IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.8, Issue.6, June 2020 E-ISSN: 2347-2693

Simplification with the Transformer - Its Drawbacks

K. Mehta[1*], H. Chodvadiya[2], S. R. Sankhe[3]

[1,2,3]Dept. of Computer Engineering, KJ Somaiya College of Engineering, University of Mumbai, Mumbai, India

[*]Corresponding Author: kunj.mehta@somaiya.edu, Tel.: +91-90041-85066

DOI: https://doi.org/10.26438/ijcse/v8i6.15 | Available online at: www.ijcseonline.org

Received: 25/May/2020, Accepted: 20/June/2020, Published: 30/June/2020

Abstract— Natural Language Processing is an active and emerging field of research in the computer sciences. Within it is the subfield of text simplification which is aimed towards teaching the computer the so far primarily manual task of simplifying text, efficiently. While handcrafted systems using syntactic techniques were the first simplification systems, Recurrent Neural Networks and Long Short Term Memory networks employed in seq2seq models with attention were considered state-of-the-art until very recently when the transformer architecture which did away with the computational problems that plagued them. This paper presents our work on simplification using the transformer architecture in the process of making an end-to-end simplification system for linguistically complex reference books written in English and our findings on the drawbacks/limitations of the transformer during the same. We call these drawbacks as the Fact Illusion Induction, Named Entity Problem and Deep Network Problem and try to theorize the possible reasons for them.

Keywords— Artificial Intelligence, Natural Language Processing, Neural Networks, Text Simplification, Transformer

I. INTRODUCTION

Natural Language Processing(NLP) is a field that has many overlaps across other fields as well as within itself. Formally, it can be defined as a collective term referring to automatic computational processing of human languages which includes both algorithms that take human-produced text as input, and algorithms that produce natural looking texts as output[1].

Text simplification is a subfield of NLP within which research through the years has identified various tasks that can potentially obtain simplified text. Examples of these tasks include but are not limited to Named Entity Recognition, Coreference Resolution, Relationship Extraction, Terminology Extraction, Complex Word Identification, Word Sense Disambiguation, sentence deletion, sentence splitting. Techniques and computational models for text simplification are then developed taking into consideration one or more of these tasks.

We find it pertinent here to present our understanding of text simplification as that is what drives our approach. Previous definitions[2,3] of simplification have emphasized on simplification as a process that reduces the complexity of text while preserving its original meaning. We concur with the definition in that simplification should preserve the original meaning of text. However, we find it more suitable to say that text simplification should increase the accessibility of the text for the readers, instead of just saying that it reduces the complexity of text. On the basis of this, we define text simplification as the process that uses various methods to make the text more accessible to readers and retaining the original meaning of the text while doing so. Moreover, often, text simplification is confused with text summarization. A critical difference between two is that simplification focuses on meaning preservation disregarding resultant length of the text, while text summarization involves reducing the resultant length disregarding information preservation; at times, each may include the other.

Past research into text simplification techniques has been mainly driven by two approaches: lexical and syntactic. Lexical simplification is the task of identifying and replacing complex words with simpler substitutes without any attempt to simplify the grammar of the text[3]. Lexical simplification can even be extended from word replacement to phrase replacement. Syntactic simplification is the technique of identifying grammatical complexities in a text and rewriting these into simpler structures[3].

However, we believe that a sentence is simpler only if its meaning is preserved. During our research into the above approaches to text simplification, we observed the following

- Both of the approaches do not guarantee that the simplified sentence will have the same meaning as the original.
- In lexical systems, replacing words with phrases may ruin the syntactic structure or a failed word sense disambiguation may change the meaning of the resulting sentence.
- In syntactic simplification too, trying to get a simpler syntax may lead to a difficulty in reading (such as when reading passive voice) and distortion in meaning.

- Another difficulty with syntactic approach is that better systems are designed only with handwritten rewrite rules which are extremely difficult to write and as such the system does not work if it encounters sentences with a novel syntax.

Based on our belief and the above observations, we decided to use techniques that are better at meaning preservation for simplification. We call this approach with a focus on meaning preservation as the semantic approach.

To analyze the performance of a simplification model, various metrics and measures have been defined. These can roughly and representatively be classified into abstract and automatic evaluation methods. The abstract evaluation methods involve readability and understandability and need human evaluators of the simplified text. The automatic metrics such as BLEU, SARI and NIST are taken from the field of Machine Translation(MT) and are used to measure how good the 'translation' between complex English (or any other language to be simplified) and simple English is. However, simplification quality is very hard to measure because of its subjectivity; in language, no answer is wrong as there can be multiple ways of arriving at a simple sentence. Because of this and problems that have been reported[3,4,5] with the defined metrics, we have not used any metric for performance evaluation. Instead, we present the output for example inputs as is.

The rest of the paper is organized as follows: Section II mentions the previous related work on text simplification, mainly using the syntactical or lexical approach; section III presents our system; and section IV analyzes the reasons why the transformer architecture failed in simplification.

II. RELATED WORK

As previously mentioned, text simplification techniques are developed taking into consideration one or more of the text simplification tasks. In this section, we review the past techniques that have been employed for simplifying text.

Earlier techniques[2] majorly focused on building models and systems performing syntactic tasks either with handwritten rules or with automatically acquired rules of grammar with limited success. Some lexical systems were also developed, again with limited success. Some of the older techniques viewed the simplification task as a monolingual translation task which allowed them to take advantage of advances in the MT field. This monolingual perspective to simplification was given a boost with the construction of the English Wikipedia - Simple English Wikipedia(EW-SEW)[6] dataset that contained aligned complex and simplified English sentences from Wikipedia. Models could now be trained to learn relationships between complex and simple sentences in the train set and apply them to new sentences.

Many of the initial systems that achieved success with the EW-SEW dataset used Statistical Machine Translation methods. One such system[7] was a tree-based system that performed syntactic simplification operations on input data - splitting, dropping, reordering, and substitution. This system was trained on a different version of the EW-SEW dataset called PWKP. Systems developed prior to this had been successful in performing each aforementioned sentence operation only individually which made this system the first to be able to integrate all sentence operations.

Other simplification systems that were successful in simplifying text used different lexical Phrase Based Machine Translation (PBMT) models. PBMT 'translates' phrases using phrase alignments and a language model of the target language while inherently ignoring syntactic structures of sentences. Siddharthan mentions that Coster & Kauchak modified their PBMT model to allow for deletion of text from the original to form the simplified version by enabling alignments between a source phrase and an empty target phrase[2] . Wubben et al[4] used a PBMT model which was without a deletion module. Instead, they extended the system to re-rank 'translations' based on dissimilarity with the input sentence. Their aim was to find phrase alignments where the simple phrase is as different as possible to the original phrase, the intuition being that such paraphrases are most likely to simplify the text.

After the Statistical Machine Translation (SMT) text simplification systems (like PBMT) hit a performance plateau, Neural Machine Translation techniques became the next big thing for text simplification with the rise of Sequence to Sequence (seq2seq) models and later, with the incorporation of attention mechanism within them. The reason seq2seq models are more popular nowadays is they are trained end-to-end and do not need external components like decoders and phrase tables that were needed for PBMT.

One of the first seq2seq simplification system was the Neural Text Simplification (NTS)[8]. NTS could perform both syntactic (though limited to only content reduction) and lexical operations simultaneously, which was a step up from previous systems that managed to do this only individually. NTS used stacked LSTMs with global attention and input feeding. To prevent simplification of named entities, they replaced all such entities with an 'unknown' token during training. This is something we tried to do within our system as well.

Another seq2seq system was the Neural Semantic Encoder (NSE)[5] whose major advantage was using memory augmented RNNs as components instead of LSTMs. Memory augmented RNNs allowed the system to access the whole input together with practically no limitations to available memory.

Analyzing the outputs[5,8] and performance score presented for these seq2seq models, we found that in the case of NTS, the best model in meaning preservation is a very weak simplifier while the best simplifier loses information during the process. Similarly, even though the NSE is better, it still sometimes loses information while simplifying.

More recently, the transformer[9] architecture has proved promising in various NLP tasks including text simplification. The transformer is able to overcome the shortcomings of seq2seq models using simple feedforward networks instead of LSTM and by employing self-attention, multihead attention and positional encoding. We believe the Key- Query-Value model of attention employed in the transformer enables it to capture the semantic meaning of sentences more effectively.

III. METHODOLOGY

Pursuing the objective of developing a simplification system for linguistically complex reference books that can simplify the content of the books and make them more accessible to readers, we devise a system as shown in Figure 1. We enable the user to input the content of the book in any manner - PDF, JPEG or just plain text.

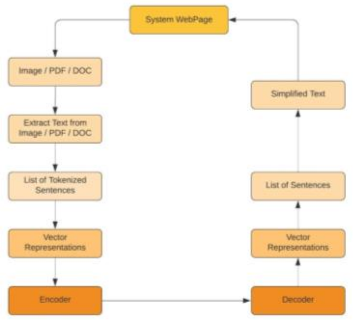

Figure 1. Data Flow in Transformer Architecture

Using various Python libraries for Optical Character Recognition (OCR), we extract the text from the PDF and JPEG files, tokenize the sentences and vectorize the words to generate word embeddings. Instead of using word embeddings like Glove, we settle for a simple mapping of tokens and their vectors as an embedding because the similarities and dissimilarities encoded in such embeddings can theoretically be taken care of by the transformer. The encoder of the transformer takes the embeddings as input, applies positional encoding, self-attention and multi-head attention to generate a context vector which the decoder uses to output a simplified sentence.

Further, keeping in mind that attention can only deal with fixed-length strings, we set the sentence length of the input to encoder during training as equal to the length of the longest sentence in the dataset. For all other sentences, we used the 'NOP' token[8] to fill in the gap. This ensures that no sentence is split randomly and its meaning lost.

The layer structure of the transformer we have implemented can be seen in Figure 2 which has been built taking inspiration from contemporary work[10] . There is one multi-head attention layer in the encoder and two in the decoder. RelU is used for normalization throughout. The softmax layer outputs probabilities for each word in the vocabulary during each time step considering the semantics and attention scores, out of which the most suitable word is selected as the next word in the simplified sentence.

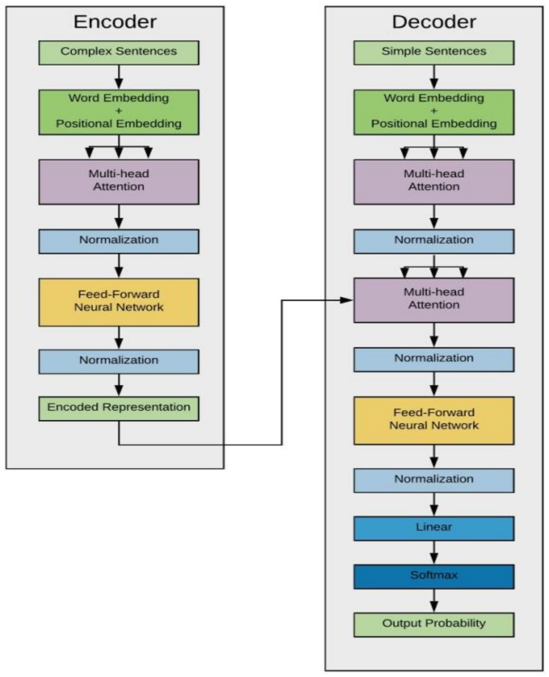

Figure 2. *Layer Structure of Implemented Transformer Model*

Having already established the significance of EW-SEW dataset, we use the first version of such a dataset[6] to train the transformer. We experiment with three models of the transformer, each having different hyperparameters. The initial model, MODEL 1 is built using 6 encoder-decoder layers, 8 head attentions, 512 dimension vector representations and trained with a learning rate (LR) of 0.0001 for 40 epochs. MODEL 2 is built with 3 encoder-decoder layers and 4 head attentions keeping the rest of the parameters the same. MODEL 3 has 2 encoder-decoder layers and 8 head attentions keeping every other parameter constant.

IV. RESULTS AND DISCUSSION

The results of training the above mentioned configurations of the model can be seen in Table 1. The error represents the relative confidence level of the transformer decoder for different words in the vocabulary with regards to the next word to be output. Higher the confidence, lower the error. For instance, if the confidence for a particular word to be output is similar to that for other words in the vocabulary then the error will be high. If the confidence is a certain order of magnitude higher, then error will be low.

Table 1. Training results of different transformer configurations

Model No.	**Encoder-Decoder**	**Heads**	**LR**	**Epochs**	**Dataset**	**Error/Loss**
MODEL 1	6	8	0.0001	40	EW-SEW	0.668
MODEL 2	3	4	0.0001	40	EW-SEW	0.501
MODEL 3	2	8	0.0001	40	EW-SEW	0.442
MODEL 4	2	8	0.0001	80	PWKP	0.418
MODEL 5	2	8	0.0001	80	PWKP	0.379

It can be seen from Table 1 that MODEL 3 outperforms the previous models, but its output is still not satisfactory. It should also be noted that we change the dataset during the fourth training. We now analyze these results and present the limitations of the transformer that leads to unsatisfactory simplification.

A. A deeper transformer model is not able to converge properly

MODEL 1 in the table has an error of 0.668. However, on reducing only the depth of the transformer as is done in MODEL 2 and the first iteration of MODEL 3, we can see that the error reduces. We believe that a deeper transformer network is not able to converge as it overfits over the data.

B. Problem with named entities

All of the transformer models have been found to encounter a problem with named entities in the dataset. The model seems to consider the most viewed frequent named entities as features contributing to the meaning of a sentence and learns it. Owing to the large number of named entities in the initially used EW-SEW dataset, the performance of transformer suffers. We tried solving this by replacing the named entity in the dataset with tokens[8] so that the model learns the meaning of the sentence and not the named entity in it. However, even this approach has an issue: because the entities are replaced by a common token, during decoding the transformer cannot differentiate between the entities and ends up mis-replacing them in the simplified sentence, destroying its original meaning. Note that due to this reason, we change our dataset to PWKP which has comparatively less named entities and performance of transformer improves slightly.

C. Fact Illusion Induction

Even our best configuration of transformer, that is MODEL 3 was found to not give satisfactory output. While analyzing it, we have found that the output sentences always contain some information that is neither expected nor present in the complex input sentence. We theorize that the information is coming from other sentences in the dataset. The model is inducing illusionary facts learnt somewhere else into the sentences that it outputs. There is a precedent of this phenomenon elsewhere and is known as Fact Illusion Induction. Below we present an example of fact illusion induction from an input-output pair that we obtained while testing the transformer model.

Complex Sentence **-** She won four Grand Slam singles titles, six Grand Slam women's doubles titles, and four Grand Slam mixed doubles titles.

Simplified Sentence - 2006 , the U.S. titles has won the major of the dominant singles 6 Grand Slam singles titles *<sos>* The dominant country in the doubles won the doubles titles.
(<sos> indicates the start of a new sentence)

As can be seen, the transformer does manage to split the sentence to simplify it. But the neural model has generalized and mapped the *games* in the complex sentence with the *2006 matches* (that is the Olympics). This induced information is quite frequent in the dataset. So, when the model recognized the *matches*, it induced the other part of mapping i.e. *2006*.

V. CONCLUSION AND FUTURE SCOPE

In this paper, we focus on meaning preservation during text simplification. We argue that any 'simpler' text should first preserve the original meaning for it to be considered simple. Based on this, we show that semantic approaches can in theory be considered better than lexical and syntactic approaches for text simplification. We also present a simplification system for linguistically complex reference books which uses the transformer architecture and we analyze the reasons why this system does not work, which is due to the drawbacks of the transformer itself. We go on to provide possible explanations as to why the transformer suffers from these limitations - Fact Illusion Induction, problems with named entities and convergence problems for deeper networks.

REFERENCES

[1] Y. Goldberg, G. Hirst, "*Neural Network Methods in Natural Language Processing*", Morgan & Claypool Publishers, **USA**, 2017. ISBN no. 9781627052955

[2] A. Siddharthan, *"A Survey of Research on Text Simplification"*, ITL - International Journal of Applied Linguistics, Vol.**165,** No.**2**, pp.**259-298**, **2014.**

[3] M. Shardlow, *"A Survey of Automated Text Simplification.",* International Journal of Advanced Computer Science and Applications, Vol.**4,** No.**1**, pp. **58–70, 2014.**

[4] S. Wubben, E. Krahmer, A. van den Bosch, *"Sentence Simplification by Monolingual Machine Translation",* In the Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, **Republic of Korea**, pp.**1015-1024**, **2012.**

[5] T. Vu, B. Hu, T. Munkhdalai, H. Yu, *"Sentence Simplification with Memory-augmented Neural Networks",* In the Proceedings of the NAACL-HLT, **USA**, pp.**79-85**, **2018.**

[6] W. Coster, D. Kauchak, *"Simple English Wikipedia: A New Text Simplification Task.",* In the Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, **USA**, pp.**665-669**, **2011.**

[7] Z. Zhu, D. Bernhard, I. Gureych, *"A Monolingual Tree-based Translation Model for Sentence Simplification",* In the

Proceedings of the 23rd International Conference on Computational Linguistics, **China**, pp.**1353-1361**, **2010.**

[8] S. Nisioi, S. Stajner, S. P. Ponzetto, L. P. Dinu, *"Exploring Neural Text Simplification Models",* In the Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, **Canada**, pp.**85-91**, **2017.**

[9] A. Vaswani et al, *"Attention is all you Need",* In the Proceedings of the 31st Conference on Neural Information Processing Systems**, USA**, pp.**5998-6008, 2017.**

[10] S. Zhao, R. Meng, D. He, S. Andi, P. Bamabang, *"Integrating Transformer and Paraphrase Rules for Sentence Simplification",* In the Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, **Belgium**, pp.**3164-3173**, **2018.**

Authors Profile

Mr. K Mehta is a senior undergraduate student pursuing Bachelor of Technology focused in Computer Engineering at K.J. Somaiya College of Engineering, Mumbai. His main areas of interest are Machine Learning, Artificial Intelligence and Data Science. He has published one research paper in the reputed Indian journal IJCA.

Mr. H Chodvadiya is a senior undergraduate student pursuing Bachelor of Technology focused in Computer Engineering at K.J. Somaiya College of Engineering, Mumbai. His main areas of interest are Machine Learning, Deep learning applications like CNN and NLP.

Prof. S. R. Sankhe is working as Assistant Professor in the Department of Computer Engineering at K.J.Somaiya College of Engineering. Her main areas of expertise are Artificial Intelligence, Machine Learning and Natural Learning Processing. She has worked on Minor Research Proposal for Mumbai University and she has filed two Copyrights.