INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 8, Issue 4 - V8I4-1143)
Available online at: https://www.ijariit.com

Lexical analyser web extension for text simplification

Harchit Mahajan
harchitmahajan2050@gmail.com
Government College of Engineering and Technology, Jammu, Jammu and Kashmir

Prateek Koul
prateekkoul12@gmail.com
Government College of Engineering and Technology, Jammu, Jammu and Kashmir

Sukhjeet Singh
sukhjeetdutta13@gmail.com
Government College of Engineering and Technology, Jammu, Jammu and Kashmir

ABSTRACT

Lexical simplification means the process of providing alternatives to the complex words in the sentence with texts that aremuch more simpler to understand, while also preserving the contextand grammar of the original text to make the whole sentence more easier to understand. All of the recent work involving lexical simplification relies on unsupervised tasks to learn simpler alternatives of complex words. But the drawback of most of these researches has been the fact that they provide simpler words without taking the context of the complex word in the sentence in account. In this paper, we are proposing a lexical simplifier which isbased on contextual learnings from the sentence. We have applied the pre-trained representation model, BERT. It is a very powerful tool which can make use of the wider context of the sentence in both forward and backward direction. We have also taken the wordfrequency indicator from the Subtlex list, to produce results that will be more correct both semantically and grammatically. We have also added a web extension for the simplification of the text on the webpage, which takes the input from the user, processes the text on the server end, and gives the result in return after computation is over.

Keywords: *Lexical simplification, Unsupervised, BERT, Pretrained Representation Model*

1. INTRODUCTION

With respect to Natural Language Processing (NLP), the process of Lexical Simplification (LS) tries to outperform earlier known task, Text Simplification (TS) by aiming the structure of the model on lexical information available in the sentence. This can also be described as the process of replacingcomplex words in the sentence provided to make it more simpler, without changing the structure of the sentence or making any additional modifications. LS is a whole pipeline which mainly consists of identifying harder words in a sentence, and searching for its best alternative. The best substitution has to be simple, while also maintaining the grammar of the sentence and upholding the original meaning ofthe sentence as much as possible. This itself could prove to be very difficult to do, since different subjects can have different understanding of an alternative provided by the model, i.e Users from different languages might not understand the synonym of the target word. LS can help all kinds of people, like children, not native speakers, etc, to deduce the sentence more efficiently. LS can be a very helpful way to simplify the sentence and some works [16] also reflect on the fact that people who have an understanding of one vocabulary can understand the sentenceand its meaning, despite the fact that the sentence may be grammatically or syntactically incoherent.

We propose to create a Lexical Simplification Model which will allow simplification of a given segment withoutchanging the definition of the segment by identifying complex words and replacing them with alternatives which retain the same meaning as the original word which then willbe deployed with the help of a web-extension using FastAPI to help users with difficulties in understand complex articles and resources online and simply them for their benefit.

The LS framework is usually divided into three subtasks: Complex Word Identification (CWI), Substitute Generation (SG), and Substitute Ranking (SR). The task for complex word identification can also be recognized as an independent task since it is usually solved with unsupervised learning methods to produce synonyms. Most simplification systems currently existing make use of a set of already defined rules to substitute the target words with their simpler counterparts from structured databases, like WordNet [3]. Some methods also make use of word embedding models for searching their synonyms by finding the most similar vectors to the target word vector.

Our LS model has the advantage of making use of the contextual meanings of the sentence in all three steps while generating a substitute for the target word and replacing it in the sentence. Let's take the example of the sentence "How a photograph of a young man cradling his dying friend sent meon a journey across India". Here, the words "cradling", "photograph" and "journey" are identified as complex words and can contain different meanings based on the context of the sentence. Some online websites which offer to simplify the text have given different results. e.g. Rewordify.com gives the result as "How a photograph of a young man cradling his dying friend sent me on a trip across India". Here, only the text "journey" is replaced with

its simplified counterpart "tour". Other websites such as Simplish.org gives the result as "How a photograph of a young man resting his coming near to death friend sent me on a journey across India". The words "cradling" and "dying" are considered as complex and are replaced with "resting" and "coming near to death" as complex counterparts. Both the grammatical and contextual meaning of the sentence changes here. Our model gives the result as "How a shot of a young man and his dead friend sent me on a tour across India". The words "cradling", "photograph" and "journey" are replaced along with preserving the contextual meaning of the sentence and making it simpler to understand aswell.

In this paper, we have explained the pipeline of our model and how it is to be implemented as a web-browser extension. The identification part depends on what the context of the text is by exploiting the features of a Bi-LSTM algorithm and making use of the transfer learning features from the Global Vectors embeddings [12]. To get the suitable substitutes, we make use of the self-supervised pre-trained model BERT [13]. We replace the complex word by the symbol '[MASK]', making a new sequence of words whose context does not change with respect to the complex word. We use these sequences and choose new substitutes with suitable probability distribution from BERT [13]. For the filtering of the substitutes, we have employed the Zipf frequency and chosen a threshold to filter out the simpler substitutes for replacing the word. Then, we use this trained model and employ it as a web-extension for the users to simplify the text on their web pages if they feel the need to do so.

The rest of this paper is organized as follows. Section 2 contains the literature reviews of the research papers studied for this project. Section 3 contains the methodology of the project. It explains the process and the pipeline of the model. Section 4 has a brief summarization of Results and Discussions of the model. We have discussed the results of our model in that section. Finally we draw our Conclusion in the last section. We have summarized a brief conclusion along with what more is left to be done in our project. At the end of this paper are all thereferences that have been cited in this paper.

A. Objectives

1) The complexity of a word can depend upon the contextof the sentence in which it is used. Some research would be done on determining a Machine Learning Model which takes the context of a sentence into account while determining the complexity of the word.
2) Out of all the candidates generated for the complex word, one must be selected which has the same meaning as the complex word and the complexity of the word should be low. Work could be done to derive an algo- rithm which can rank candidates by taking into account the meaning and complexity of the word.
3) As a result of the above two objectives, creation of a text simplification system for overall simplification of the textual information.
4) Creation of Web-based extension which will take se- lected content from web pages, communicate with text simplification system, relay the simplified content backto the web-pages.

2. RELATED WORKS

Earlier research on simplification of text was usually aimed at reducing the complexity of the grammar of the text while retaining its original meaning and maintaining the structure as before [7]. This task was usually achieved by implementing some text transformations at the syntactical and lexical level. In more recent times, several systems have integrally approached this task [8], [17], [18]. These types of comprehensive systems can perform a lot of the simplications processes, all at once, but this might result in the output having errors in the grammar, and may also result in complete change of the meaning of the text, thus making the original sentence more complex for the user to understand [9].

There was a paper published by Sian Gooding, Ekaterina Kochmar, by the name of "Recursive Context-Aware Lexical Simplification" at the International Joint Conference on Natural Language Processing in November 2019 [1]. They have proposed an architecture for the recursive context- aware lexical simplification (REC-LS). This architecture is more than capable of making use of a wider context of the words that need to be simplified. Then, the model suggests alternatives for the target words, while also taking in the previous simplifications of the sentence in account, thus making the output more grammatically correct and semantically appropriate. This architecture outperformed state-of-the-art systems in lexical simplification in 2019. In its task of complex word identification, the paper suggests to have used the SEQ Model [19], where the SEQ model labels each word in the sentence with a lexical complexity score. This score represents the likelihood (p) of each of the words available in the sentence being a subset of the more complex set of words. Whether the target word has to be considered as complex or not was defined by the predefined threshold for p. In the Substitute Generation task, the system generated the substitute candidates from the WordNet or NodeBox English Linguistics.

However, SEQ Model uses Threshold-Based methods like word frequency or word count, but only by using word frequency to determine complex words is not efficient.And itonly focuses on the targeted word to be substituted without taking the context of the word into account.

SimpLe: Lexical Simplification using Word Sense Disambiguation proposed by Nikolay Yakovets, Ameeta Agrawal, et. al. [7]: It is capable of (1) assigning meaning to every content word in text and finding out the similarities and relatedness using SenseRelate Perl Toolkit (2) taking the generated meaning of words into account and generating the best candidate for substitution. The system takes text and processes sentence by sentence. It follows two steps- (1) Word Sense Disambiguation - evaluate meaning of words and createsfile for word in base form and find words with similar meaning; (2) Lexical Simplification - compare candidates on length and usage count. Select the most frequent candidate. Context of sentence does not play a major part for candidate generation which can cause the candidates to have different meaning than the intended meaning in the sentence.

Text Simplification Based on Lexical Analysis proposed by Aysha, Gangothri, Kavya, Sherin, Anjali, et. al. [4]: The paper proposes factors like Length, Morphology, Familiarity, Ambiguity, Context for Complex word identification. Their system performs lexical simplification in 3 stages - (1) complex word identification using 4 parameters; (2) substitutes generation from two datasets namely Wordnet and Word2vec; (3) ranking each substitute.System uses two datasets namely Wordnet and Word2vec allowing them to generate a good number of candidates. Their ranking system for substitutes is not effective, thus making the end results of low quality.

Recursive Context-Aware Lexical Simplification proposed by Sian Gooding, Ekaterina Kochmar, et. al. [1]: The paper presented a novel architecture for recursive context-aware lexical simplification, REC-LS, that is capable of (1) making use of the wider context when detecting the words in need of simplification and suggesting alternatives, and (2) taking previous simplification steps into account. The system performs lexical simplification in 3 stages - (1) identify the complex words within the text along with the context of the text, using a threshold for complexity; (2) generate the substitutes from different datasets; (3) filter and rank each substitute; (4) use recursion on each instance of new substitute in the text and calculate the complexity of text again; (5) test the full LS pipeline on BENCH-LS and CERF-LS datasets.

Multi-Word Lexical Simplification proposed by Piotr Przybyla, Matthew Shardlow, et. al.[8]: It proposed the task of multi-word lexical simplification, in which a sentence in natural language is made easier to understand by replacing its fragment with a simpler alternative, both of which can consistof many words.

The task is split in two actions; - (1) Contribute a dataset of 1462 sentences with 7059 simplifications obtained through crowdsourcing. (2) Designing a method for generating simplifications automatically. Candidate generation uses a two-step recursive procedure.

The system uses unsupervised learning for the simplification method. Candidates are ranked on the basis of probability, similarity and familiarity. The identification of words depends on obtaining simpler fragments from crowdsourcing. The crowdsourcing method uses only 1277 sentences, which have 5 replacements.

3. PROPOSED METHODOLOGY

We are looking to create a lexical simplification model which can then be implemented in a web browser extension to simplify the text on the web pages which the user can then useto make the text simple to understand and preserve the semantic meaning of the text. The complete method can be divided into 2 tasks - making the simplification model using BERT [13] and creating the extension.

A. Simplification Model

The pipeline of simplification of the text can be divided into 3 sub-tasks, namely complex word identification, substitute generation, and substitute filtering and ranking. In this section, we briefly explain each step and the method used in the simplification process. Refer to Fig. 1 for the architecture of our whole simplification model and the wholeflow diagram of the model is shown in fig. 4.

1) *Complex Word Identification (CWI):* The main goal of this task is to identify all the complex words in the text which may need to be simplified by the model. Complex word identification from here on out would be represented byCWI in this paper.

2) The dataset used for the training and testing purpose of the CWI model is taken from [11]. This dataset is divided into a training set and a test set. We have used the pre- trained Glove Vectors (GloVe) embeddings [12] to create an embedding matrix for all the words in the training dataset. The GloVe Embeddings [12] are an unsupervised learning algorithm developed by researchers at Stanford University. This algorithm aims at generating word embeddings with the help of co-occurrence matrix. The co-occurrence matrices are aggregated from the given corpus. We have made use of the 300-dimensional GloVe embeddings [12] as word representations, where dimensions represent the features of the words that have been pre-defined by the researchers, like the length, morphology of the word, ambiguity of the word, etc. This algorithm is then used to predict whether the target word is complex or not and saves the result in a binary matrix.

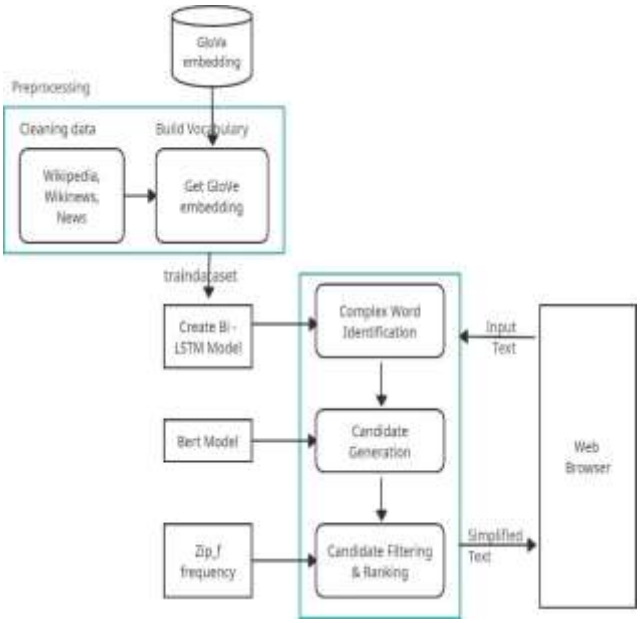

Fig. 1. An architecture of Bi-LSTM

The CWI task uses a bidirectional long short term memory (bi-LSTM) algorithm [10] which is trained and tested on the dataset prepared by Yimam et al. (2017). With the help of the bidirectional LSTM [10] will run the inputs in both forward and backward directions. This will help the model to understand the context of the text in both directions. It preserves the information from the future as well as the past using the two hidden states combined. As it is clear

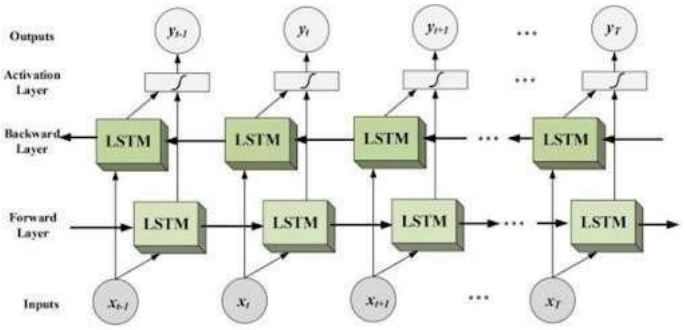

Fig. 2. An architecture of Bi-LSTM

from the fig. 2, the flow of data is in both the directions. Bi-LSTM is employed for sequence to sequence learning tasks.It makes use of the hidden states to pass the data and process it. The forget gate is the first activation in one LSTM [10] cell.

It makes use of the previous hidden state h_{t-1} and the input x_t. The input gate makes use of the next activations, sigmoid and the tanh function. Finally, the output gate makes use of the last activation and the tanh function above it. This algorithm is implemented on both sides. The mathematical equations corresponding to the inner working ofa bi-LSTM algorithm are mentioned from equations (1) to (5).

$$ForgetGate : f = (W_f[h_{t-1}, x] + b_f) \quad (1)$$

$$InputGate : i = (W_i[h_{t-1}, x] + b_i) \quad (2)$$

$$Gategate : g = tanh(W\, g[h_{t-1}, x] + bg) \quad (3)$$

$$CellState : c_t = f \cdot c_{t-1} + i \cdot g \quad (4)$$

$$OutputGate : o = (W\, o[h_{t-1}, x] + bo) \quad (5)$$

Where the sigmoid function is defined as

$$\frac{1}{1 + e^{(-x)}}$$

This model provides advantages as compared to other existing models since it takes the context of the text in account in both directions and requires less features to be taken into account while determining the complex words from the sentence.

3) Substitute Generation: This task is performed to produce the substitute candidates for the complex word w identified by the CWI task. We have used the pre-trained language model BERT [13] which produces the substitute candidates for complex words w. In this section, we willbriefly summarize what BERT [13] is and how it works. BERT [13] stands for Bidirectional Encoder Representation from Transformers. Transformers are based on attention and don't require any sequential computation per layer, only a single step is needed. Additionally, the gradient steps that need to be taken from the last output to the first input are onlyone. For RNNs, it increases with how much longer thesequence is. BERT [13] is a self-supervised model and makesuse of transfer learning and pre-training. These are the two steps in the BERT [13] framework. During the pre-training phase, the model is run on unlabelled data. BERT [13] is also optimized for better performance on the data. It uses MaskedLanguage Modeling (MLM) and Next Sentence Prediction (NSP). MLM helps in predicting the missing token in the given sequence, provided its right and left context. The nextoptimization is of the NSP task. It processes the new sequenceof text with masked tokens and prepends every sentence withthe delimiter as the specialized token [CLS] and the delimiter of the appen- dations of the sentences as separator token[SEP]. As shown

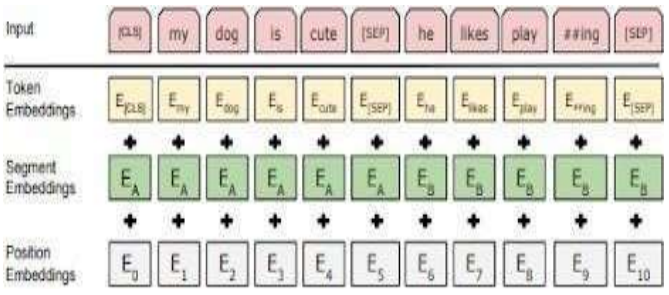

Fig. 3. BERT Objective

in fig. 3, BERT [13] has a predefined set of rules to represent the input text of the model. The Position Embeddings are usedto express the position of words in the sentence S. Segment Embeddings breaks the input into pairs and takes them as input for performing different tasks like Question Answering. The Token Embeddings are the embeddings learned for the specific token from the WordPiece Token vocabulary. For a given token, its given input is constructed by summing the corresponding token, position and segment embeddings.

The BERT [13] pre-trained model takes the sentence as input, replaces the complex word with the special token"[MASK]" making a new sequence of letters. Then it concatenates both the original and the new sequence of the words with the to- ken"[SEP]". This is how BERT [13] uses the making language model and gives new substitutes to the [MASK] token. Each new substitute generated is used to compute the probability distribution of the sequence and best of these substitutes are selected for the next step, filtering and ranking.

4) Filtering and Ranking: After Substitute Generation, the next task in simplification of the text is to filter them and replace the substitutes with the best option. Threshold based filtering is performed to remove some of the complex substi- tutes. For this purpose, Zipf frequencies are considered and their value is set to a specific frequency. The Zipf frequencies are extracted from the SUBTLEX list [14]. This list has been chosen because some experiments [15] have shown that the word frequencies from this corpus correlate on human judgements on simplicity. The Zipf frequency of a word is the base-10 logarithm of the occurrence of a word per billion words.

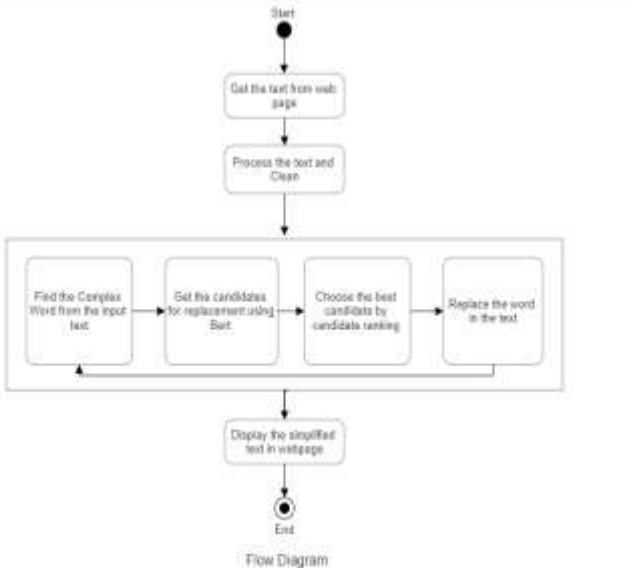

Fig. 4. Flow Diagram of our model

B. Web-Browser Extension

The next step of this project would be to implement this model as a web-extension for Chrome Browser. The users would be able to select the text that they want to be simplified,then use the extension. The extension will be supported in onlyone language, i.e. English.

To use this extension, navigate to the web page and select the text that needs to be simplified, then right click on it. The Chrome dialog box appears. Select the Lexical Simplification Extension.

Lexical Simplification Extension gets the selected text from the webpage and then calls the server using FastAPI. We have used FastAPI over other alternatives because of its more modern framework, better speed, robustness, easy to imple- ment, short and intuitive.. The development of FastAPI was done keeping the speed of operation, experience of developers and open standards in mind. It supports asynchronous tasks, which results in better speed than other alternatives. The architecture of FastAPI has a built-in system for validation of data types so that it can detect invalid initializations during the processing phase and returns the errors in JSoN format. For data validation it uses Pydantic, i.e. a tool that allows system datatypes to be defined.

For our project, we have used our local machine as a server for performing all the computational processes. The Lexical Simplification Model is stored on the server, and takes the selected text from the webpage as input and runs the model and generates the simplified text, which then it sends back to the browser page through FastAPI. Fig. 5 shows the flow diagram of how the web extension interacts with the FastAPI to perform the simplification task on the webpage. The Lexical Simplification Extension then scans the webpage

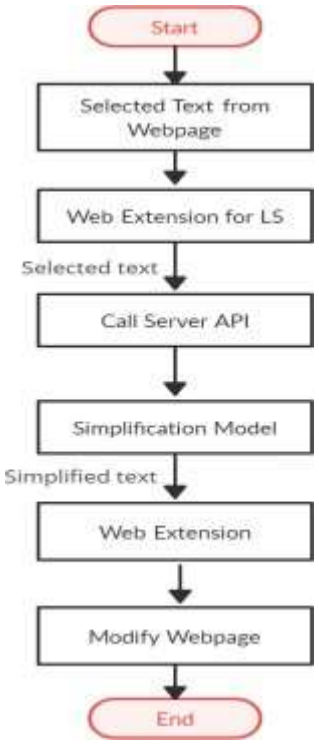

Fig. 5. Flow Diagram of Web Extension

and replaces the selected text by simplified text sent by Lexical Simplification Model from the server. The web extension has been deployed on the Chrome web store under the name of Lexical Simplification. Anyone interested can download the extension from the store on their local device.

4. RESULTS AND DISCUSSION

In this section, we will discuss the results of our CWI model and provide further discussions on what more we are going to implement in the project.

A. Results

We report the results on the CWI Shared Dataset prepared by Yimam et al. (2017) [11] on Wikipedia, Wikinews and News Datasets. We have used both the WIkinews and News datasets since the text in News are written by professionals while those written by Wikinews are written by common people so it is bound to contain more grammatical mistakes. After running the model for 5 epochs with 199 iterations in each epoch, we have plotted the graphs for the quantitative analysis of how our model behaves and gives results. We have employed regular metrics like Accuracy, Recall, Precision, F1 Score for measuring the metrics of the CWI algorithm. These are used to measure how the model performs with our dataset. We observe that the total loss of the model keeps decreasing

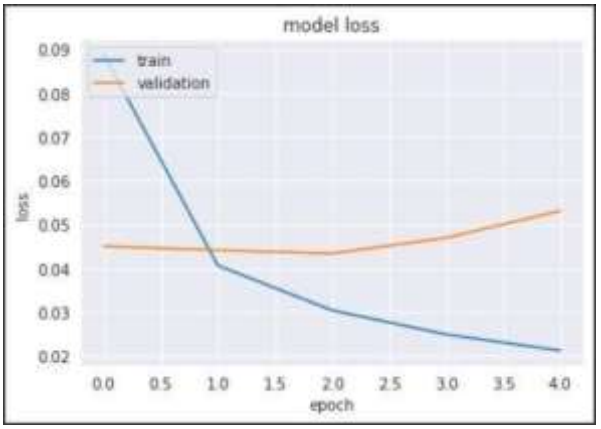

Fig. 6. Graph between Model Loss and Epochs

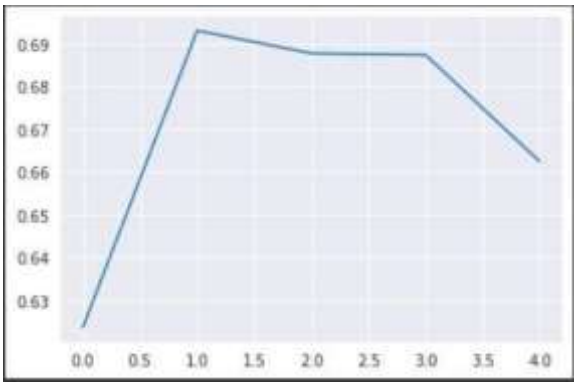

Fig. 7. Graph showing variation in F1-Score as epochs increase

after every epoch. This means that with every iteration the model learns. The dataset contains a total of 2041 sentences out of which 1988 are split into training data and remaining 53 sentences are reserved for the validation of the model for CWI task. The training loss and the validation loss of the model after running for 5 epochs with 199 iterations each remains almost constant with 0.02 loss for training the model and 0.05 loss while validating the model at the end of epochs.

The F1 Score is the harmonic mean of the Precision and Recall. As can be seen in the plot, the F1 Score for CWI task remains in between 0.65 to 0.70. This is a very good indicator that the model is performing well. We have observed that many of the models for the CWI task have a F1 Score between 0.68 to 0.80. The figures 6 and 7 show the graph between F1 Score and Model loss (both in training and validation) with respect to the number of epochs.

BEFORE :

Fig. 8. Highlighted text on the webpage before using the extension

AFTER :

Fig. 9. After the use of the extension

For running the extension, simply highlight the text that needsto be simplified by the user and right click so that the chrome dialog box appears. Select the Lexical Simplification option.It sends the selected sequence of texts to the server, which returns the simplified version on the webpage. This can be illustrated with the following two figures given below (fig. 8 and fig. 9). The text that is highlighted to be simplified is ”These pages are necessary to present important upcoming events that you want your visitors to know about. This could be anything from musical tour dates, business conferences or fun- filled workshops.” The words ”upcoming” and ”conferences” were identified as complex and were replaced by ”new” and ”events” respectively.

B. Discussion

We have till now done the CWI model which has a model loss of around 0.03 after 5 epochs and F1 Score of 0.68. The complex words identified are expressed in a numpy array of0s and 1s making this a binary classification array. These words have substitutes generated from BERT [13] pre-trainedmodel, which leverages on the masking language model and next sentence prediction model of BERT [13]. It focuses on\the context of the complex word. The ranking of these substitute is done with Zipf frequencies with the threshold of 3.1.For the future, we are looking to add more metrics intothe ranking and filtering task of the pipeline to select the substitutes with better contextual coherence in the sentence. These metrics include the cosine similarity, ngrams featuresof the text, length of the complex word, etc. and would give different weights to each of these metrics to make the filtering process based on equity of these metrics.

5. CONCLUSION AND FUTURE SCOPE

We propose a simple Framework based on BERT [13] for lexical simplification. Existing lexical simplification mod- els only consider the context into consideration on the last stage only (Substitute Ranking), but by using BERT [13] for substitute generation and Bi-LSTM algorithm [10] for complex word generation our system considers the context in all three stages of lexical simplification. The complex word identification model is completed and has a high accuracy. The substitute generation uses pre-trained BERT’s [13] masked language modeling (MLM). The substitute ranking is based on Zipf Frequency of the word. We have created a Browser Extension which simplifies the text on the webpage with the help of our lexical simplification model on one-click.

To further improve the system we can fine tune BERT for the process of Substitute Generation of target word. In the sub- stitute ranking phase more matrices such as co-sine similarity taken into consideration to improve the ranking system. The current system only works with single words system can be improve to include phrases as well.

6. REFERENCES

[1] Sian Gooding and Ekaterina Kochmar. 2019. Recursive Context-Aware Lexical Simplification. In Proceedings of the 2019 Conference on Empir- ical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4853–4863, Hong Kong, China. Association for Computational Linguistics.https://aclanthology.org/D19-1491.pdf

[2] Paetzold, Gustavo & Specia, Lucia. (2017). A Survey on Lexical Simplification. Journal of Artificial Intelligence Research. 60. 549-593. 10.1613https://eprints.whiterose.ac.uk/132086/

[3] George A. Miller, WordNet: A Lexical Database for English, 1995,Communications of the ACM Vol. 38, No. 11: 39-41 https://wordnet.princeton.edu/documentation

[4] Sanja S˘tajner, Horacio Saggion, Simone Paolo Ponzetto, Improving lexical coverage of text simplification systems for Spanish, Expert Systems with Applications, Volume 118, 2019, Pages 80-91, ISSN 0957-4174,https://doi.org/10.1016/j.eswa.2018.08.034.

[5] Yakovets, N. & Agrawal, A.. (2013). SimpLe: Lexical simplification using word sense disambiguation. Frontiers in Artificial Intelligence and Applications. 253. 131-138. 10.3233/978-1-61499-258-5-131. https://www.researchgate.net/publication/286492564

[6] Piotr Przybyła and Matthew Shardlow. 2020. Multi-Word Lexical Sim- plification. In Proceedings of the 28th International Conference on Com- putational Linguistics, pages 1435–1446, Barcelona, Spain (Online). International Committee on Computational Linguistics. https://aclanthology.org/2020.coling-main.123/

[7] Chandrasekar, Raman & Bangalore, Srinivas. (1997). Using Syntactic Information in Document Filtering: A Comparative Study of Part-of- speech Tagging and Supertagging.. 531-546. https://www.researchgate.net/publication/221510210

[8] Zhang, Xingxing & Lapata, Mirella. (2017). Sentence Simplification with Deep Reinforcement Learning.https://aclanthology.org/D17-1062.pdf

[9] Mandya Angrosh and Advaith Siddharthan. 2014. Text simplification using synchronous dependency grammars: Generalising automatically harvested rules. In Proceedings of the 8th International Natural Language Generation Conference (INLG), pages 16–25, Philadelphia, Pennsylva- nia, U.S.A.. Association for Computational Linguistics. https://aclanthology.org/W14-4404.pdf

[10] Hochreiter, Sepp & Schmidhuber, Ju¨rgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735. https://www.researchgate.net/publication/13853244

[11] Seid Muhie Yimam, Sanja Stajner, Martin Riedl, and ˇ Chris Biemann. 2017. CWIG3G2 - Complex Word Identification Task across Three Text Genres and Two User Groups. In Proceedings of the 8thInter-national Joint Conference on Natural Language Processing (IJCNLP) https://aclanthology.org/I17-2068.pdf

[12] Pennington, Jeffrey & Socher, Richard & Manning, Christopher. (2014). Glove: Global Vectors for Word Representation. EMNLP. 14. 1532- 1543. 10.3115/v1/D14-1162.https://aclanthology.org/D14-

1162.pdf

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Lan- guage Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Lin- guistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.https://aclanthology.org/N19-1423.pdf

[14] van Heuven, W. J. B., Mandera, P., Keuleers, E., & Brysbaert, M. (2014). SUBTLEX-UK: A new and improved word frequency database for British English. The Quarterly Journal of Experimental Psychology, 67(6), 1176–1190. https://www.researchgate.net/publication/259718817

[15] Marcos Zampieri, Shervin Malmasi, Gustavo Paetzold, and Lucia Spe-cia. 2017. Complex Word Identification: Challenges in Data Annotation and System Performance. In Proceedings of the 4th Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2017), pages 59–63, Taipei, Taiwan. Asian Federation of Natural Language Processing https://arxiv.org/pdf/1710.04989.pdf

[16] Xiaojun Wan; Automatic Text Simplification. Computational Linguistics 2018; 44 (4): 659–661. https://www.cambridge.org/core/journals/natural-language- engineering/article/abs/horacio-saggion-automatic-text- simplification-synthesis-lectures-on-human-language- technologies-april-2017-137-pages-isbn1627058680-9781627058681/292E0D2B8392F829F872FF87B134283

[17] Zhu et al., Lexical Simplification with Pretrained Encoders, 2020 Pro- ceedings of the AAAI Conference on Artificial Intelligence https://ojs.aaai.org/index.php/AAAI/article/view/6389

[18] Colby Horn, Cathryn Manduca, and David Kauchak. 2014. Learning a Lexical Simplifier Using Wikipedia. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 458–463, Baltimore, Maryland. Association for Computational Linguistics. https://aclanthology.org/P14-2075/

[19] Sian Gooding and Ekaterina Kochmar. 2019. Complex Word Identi- fication as a Sequence Labelling Task. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1148–1153, Florence, Italy. Association for Computational Linguistics. https://aclanthology.org/P19-1109/

