Advanced Natural Language Processing (968G5) Coursework
The Microsoft Research Sentence Completion Challenge

Neil Narciso Fabião
School of Engineering and Informatics
University of Sussex,
Falmer Brighton BN1 9RH United Kingdom
Email: nf268@sussex.ac.uk
Candidate Number: 250945

***Abstract*—The MRSCC (Microsoft Research Sentence Completion Challenge) is a task which involves the prediction of missing words in a sentence over five possible alternatives. For this research, the two approaches considered were N-grams and Neural language models (NLM). The N-grams are probabilistic models which can predict the next word in the sequence given the word present in the corpus. However, this approach does not work well with unknown words, bearing long search times and dependency on the probabilities of the existing corpus [3]. On the other hand, NLM uses Masked language models, randomly masking words present in the sentence and predicting possible outcomes. Masking allows the models to look at the words in the sentence in both directions at the same time [4]. Capturing the full context of the sentences. For this research, the score based on accuracy entailed correct answers. N-grams produced a baseline score of 19% utilising the right context window and 28% for the left and right context window. Baseline BERT with 70% accuracy was later fine-tuned using the Roberta model. With Conan Doyle books in the Gutenberg library, the model achieved 80% accuracy with 105 books and eight epochs. Lastly, N-grams demonstrated an increase in perplexity by 20% as the corpus data grew while NLM stayed below a score of 11 [5]. Nonetheless, trigrams performed better as the training corpus data increased.**

Keywords: N-gram, Neural language models, BERT, MRSCC

1. Introduction

The Microsoft Research Sentence Completion Challenge (MRSCC) is a dataset that includes 1040 sentences from which a system has to predict the most likely missing word [1]. Created using the Project Gutenberg data, five books are from Conan Doyle's Sherlock Holmes novels. Each sentence is comprised of five candidate words where only one is the correct option. The candidate words were created automatically by the n-gram language model and groomed further by human beings. Thus, ensuring a more challenging task to solve using a language model given the probabilistic nature of local context words. In addition to being ambiguous and not easily identifiable since most options are grammatically correct.

The objective of this coursework is to investigate how two different language models perform predictions using the MRSCC dataset. The different approaches are :

1) N-gram models
2) Neural language models

The first approach N-gram model consists of unigram, bigram and trigram. In addition to their variations of context window, namely left context, right context and left and right context. The second approach is the Neural language model or transformer base model named BERT. The different modifications used in this research are distilroberta-base, Roberta-base and Roberta-large.

The performance method used for evaluating the different approaches is accuracy. Which accurately calculates the number of correct answers by the ground truth given to us by the authors of the (MRSCC) dataset [2]. In spite using accuracy is not common baseline practice when measuring the performance of both statistical and machine learning approaches. For this particular task accuracy was the measure of choice given the nature of the task. In this task, there is only one correct option to choose from the possible 5 even though there are other grammatically correct options. Furthermore, the authors of the dataset utilized accuracy as a performance metric even when evaluating against human subjects.

Model Name	Accuracy (%) in MSRSCC
Unigram	25%
Bigram	28.1%
Trigram	28.4%
BERT	70%
Roberta	80%
Roberta-large	86%

TABLE 1: Result of the two approaches investigated in the MSRSCC dataset.

During this coursework, the following research question emerged :

1) Does the size of the training corpus affect the accuracy of the two approaches?
2) Does perplexity score affect the model's accuracy? What strategies can be implemented to overcome these challenges?
3) What approach leads to greater accuracy for NLM?
4) What challenges arise from working with NLM and N-grams?

The best metrics achieved in this research are in table 1. The baseline metrics of the Neural language models were achieved using pre-trained versions offered by the Huggin-face community. Moreover, to achieve optimal results the models were primarily re-trained utilizing the 15 novels of Arthur Conan Doyle as well as random books located in the Gutenberg library. The hyperparameter tuning of both approaches was performed with different variations of the corpus data, training epochs, context windows and other factors.

This research is structured as follows. Section 2 is the Methodology with technical details of the implementation of the algorithms and the different strategies to tackle the MRSCC task used during this research. Results and analysis in Section 3. Lastly, section 4 presents the discussion.

2. Methodology

This section of the report, an explanation of the two approaches implemented in the research is presented. In addition to the experimental setup and the modules necessary to replicate the experiment.

2.1. N-gram models

Predicting is a challenging task when it comes to the future. However, with sentences assigning a probability to a sequence of words is beneficial. For example, given the sentence "Please turn your homework ". There are specific words which are likely to occur given the order and context of the sentence. For instance, one may suggest the "in" and others "refrigerator", which is incorrect. Probability is essential for machine translation, error correction and speech recognition. Because, they are robust to noisy, ambiguous input and suggest the most probable sequence or word in a sentence.

2.1.1. Understanding N-gram models. N-gram is a simple model that assigns probabilities to sequences of words and sentences. One of the first language models or LM. An n-gram with one word is called a unigram, two words bigram, three words trigram, four words quadrigram and more than five words 5-grams, etc. The sequence of two words in a bigram can be "turn away" or "subject matter", while trigrams "Please turn your" or "it's raining tonight".

The prediction of a missing word which is most likely to occur in a sentence is decided based on the frequency of the word existing in the corpus data. To compute an N-gram understanding how to get the probability of a given word is necessary. Which is given by $P(w|h)$. Read as the probability of a word w given some history h.

$$P(the|its\ water\ is\ so\ transparent\ that) = \frac{C(its\ water\ is\ so\ transparent\ that\ the)}{C(its\ water\ is\ so\ transparent\ that)}$$

Figure 1: An example of a probability of a word present in a sentence [3]. Where C is the frequency of that particular sequence of words

Given full training corpus for the example above was provided. The probability of homework would be $P(\text{tomorrow}|\text{Please turn your homework})$. However, to calculate n-grams there is a need for conditional probability. Which can be written as $P(A,B) = P(A)P(B \mid A)$. In addition to the chain rule which estimates the probability from relative frequency counts. An example of probability is given in Figure 1. Unigram probability selects the single most likely word by choosing the most likely word to occur in the corpus. Which is given by :

$$P(w_{1:}w_k) = \prod_{i=1}^{k} P(w_i)$$

The probability of a word is entirely dependent on the number of times a word appears in the corpus. The target word appearing is equal to the frequency of the word in the corpus (w_i) divided by the count of all the words in the data (w_n). In this approach the context words are not considered. Assuming that the probability of the word is independent of the other words seen in the sentence. Thus, bigrams and other n-gram probabilities address this issue by accounting for the previous words during their calculations.

$$P(w_{1:}w_k) = \prod_{i=1}^{k} P(w_i \mid w_{i-1})$$

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

2.1.2. Disadvantage. One of the main drawback of using N-grams is the ability to account for unknown words. For instance, if the sentence contains the word "despicable", and that word was never seen during the calculation of the N-gram probability. The likely hood of having to return an error message or wrong answers is high. Nonetheless, some techniques allow us to overcome this challenge such as the addition of the "UNK" token for each word or a given new sequence. As well as, Laplacian (add-k) smoothing and discounting. Moreover, the time it requires to access the probability dictionary for each word in the data can be exhaustive to the CPU. Requires long training time.

The context window where each probability is retrieved is important. N-grams are unidirectional. This means that the probability of each word is calculated in one direction. Either from left to right or from right to the left. To increase coverage of the words expanding the context window is an alternative. The next section introduces NLM which employs the concepts of transformers and different encoding and decoding techniques. Which overcomes the challenges mentioned in this section.

2.2. Neural language models

Neural language models comes from Neural networks, a concept from machine learning. In neural networks the representation of words is changed to vectors or word embedding. That allows the computer to encode the words in the sentences and learn the each representation as a vector space. From which, Neural language models (or continuous space language models) make use of continuous representations or word embeddings to make their predictions of the most likely word in the MRSCC [5].

2.2.1. BERT. BERT stands for Bidirectional Encoder Representations from Transformers. Which is an NLM that uses a masked language model (MLM) which was inspired by the Cloze task (Taylor, 1953) [1]. These models make use of transformers, encoders and decoders. From which there are two main elements such as the attention mechanism and the feed-forward neural network [6]. The attention is calculated as each input in the sequence is transformed to an output matrix with Query (Q), Key (K) and Value (V) vector. Given by the equation provided by [6]:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Where d_k is the dimension of the key vector and the softmax function is used to obtain the weights of the values. In the Multi-headed self-attention mechanism this step is repeated over the same sequence where Q, K and V are generated from randomly initialized projection matrices. Learned during training these matrices help the multi-heads create different dependencies over the input sequence. From which the embedding from this sequence returns an attention output which is given to the feed-forward neural network and then to the next block of the transformer encoder.

The base architecture of BERT has 12 transformer blocks, 16^3 attention heads the same size as Open AI GPT [1]. However, BERT is bidirectional while Open AI GPT only attend to the context to its left. Having the ability to be bidirectional allows for improvements in perplexity. Where a low perplexity indicates that the model can predict a sample or probability distribution optimally. Moreover, being bidirectional means that the models can learn from future and past contexts. Learning syntactic and semantic relationships similar in a way the computer can take advantage of the complete vocabulary. Furthermore, using subword tokenization allows the model to account for unknown or rare words. This enables the vector representation of rare words to be stored and shared among similar words such as the e.g solidifies and comprehending.

BERT base was trained on two tasks. Masked LM and next sentence prediction (NSP). In the Masked LM, a proportion of tokens is masked and the model has to predict the masked tokens. On the other hand, NSP is similar to the Question and answering task. The model is given two sentences and it masks certain elements. Then given a sentence it has to predict if there is an importer token. During the training of these tasks, 16GB of data was given to the model from which the model learn to leverage and adapt to the features present in the data. Furthermore, since the model was trained solely on a general text it enables the growth of the model's capability given specific domain-dependent tasks and corpora. Known as transfer learning using bidirectional language models makes it an effective strategy for neural language processing [6].

Lastly, one of the main points for BERT is the limit of input tokens to 512 as max length. Which is achieved with the assistance of padding and truncating sequences to 512. Where the attention mask becomes a vector of 0's for padding and 1's for input tokens.

2.2.2. RoBERTa. RoBERTa or Robustly Optimized BERT Pretraining Approach is an NLM which improves on BERT [4]. BERT is a general model which makes meaning it is not optimized and trained for specific tasks. Moreover, RoBERTa is trained for longer periods and in larger batches with more training data. Removing next sentence prediction objective or NSP and relying only on MLMs. Training on the long sequence of data and dynamic masking of the training split. Even though BERT and Roberta are different, the architecture remains the same. Nonetheless, Roberta has improved performance down streaming certain tasks and this results in a model which outperforms the BERT on the MRSCC [4].

2.3. Coding the experiments

The experiments were coded using python, on a MacBook Pro 2017 with a 2,3 GHz Dual-Core Intel Core i5 processor and 8 GB 2133 MHz LPDDR3. However, due to computational requirements, the simulations ran on a Windows OS with Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz 3.60GHz and 64.0 GB of RAM. In addition to google collab pro for the Neural language models since it required powerful GPUs [9]. Moreover, to achieve plausible and statistically valid results. Each implementation ran a total of 3 to 5 evaluations on a single model.

2.3.1. N-gram implementation. N-grams implementation has many variations. For this research 3 variations were tested. Namely, left context, right context and using both left and right context. These implementations were given during the tutorial labs of the ANLP course.

Hyperparameter tuning - For this section of the coursework the different values tested are present bellow in Table 2 :

Hyper-parameter	Parameter values to be tested
Training corpus size	5 to 225
Context Window seize	1 to 3
Training corpus split	0.1 to 0.9 (100 books)
Discounting	0.70 and 0.75

TABLE 2: Parameters to tested in this coursework

Cleaning the data - For this task, the data was not cleaned and remained the same. Only the books which contained unwanted characters and bad Unicode references were removed from the training set.

Experiment 1 - This experiment involved understanding which data split was the most optimal for the different n-grams (unigram, bigram and trigram). The training involved a data size of 100 and the different splits were 0.1 to 0.9.

Experiment 2 - Uncovering if the data size affected the 3 different contexts implemented. The starting size was 5 and 10 books and incremented in size of 25 until 225.

Experiment 3 - Understand if the window size benefited a particular context. The window sizes used were 1 to 3.

2.3.2. NLM implementation. The different implementations used in this research are Distilroberta, Roberta and Roberta large. The different re-training strategies are outlined during this section of the report.

Cleaning the data - To fine-tune the data an online tool for cleaning the dataset was used and can be found on [7]. This ensured unwanted proportion of the dataset remained during training.

Training with more than a single mask - This approach involves creating a sentence filled with each candidate word and then tokenizing the entire sentence. From which the special words in the sentence are transformed into smaller chunks and allow the model to have a more in-depth understanding of the training corpus data. By masking in this manner special words are accounted for in the corpus and the model performs with a higher degree of accuracy.

Hyperparameter tuning - The different values tested for NLM are as follows :

Hyper-parameter	Parameter values to be tested
Training corpus size	5 to 105 and 500
Learning rate	1e-3 and 1e-6
Weight Decay	0.1 and 0.2
Batch Size	8, 32 and 64
Length of sequence	128 and 512
Training epoch	6, 9 and 15

TABLE 3: Parameters to tested in this coursework

Experiment 1 - This experiment involved understanding the effects of off-the-shelf against pre-trained models on 15 books from Conan Doyle in the Gutenberg dataset.

Experiment 1.1 - Understand if increasing the size of the training dataset improved the model. The number of books used was 15, 50, 100 and 500.

Experiment 2 - Does the weight decay affect the accuracy of our model. The weight decay parameters used were 0.1 and 0.2.

Experiment 3 - Training for longer epochs versus shorter epochs. The epoch sizes were 6 and 10 respectively.

3. Results and analyses

This section of the research demonstrates the results of the approaches used during this research.

As seen in Figure 3 (in the appendix) as the training data increases unigram score goes below 26 % while bigram and trigram increase as more data is provided during training. Thus, the best split or corpus size for trigram is more than 70 books (0.7) while for bigram 50 (0.5) and unigram less than 20 (0.2). When testing for the best context window, the worst result came from looking only at the right context with the lowest result being 0.18 as seen in Figure 3 and Table 4 (in the appendix).

Moreover, the NLM outperformed N-grams as seen in Figure 4(in the appendix). Which Roberta-Large was the model with the highest accuracy without requiring re-training. In addition to presenting the lowest perplexity score. The weight decay parameters did not affect the score of the models while the batch size improved the speed of training our models.

As the window size increased trigrams performed better than unigram while unigram worked best with a window size of 1. Discounting factor affected how unknown words were addressed during the training of the corpus. Nonetheless, the N-grams models did not reach above 40 %. Training on larger corpus size for NLM such as Roberta large led to a decrease in score by 1% while for baseline BERT it increased by the same value reaching a score of 71.2 % compared to the NLM baseline of 70 %.

4. Discussion and Conclusion

As the size of the corpus data increases, bigrams and trigrams perform better than unigrams. This is due to the size of the training data increasing and fewer unknown words being present during the evaluation. Furthermore, the model to achieve the best accuracy was the off the shelf Roberta-Large with 86% as seen in Figure 4 (in the appendix). The size of Roberta large is greater compared to the other models since more corpus training data was used during training. However, when trained with 15 books the accuracy decreased by one percentage point. This may be given the time it takes to train this particular model since it has over 24 transformers blocks and 1024 hidden layers. Which are affected by bias when trying to fine-tune this version of the model given the training data used during train containing unfiltered corpus from the internet [9].

Moreover, perplexity as a measure of performance is not a good metric since the goal of the coursework is selecting the correct answer. Nonetheless, perplexity allows us to measure how well the model understands the data. The

models that had lower accuracy in this case NLM (Roberta-Large) with values below a score of 3. During this research lower perplexity equated to a model having higher accuracy. On the other hand models with higher perplexity values such as unigram as seen in Figure 5 did not perform well. Often generalized the information in the corpus and at times achieved scores similar to random choice.

For the models used in this coursework, the notable methods to avoid perplexity for NLM were increasing training time which can be achieved by using a higher number of epochs. This will equate to the model having an in-depth understanding of the data. Moreover, transformers models gave their bidirectional property can read the context on both sides instead of just left to right [6]. This is an advantage over N-grams which can only read the context from one direction. In addition to taking advantage of word embedding which creates a representation of the corpus data in a manner that the computer can understand with ease.

All in all, the MRSCC is a task that demonstrated interesting perspectives for the field of NLP research. N-grams or probabilistic methods pick the candidate words based on the local context and are limited to the existing data in the corpus. On the other hand, transformer models are pre-trained from online corpus, are not limited to local context and can outperform N-gram methods. However, the state of the art architectures due to their size and computational requirements. Limit potential NLP researchers which do not have the computing resources to run such models and improve upon them in the process. Making this task hardware dependent. Thus, efforts such as Green AI [11] provide a starting point for the discussion about the future of NLE and AI.

References

[1] Devlin, J., Chang, M.-W., Lee, K., Google, K. and Language, A. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. [online] Available at: https://arxiv.org/pdf/1810.04805.pdf

[2] Zweig, G. and Burges, C.J.C. (2011). The microsoft research sentence completion challenge. [online] Available at: https://www.microsoft.com/en-us/research/publication/the-microsoft-research-sentence-completion-challenge/

[3] Daniel, J. and Martin, J. (2021). Speech and Language Processing. [online] Available at: https://web.stanford.edu/ jurafsky/slp3/3.pdf.

[4] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V. and Allen, P. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. [online] Available at: https://arxiv.org/pdf/1907.11692.pdf.

[5] Jing, K. and Xu, J. (2019). A Survey on Neural Network Language Models. [online] Available at: https://arxiv.org/pdf/1906.03591.pdf [Accessed 28 Apr. 2022].

[6] Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, Ł. and Polosukhin, I. (2016). Attention Is All You Need. [online] Available at: https://arxiv.org/pdf/1706.03762v5.pdf.

[7] Kiasari, P.M. (2019). gutenberg-cleaner. [online] GitHub. Available at: https://github.com/kiasar/gutenbergcleaner [Accessed 18 Apr. 2022]

[8] paperswithcode (15AD). Papers with Code - Transformer QA. [online] paperswithcode.com. Available at: https://paperswithcode.com/model/transformer-qa.

[9] Google (2019). Google Colaboratory. [online] Google.com. Available at: https://colab.research.google.com/.

[10] Hugging Face (n.d.). Hugging Face – The AI community building the future. [online] huggingface.co. Available at: https://huggingface.co.

[11] Etzioni, R.S., Jesse Dodge, Noah A. Smith, Oren (2020). Green AI. [online] cacm.acm.org. Available at: https://cacm.acm.org/magazines/2020/12/248800-green-ai/fulltext.

Appendix

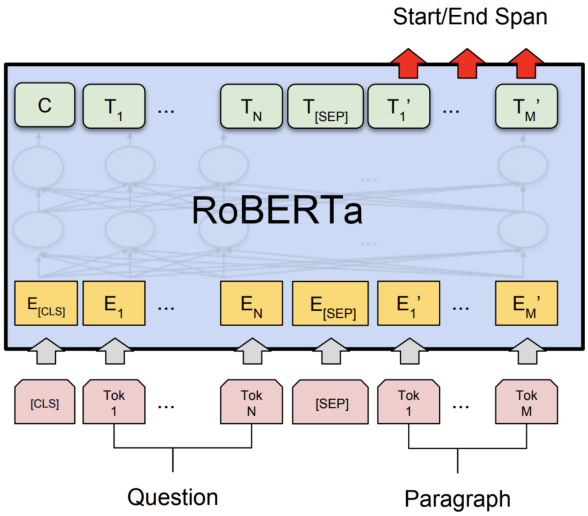

Figure 2: Example of roberta question and task. On the right hand side the model receives the paragraph or sequence of word which is then transformed using the encoder and decode. During the transformation the model assigns [CLS] and [SRP] tokens. However, in the implementation of this coursework Roberta uses $< MASK >$ tokens for the target area or blank spaces in the questions. From there using the resulting word embedding our RoBERTa model compares the output with of the original sequence.

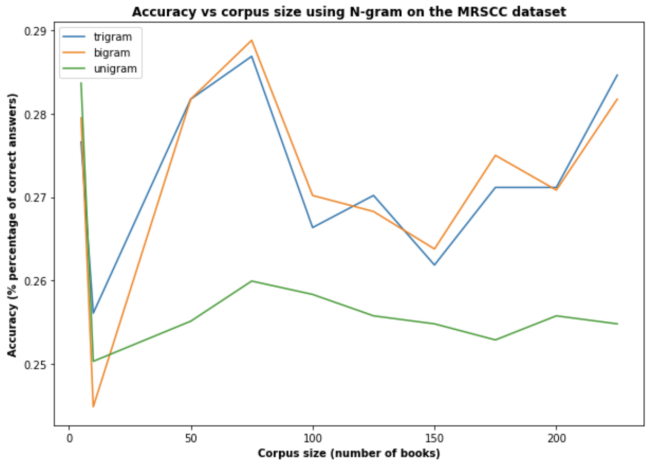

Figure 3: Visualization of accuracy score as the corpus size increases for N-gram. Unigram had a great start w with 5 books at 28.5 % however as the corpus increases the score decreased exponentially only rising closer to 26 %. On the other hand bigram and trigram observe and increase in accuracy as the number of training samples increases.

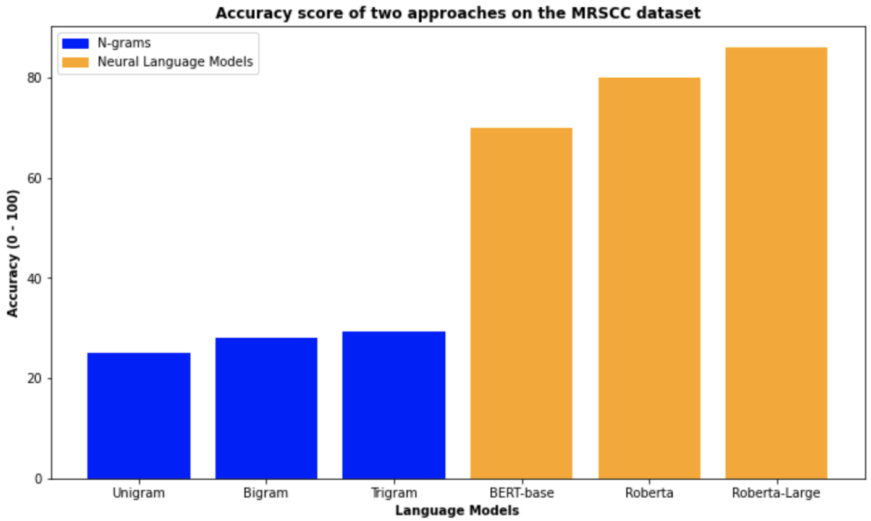

Figure 4: Scores achieved from the two approaches used in this coursework. In orange are the Neural language models while in blue the N-grams. The implementation for the N-grams both use left and right context. On the other hand for the Neural language models, BERT-base used 15 books, Roberta with 105 books and roberta large was downloaded or off the shelf.

Test case id (corpus size 100)	**left context score**	**right context score**	**left and right context**
1 (window size 2)	0.2580	0.1939	0.2581
2 (window size 3)	0.2583	0.2038	0.2583

TABLE 4: Results table of unigram with different context window

Test case id (corpus size 100)	**left context score**	**right context score**	**left and right context**
1 (window size 2)	0.2701	0.2096	0.2730
2 (window size 3)	0.2701	0.2096	0.2730

TABLE 5: Results table of bigram with different context window

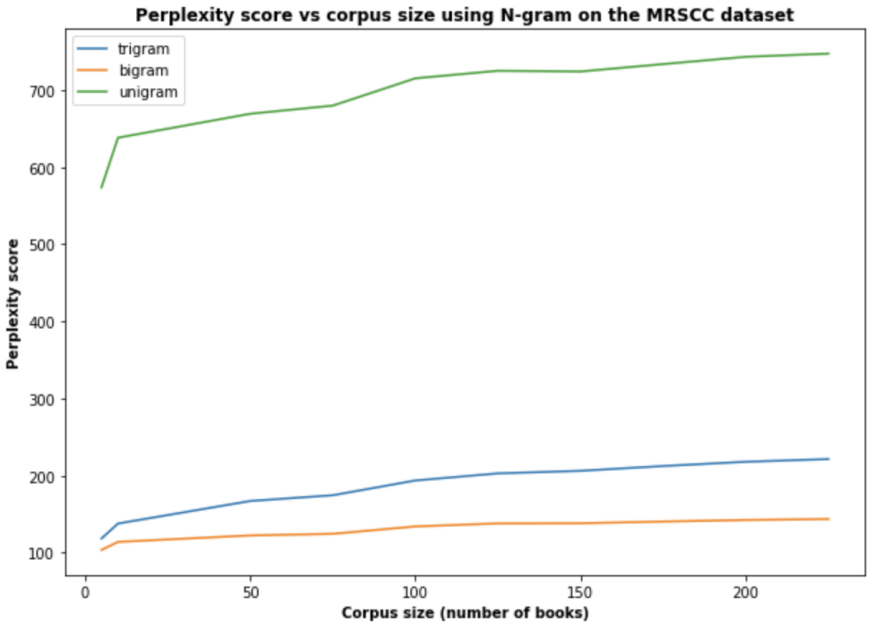

Figure 5: Visualization of perplexity score as the corpus size increases for N-gram. Unigram displays an exponential growth while trigrams and bigrams increase in similar proportion from their starting score.

Test case id (corpus size 100)	**left context score**	**right context score**	**left and right context**
1 (window size 2)	0.2644	0.2086	0.2692
2 (window size 3)	0.2663	0.2086	0.2430

TABLE 6: Results table of trigram with different context window

Test case id (trained with 10 epochs, decay of 0.1)	**Bert-base(distilroberta)**	**Roberta-base**	**Roberta-Large**
1 Baseline (off-the-shelf from hugginface)	0.702	0.7813	0.86346
2 (15 books from Conan Doyle)	0.7128	0.7875	0.85865
3 (50 books where 15 are Conan Doyle)	0.70385	0.79423	N/A (not completed)

TABLE 7: Results table of the best Neural language models with different different corpus data. Moreover, the training epochs used to achive the results were above 10.