COMP 550 Final Project: Text Simplification with Neural Machine Translation

Anonymous ACL submission

Abstract

In this project, we implemented a Recurrent Neural Network (RNN) encoder-decoder machine translation model to approach text simplification, treating the target language, simple English, as a different language than plain English. We employed the Wikipedia dataset that contains 167K aligned sentence pairs from the original English Wikipedia and the Simple English Wikipedia. Comparing the RNN model with a baseline lexical simplification (LS) model, we analyzed various metrics, such as text similarity, readability, information retained, and the level of simplification. Overall, both models achieve a simplified result, however, with notable differences in the similarity scores, as a result of the difference in approaches. Throughout, we find that a neural network approach is comparable to and sometimes outperforms the rule-based lexical simplification method.

1 Introduction

In their 2016 paper (Wang et al., 2016), Wang et al. proposed using an RNN encoder-decoder neural machine translation (NMT) model to conduct text simplification. Prior to the writing of the paper, the authors believed that no one had approached text simplification with NMT; instead, most text simplification tasks were implemented with rule-based models, one of which was the LS model. However, due to the lack of data at the time, the authors proposed that, for future work, the machine learning community could use Wikipedia and Simple Wikipedia as the source of data to carry out NMT text simplification.

In this project, we set up to test the method proposed by Wang et al. We first built a baseline LS system that simplifies text by substituting less common words with their more commonly used counterparts. We then built an NMT model using the RNN encoder-decoder architecture trained and tested on the 167K aligned sentences from the English Wikipedia and the Simple English Wikipedia. We examined the resulting simplified texts with cosine similarity score, Jaccard similarity score, Dale-Chall readability score, Flesch-Kincaid readability levels, and manual inspection.

2 Datasets

We employed the dataset published by Kauchak (Kauchak, 2013) in 2013. This dataset contains 167K pairs of aligned sentences from the English Wikipedia and the Simple English Wikipedia downloaded in 2011.

For the Naive LS model, we used the data available through the Python library wordfreq (Speer, 2022) to rank the words based on complexity and to identify which words to replace and which words to replace them with. Furthermore, we used nltk's WordNet (Fellbaum, 1998) to find replacements for a given word based on its word sense.

For the RNN model, 80% of the Wikipedia dataset was used for training and 20% for testing/validation. Each sentence was removed of some punctuation, tokenized, vectorized, and padded to the same length before use.

Finally, due to the sheer volume of the data and the limited resources of our study, we took from the validation set 8,384 instances (5% of the total data) to compare the performances of the LS model and the RNN model. Each model was fed these instances to simplify and was evaluated on their performance through a collection of metrics.

3 Methods

3.1 Baseline Lexical Simplification Model

The baseline was implemented using a simple workflow. The first step involves the identification of complex or difficult words, followed by finding adequate synonyms based on their senses and replacement with a simpler word.

Starting with the identification portion: Although this is somewhat subjective and depends

highly on the context and the level of the reader, a couple of generalizing assumptions can be made for the case of a Naive Baseline. In this case, we assumed the rarity of the word is proportional to its complexity, which is generally true, with less frequent words being generally more complex, as they are used in more specific situations. Then, words that appear more frequently are more likely to be less complex or difficult.

In the implementation, the Zipf frequency was used to approximate the complexity of words. If a word fell below a certain pre-determined threshold, then it was selected for replacement.

For the selection of a synonym, word sense disambiguation was performed to determine the sense of the word and a simpler lemma from the synset was selected to replace it. The complexity was also rated in terms of its Zipf score. With a higher frequency being preferred, we selected the highest-frequency word in the list of synonyms.

3.2 RNN Neural Machine Translation Model

As proposed in the paper by Wang et al., we built an RNN encoder-decoder NMT model with an additional attention layer. We closely followed the tutorial by TensorFlow (TensorFlow), which is in turn based on the paper by Luong et al (Luong et al., 2015).

The architecture relies on an encoder-decoder with an attention mechanism. Given an input sentence from the English Wikipedia (the context sequence), the context sequence first goes through the encoder, a 256-unit bidirectional RNN. We allow a bidirectional flow of data in the encoder as the context sequence is constant and would not be modified. The encoder processes the context sequence and passes the output to the attention head to generate the key and value vectors.

The decoder takes as input the Simple English Wikipedia sentence (the target sequence) corresponding to the context sequence. The target sequence goes through the decoder's 256-unit unidirectional RNN. The RNN is unidirectional as the decoder generates the tokens of the target sequence one at a time in order. At each step, the decoder masks the tokens of the target sequence that have not been generated so far; it then predicts which token/word should come up next. After the target sequence has been processed, the decoder feeds the output to the attention head as the query vector.

The attention head takes as input the outputs of the encoder and the decoder. It calculates the weighted average across the key and query vectors, measuring the similarity between them. These attention weights direct the model's attention to certain parts of the context sequence when the model is generating the target sequence.

During training, the Adam optimizer was used. Early stopping was also in place to prevent overfitting. The model was trained on around 134K pairs of sentences and tested on around 33K pairs of sentences. The final validation/test accuracy is 73.9% after training for 33 epochs.

3.3 Evaluation Metrics

The evaluation of our two text simplification models involves the usage of numerous metrics to evaluate different aspects of the generated text. These metrics can be divided into two separate categories and have been chosen to provide an overall understanding of the model's performance.

3.3.1 Text Similarity/Meaning

- **Cosine Similarity:** This metric measures the cosine of the angle between two arrays of sentences, providing a numerical representation of the similarity between sentence embeddings regardless of their size.

- **Jaccard Similarity:** The Jaccard index compares the members between two sets to obtain the shared and distinct members. It is calculated by measuring the intersection of the two sets divided by the union of the sets.

These metrics aim to evaluate the semantic similarity and overlap in words between the original and simplified text. Cosine similarity captures the similarity in sense, while Jaccard measures the word overlap, analyzing these scores would provide an understanding of the text similarity between the original text and the simplified sentences produced by the models.

3.3.2 Text Readability

- **Dale–Chall Readability Score:** This metric evaluates the readability of a text based on the average sentence length as well as the percentage of difficult words/words contained in the Dale-Chall list.

- **Flesch-Kincaid Readability Grade Level:** Flesch–Kincaid Grade Level is a variant of Flesch's Reading Ease Formula. It evaluates

a text in terms of the U.S. grade level and is the most popular formula to classify texts into grade levels. This metric assesses readability by considering average sentence length and syllable count.

These metrics provide a basic measure of the text's complexity, analyzing various aspects of readability as well as helping determine the success of the simplification models.

The combination of these metrics forms a robust evaluation tool, allowing us to evaluate the performance of our RNN model in comparison to the ground truth and the baseline model.

Grammar was not evaluated in our methods due to the time required to evaluate the sentences. We had originally intended to use the Language Tool Python and Ginger to judge the grammar of the simplified text, but the computation and time required to run the tool were too high. Therefore, we decided to manually inspect the grammar of part of the output.

4 Results and Analysis

4.1 Ground Truth Simplified Text

The ground truth simplified text has a cosine similarity score of 0.436 and a Jaccard similarity score of 0.08 with respect to the original text. The cosine similarity score indicates a moderate resemblance between the two texts, while a perfect similarity score should be 1.0, this discrepancy can be attributed to the sensitivity of the tokenization and the word embedding we used for our evaluation function. For the Jaccard score, this can easily be explained by the replacement and paraphrasing of the original text which could lead to a reduction the same words being used in the two text which heavily impacts the score. To compensate for these variations, the ground truth score can be regarded as the score of a simplified text produced by a human, and to which we can compare our models. Furthermore, the Dale-Chall score of 6.41 is lower than the original text (6.59), which suggests that complex words were remove, but it did not significantly alter the text's readability, hence why it is still quite close. The simple text has a Flesch-Kincaid score of 8.8, which is lower than the score of the original text (10.1) by a significant margin, implying that there was a notable simplification in terms of sentence structure and syllable count; corresponding to a more simplified text in the readability and understanding aspects.

4.2 Baseline Lexical Simplification Model

The output of the baseline LS model has a cosine similarity score of 0.471 and a Jaccard similarity score of 0.09 with respect to the original text. The LS model scores higher on the similarity score than the RNN model or the ground truth as it replaces only the complex words, leaving the sentence structure and context untouched. Since there is no change in the phrasing, this allows the output of the baseline model to be similar to the original text.

For the Dale-Chall score, the LS model scores 6.22, which shows some degree of simplification that is lower than the sample from the ground truth simple text, but higher than that of the RNN model. Then, for the Flesch-Kincaid score, the baseline achieves the same score as the original, 10.1, which again can be attributed to the nature of the model where the sentence structure is largely unchanged.

The LS approach used in the baseline model leads to some difficulties in the grammatical structure of the text. In cases where the complex word in question is a noun, the output is usually grammatical and the task of text simplification proceeds smoothly. An example of this is the phrase "His dissertation topic was 'Representation of American Sign Language for Machine Translation'", which is simplified to "His thesis topic was 'Representation of American Sign Language for Machine Translation'". The word "dissertation" is replaced with "thesis", a much more common word. Note that what is quoted in brackets however is untouched, as the name of the paper should not be simplified.

On the other hand, there are examples where the tense and the plurality of the new word need to be modified to fit the sentence structure accordingly. This is often not being done correctly. For example, "Wilson was sent bankrupt due to a drought in the late 1830s and the subsequent depression" is simplified into "Wilson was sent break due to a drought in the late 1830s and the subsequent depression", which does not make sense in the context. So, the baseline model's grammatical performance is highly dependent on the sentence structure. The more complex the structure, the more difficult it is for the model to select an adequate replacement that fits both semantically and syntactically.

4.3 RNN Neural Machine Translation Model

The output of the RNN model has a cosine similarity score of 0.443 and a Jaccard similarity score of

0.08 with respect to the original text. The cosine similarity score is slightly lower than that of the ground truth simple text, but the RNN output scores higher in Jaccard similarity. For the Dale–Chall score, the RNN output attains a score of 6.05, lower than both the original text (6.59) and the ground truth simple text (6.41), implying that there is a large reduction in terms of "complex" words. For the Flesch-Kincaid score, the RNN output also has a lower score compared to the original text, with a score of 9.6 compared to 10.1; however, the score is higher than the simple text's 8.8. Regardless, we can see that the RNN model does indeed simplify the original text, while also being reasonably similar to the original text with respect to the ground truth.

Despite having similar cosine and Jaccard similarity scores as the simple text, we cannot conclude that the RNN model retains the core information of the original text, whereas we assume that the simple text does. Therefore, we manually inspected the output. We found that the output often missed information when it concerned numbers and proper nouns. Overall, some simplifications were up to the quality of the ground truth simple text, while others omitted or altered information. Below we show 4 examples.

4.3.1 Example 1. Breaking up long sentences

The RNN output successfully breaks up some long sentences into two shorter sentences. We can try to understand the logic of the RNN model by inspecting the attention scores. Figure 1 shows that the word "who" in the original sentence is attended by a period in the simplified sentence. This could mean that the model learns to turn relative clauses into new sentences.

4.3.2 Example 2. Omitting complicated parts

The RNN model learns to sometimes omit an entire part of the original sentence if the part contains several complex words. In Figure 2, we can see that "mandate police and judicial cooperation between countries who are members" is not a part of the output and that these words are barely attended by the output.

4.3.3 Example 3. Closely following the original sentence

For relatively simple sentences, the RNN model does not simplify the text much. On the attention plot, this is demonstrated by a clear diagonal line,

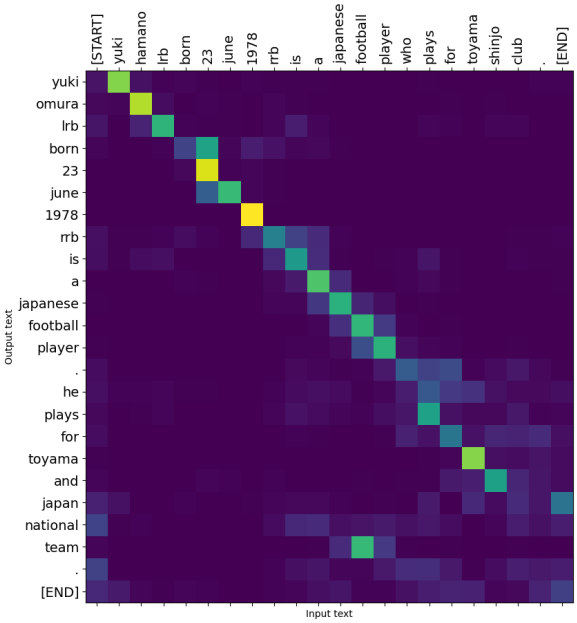

Figure 1: Attention plot of a long sentence broken up into two sentences

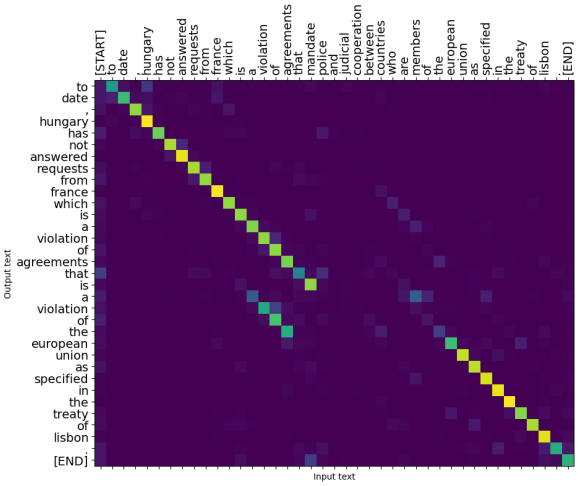

Figure 2: Attention plot of a complicated sentence

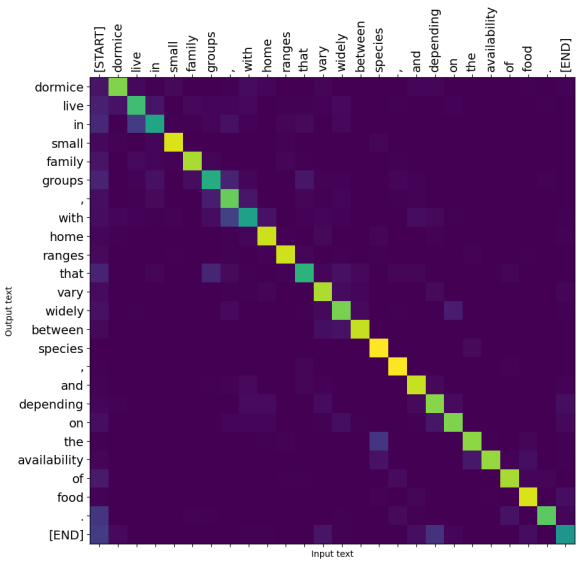

Figure 3: Attention plot of a simple sentence

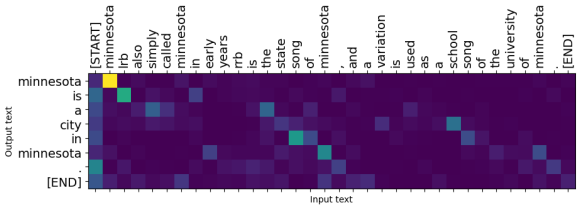

Figure 4: Attention plot of a sentence with proper nouns

as in Figure 3

4.3.4 Example 4. Complete failure

Sentences with locations, numbers, and proper nouns prove especially tricky for the RNN model– the output can be completely butchered. Figure 4 shows an example.

The sub-par performance with some sentences might be due to the complicated nature of the text simplification task. In the paper by Wang et al., the authors correctly posited that a single RNN model might not be enough to do text simplification. They suggested that text simplification should be split into sub-tasks, such as splitting and merging sentences; each subtask could then have a dedicated RNN encoder-decoder. This method is supported by Sulem et al. (Sulem et al., 2018) as they experimented with a splitting algorithm for NMT. We also think that there could be specific training on retaining precise information such as years, distances, and names.

Alternatively, RNNs could be used to strengthen rule-based text simplification models. Wang et al. suggested that we could combine LS with RNN, using the latter to help score candidate words in an LS system. Regardless, of our results, RNNs seem to have the potential to advance text simplification.

4.4 Comparison of the LS and the RNN models

Table 1 summarizes the results.

Table 1: Comparison of different texts

Text/Metric	Cosine	Jaccard	D-C	F-K
Original	-	-	6.59	10.1
Simple	0.443	0.075	6.41	8.8
LS output	0.471	0.091	6.22	10.1
RNN output	0.437	0.080	6.05	9.6

Both the LS model and the RNN model simplify text to different extents. However, to fairly judge the performance of the two models, we would need to compare them in terms of core information retained and grammaticality, which requires more manual inspection.

5 Discussion and Conclusion

Judging by different metrics, the NMT model is indeed a viable approach to the problem of text simplification, with improvements over the Naive rule-based LS model. Ultimately, the approach of changing the sentence structures allows for more freedom of expression and allows the model to approach the problem more freely. By contrast, the LS model is constrained to only being able to change certain words while trying to maintain the overarching sentence structure. Still, further refinement is needed for the NMT model. As mentioned in the previous section, we could utilize more than one RNN for text simplification. We could also experiment with using transformers, which could be even more expressive. Lastly, while the metrics used allowed for the quantification of similarity as well as readability, further improvements could be made, such as the inclusion of a grammaticality score to evaluate the syntax or the inclusion of other text similarity metrics such as BLEU and METEOR.

6 Statement of Contributions

Joshua worked on the evaluation metrics. William worked on the LS model. Sienna worked on the RNN model. All of them wrote the report. Repo Link: https://github.com/jjcs2000/COMP550Project.

References

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

David Kauchak. 2013. Improving the quality of suggestions for medical text simplification tools. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1537–1546, Sofia, Bulgaria. Association for Computational Linguistics.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation.

Robyn Speer. 2022. rspeer/wordfreq: v3.0.

Elior Sulem, Omri Abend, and Ari Rappoport. 2018. Simple and effective text simplification using semantic and neural methods.

TensorFlow. Neural machine translation with attention.

Tong Wang, Ping Chen, John Rochford, and Jipeng Qiang. 2016. Text simplification using neural machine translation. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 4270–4271. AAAI Press.