, the performance it has demonstrated in the SQuAD dataset shows that it holds great potential to be applied to any context and questions asked on it, as long as they hold a certain structural similarity, or the model is tuned to be able to handle raw data of any form.

What we are going to show next is a couple of examples of answers that we were able to generate from our implemented BiDAF model. The first one is an example of a correct answer, the second one of a wrong answer.

- Question: *"The Center in Paris is located near what river?"*
  True answers: *"The Seine", "Seine"*.
  Generated answer: *"seine"*.

- Question: *"What did Luther tell the legate about the papacy?"*
  True answers: *"papacy was the Antichrist"*.
  Generated answer: *"biblical church because historical interpretation of bible prophecy"*.

We tried to give an explanation to why the second answer is not even near the desired output and we found an answer by investigating the development dataset. The context related to the second question is one of the longest of the set, therefore it is sure that by limiting the context threshold to 200 characters it wasn't able to train on such long contexts, and therefore it tried to guess the answer using previously learned knowledge from shorted context and maybe more direct answers such as the first example. Therefore, we think that this might have happened many times during our training and because of the computational limitation we weren't able to overcome the obstacle and get a higher performance, comparable to the baseline models, which were a lot lighter to train.

3.3 Work Division

So far, the work division that we imparted was as follows:

- Rahul worked on comparing different models, cleaning the initial textbook dataset and helping set up the ALBERT baseline model.

- Pietro helped in setting up the BiDAF model and tuning it for best performance, as well as researching possible models.

- Ruhani worked on cleaning the stack overflow data, textbook question answering data and then on setting up BERT baseline model.

4 Conclusion

Through this project, our team had the opportunity to explore the wide field of machine comprehension and question answering. Through a thorough literature review, we understood the state of the art models performing such tasks at above human accuracy. We adapted from these models and tried to implement papers which produced good results on the SQuAD dataset in specific.

Furthermore, our results resonate with the research papers we referenced, in that pre-trained models such as ALBERT tend to outperform those trained on less data, and that computational limtiations, which imply fewer choices of model hyperparameters are one of the main drawbacks of our proejct. Our EM and F1 scores attained on SQuAD were roughly similar as well. This serves as a proof of concept that these models can be extended to raw, real world data and after much training and tuning, they can prove useful in comprehension of complex contexts and queries as well.

Potential for further research could be in trying these models in real-world unstructured data such as the SLP textbook and Stack Overflow QnA that our team initially had in mind. But the great obstacle in performing these tasks is to create a structured and labeled dataset, ready to do supervised and semi-supervised learning. An ensemble of these models could also be tested for greater accuracy. we could also try a greater fine tuning of layers and parameters for the very purpose of this dataset, to obtain more accurate results, but this would require more time and greater computational capabilities such as those from cloud computing providers.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Google. Google colab.

HuggingFace. Albert.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., USA.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations.

Chris McCormick. 2020. Question answering with a fine-tuned bert.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Pranav Rajpurkar. 2020. Squad explorer.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension.

Hai Zhao Zhuosheng Zhang, Junjie Yang. 2020. Retrospective reader for machine reading comprehension.

Junru Zhou Sufeng Duan Hai Zhao Rui Wang Zhuosheng Zhang, Yuwei Wu. 2019. Sg-net: Syntax-guided machine reading comprehension.

A Preparing SLP Textbook

We have used the textbook "Speech And Language Processing: Third Edition draft" (Jurafsky and Martin, 2009) as testing data for this project. The content of the chapters sums up to 174,335 words, and around 1,062,805 characters. The data needed to be formatted and cleaned before using it for our model. We needed it to contain only plain (unstructured) text, and not information like equations, tables, algorithm/pseudo codes, etc. which would over-complicate the task for our model. Hence, the following steps were taken to pre-process the data:

1. Convert the PDF file of the textbook to XML file format using an online tool.

2. Then using XML file, we extract the text depending on the font size and type. This helps in extracting the text that we need (as it is unique throughout the textbook) and removing unnecessary text like equations, tables, page numbers, algorithm/pseudo codes, etc.

3. Some additional cleaning required which was done through the help of Find & Replace.

This process cannot get us a 100% cleaned data, but cleans most of the text and is suitable for our model. Now the data is just a large corpus of text - sequence of words. The size of the data comes to 1.1 MB down from 17.5 MB of the original PDF file. The processed data now has 175,256 words and 888,109 characters.

B Gathering Training Data for SLP

1. Firstly, we tried our initial idea of being able to answer Stack Overflow queries. This we thought would be an advanced use case to achieve, so we tried to obtain stack overflow questions. In fact, obtaining the question and answer was not the difficult part. It was being able to model the complex questions and the convoluted answers, especially those mostly coming from subjective experiences or opinions.

2. We tried to procure textbook questions and answers from SLP itself so we could reduce the gap between the testing QnA and the text trained on, but we discovered that SLP exercise questions weren't straightforward either. They often required involved calculations or

understanding and reasoning which was beyond the scope of our model.

3. Lastly, we attempted manually writing simple questions and answers to test our model. This approach was feasible for small say 10-20 QnA, but it was not scalable to obtain a measure for our performance. Not to mention, self designing the test set could be influenced by bias as someone who knew the model and its code.

C Stack Overflow QnA

As per our initial plan, we worked on trying to gather Stack Overflow questions tagged as 'nlp'. We gathered this using the `data.stackexchange.com` interface which allows users to query the database of all Stack Overflow questions. We applied the following constraints when collecting these data:

1. Posts which are questions and have a 'nlp' tag attached to it;

2. Questions that weren't deleted or closed by the community, which happens when the questions are off-topic, vague or duplicate;

3. Questions that have an answer that was accepted as the right answer by the user who posted.

This way, we narrowed it down to from a collection of 16447 questions to 6477 questions, including their accepted answers. However, by a manual exploration of the data collected, we realized that it was not suited for the task at hand. The reasons why we discard it are the following:

- The questions were rather long and often grammatically complex. Simplifying them is another NLP task by itself.

- The questions may not revolve around something directly or explicitly discussed in the textbook, including a significant amount of subjective questions.

- The tags may have been misplaced introducing some wrong samples in our dataset.