Neural Text Simplification

Vivek Bhave
UMass Amherst
vbhave@cs.umass.edu

Kautilya Rajbhara
UMass Amherst
krajbhara@cs.umass.edu

Jay Panchal
UMass Amherst
jpanchal@cs.umass.edu

Abstract

The task of sentence simplification aims to create a simplified, easy to comprehend sentence from a usual ambiguous or vocabulary rich english sentence which many users like non-native English speakers or language learners may find difficult to follow. In this project we apply different deep-learning based approaches to solve the problem of sentence simplification and achieve the SARI score of **0.44** on one of the approaches which is higher than the current SOTA of 0.41. Specifically, we have used transformers (with a novel objective function), a completely novel BERT-GPT2 model as well as more recently developed seq-2-seq models T5 and BART on this task. Other than the implementation we also present in depth statistical analysis and intuition of the working of all models and possible approaches which could be tried out with more computing resources.

1 Introduction

English - one of the most widely used languages can sometimes be highly ambiguous or difficult to understand. Long complex sentences, seldom used words or phrases make the problem of comprehending English literature quite difficult. Non-native English speakers, English language learners or people having disabilities like aphasia and dyslexia can sometimes face a uphill task in understanding the language. Creating a sentence simplification tool which can take a normal sentence as input and convert it into a simple and easy to understand sentence can solve this problem. In this project we aim to experiment with different methods, improving on previous approaches to solve this problem of sentence simplification.

This task like many other tasks in NLP has been traditionally approached using statistical techniques. Specifically, many attempts have been made in approaching this task as a statistical intra-language machine translation task. In recent years with the advent of deep-learning, neural network based techniques were attempted, however they didn't significantly outperform rule based methods due to lack of large volumes of high quality datasets as well as lack of generative capabilities of simpler neural network architectures (Sanqiang Zhao, 2018). However, with the advent of sophisticated sequence-to-sequence (Seq2Seq) deep-learning architectures like the Transformer (Vaswani et al., 2017) as well as availability of multiple parallel corpora for the simplification task, things have started to change. Now, more than ever, deep learning is more applicable to the sentence simplification task.

In this project we have tried various transformer based approaches [1] Although, seq-to-seq encoder-decoder models have been tried before for this task, here we have attempted to experiment with recent state-of-the-art pre-trained models like T5 (Colin Raffel, 2019) and mBART (Lewis et al., 2019) to fine-tune them for the simplification task. Furthermore, we have also tried a completely novel BERT-GPT2 architecture where we use pre-trained BERT as an encoder and pre-trained GPT2 as a decoder and we fine tune the entire pipeline for our simplification task. Additionally, in order to setup a baseline to compare all the above mentioned models, we also train a Transformer model from scratch with a simple negative log loss objective and a modified log loss objective incorporating simplification rules from simplePPDB (Pavlick and Callison-Burch, 2016)

This report has been organized as follows. Section 2 describes the previous literature related to this task. Section 3 is about the datasets used. Section 4 details on all the approaches we have tried

[1]Implementation can be found on https://github.com/kautilya96/neural-text-simplification Code has been submitted on Moodle by Kautilya Rajbhara.

for solving the sentence simplification task. Section 5 describes the SARI evaluation metric that we have used in our experiments. Section 6 discusses the results obtained on all the approaches and gives the intuition behind the results obtained. In Section 7 we suggest a few more approaches that might lead to even better results in the future. Finally, we conclude with section 8.

2 Related Work

This task has been approached in a lot of different ways over the last decade. Initially researchers applied statistical rule base methods, then deep learning architectures, later attention mechanisms and very recently transformers.

In earlier days, Sentence paraphrasing which is quite similar to the sentence simplification task was perceived as preliminary task to achieve sentence simplification. One such effort to use parapharsing techniques to solve the simplification task is Simple PPDB. PPDB (ParaPhrase DataBase) is a series of efforts made by researchers to solve the problem of sentence paraphrasing using statistical methods. First such effort was made by Ganitkevitch et al. (2013) in which they captured over a 140 million meaning preserving syntactic patterns from various bilingual corpora. Two words/phrases were considered as paraphrases if the meaning of both translates to the same word/phrase in a different language. In the following research work by Pavlick et al. (2015) these set of rules obtained were manually annotated by humans and then ranked. Every transformation pair in the PPDB included word embedding similarities, style annotations and fine grained entailment relations. Most recently, Pavlick and Callison-Burch (2016) came up with Simple PPDB a new small but effective set of paraphrase rules specially designed for the task of sentence simplification. Pavlick and Callison-Burch (2016) used a supervised learning model to map simplification scores with each sentence pair. The ranking of rules from these scores was of really high quality to perform lexical simplification. The newly developed 4.5 million rules were the largest resource for text simplification at that time created only for lexical simplification.

By 2017, many researchers started experimenting with deep-learning techniques for the sentence simplification task. Zhang and Lapata (2017) came up with a Deep Reinforcement Learning based approach in 2017 for sentence simplification. They proposed a novel combination of two individual loss functions for this task. They created a neural encoder-decoder model and integrated it in a reinforcement learning framework. The reward function of the reinforcement learning algorithm is a weighted combination of simplicity, relevance and fluency. This reinforcement loss is combined with a pre-trained encoder-decoder based objective function used to capture lexical simplification. This model gave competitive results as compared to other simpler neural-network based models on various datasets like WikiLarge, WikiSmall and Newsela.

To improve the performance of neural models, Sanqiang Zhao (2018) came up with a new loss function integrating statistical models. The authors had used a multi-layer multi-head attention transformer architecture proposed by Vaswani et al. (2017). The integration of PPDB rules is done through two models - Deep critic Sentence Simplification Model, and secondly Deep Memory Augmented Sentence Simplification Model. The deep critic sentence simplification model rewards the application of any of the rules from Simple PPDB and the augmented sentence simplification model allows recording multiple key value pairs for each rule in Simple PPDB. This integration combines the advantages of both neural models and statistical methods and helps the model to select more accurate simplification rules. This model was trained on the WikiLarge dataset and validated and tested on the TurkCorpus and seemed to outperform multiple sate-of-the-art sentence simplification models prevelant at that time.

One of the most recent works in Sentence Simplification which we referred was by the one by Fang and Stevens (2019) at Stanford University . They used similar approach to (Zhao et al., 2018) with regards to incorporating simplification rules from Simple PPDB into a Seq2Seq architecture. The main difference was in their approach and Zhao et al. was that they used a Transformer XL (Zihang Dai, 2019) architecture instead of the Transformer architecture. The author have tried two main ways of integrating the Simple PPDB rules into the training objective of the Transformer-XL model. The first one is using a modification loss, which is similar to the one used by Zhao et al.. The second one is a weighted loss, which takes into consideration the ratio of rules applicable and has a tunable hyper-parameter to adjust it's weight. The authors have trained and tested on the WikiSmall dataset and found that the weighted loss works better than the modified loss. In our implementation, we used a loss similar to weighted loss for training the baseline Transformer model on WikiLarge dataset. Section 4 on approaches discussed about this in more detail.

3 Dataset

Different datasets have been created for the task of sentence simplification over the years. We have used a total of three different datasets for our experiments - WikiLarge, Turk Corpus and PWKP (Parallel Wikipedia). Each of the datasets consists of normal-simple sentence pairs.

1. **WikiLarge** - The WikiLarge dataset [2] consists of 296,402 sentence pairs in the training set, 992 pairs in the validation set and 359 pairs in the test set. The original sentences have been extracted from articles on wikipedia and the simplified sentences are the corresponding aligned sentences on Simple Wikipedia. The large volume of data is definitely useful to train / finetune large complex models.

2. **Turk Corpus** - The Turk Corpus was released by Xu et al. (2016). The Turk corpus consists of 2000 sentence pairs for training and 350 for testing. This dataset was created with the help of human annotators on Amazon Mechanical Turk. 2350 sentences written in normal English were picked by the authors. These sentences were given to 8 people on Amazon Mechanical Turk with guidelines to construct a simpler version of them. The researchers then selected the best simplified sentence out of the eight available choices for each of the sentences and have released these sentence pairs as the Turk Corpus dataset.

3. **PWKP** - Zhemin Zhu (2010) came up with a Tree Based translation model for sentence simplification and in the process came up with the PWKP (Parallel Wikipedia) corpus. This corpus has been created from aligned sentence pairs on Wikipedia through a series of text processing mechanisms - Article Pairing, Plain text extraction, preprocessing and then sentence alignment. This corpus has close to 140,000 sentence pairs. However, in a few of the sentence pairs the normal sentence is broken down into two or more simpler sentences to make the sentence significantly lucid for the reader. We choose to remove these examples from our training set for couple of reasons. Firstly, we think that since the other datasets that we use does not contain such type of sentence pairs where an input sentence is broken down into multiple simpler sentences, allowing such examples to be a part of our training set, would not be a fair basis for comparison between the datasets. Secondly, we thought it would be difficult for the model to learn to break a complex sentence down into multiple simpler sentences from this relatively small subset of examples and may lead to poor model performance.

On careful analysis of the data from WikiLarge dataset we realized that several of the sentence pairs were not of good quality. In many of the sentences, the source and target sentences were quite unrelated to each other and had very little in common. In a few cases, the target sentence would be very short as compared to the source sentence, and thus would convey little to no information of the source sentence. However, it had about 296K sentence pairs as a whole and majority of them were of good quality. In comparison, the sentence pairs of the Turk Corpus are manually annotated and reviewed. This made the quality of the sentences very high. One downside here was the size of the dataset since Turk Corpus has only 2.3K sentence pairs which is way less than what is required by our huge Seq2Seq models. The PWKP corpus was probably the best of the three datasets having large volumes of high quality data.

4 Approaches

4.1 Transformer

Baseline Neural Network model for our sentence simplification task is based on Transformer architecture proposed in Vaswani et al. (2017). Transformer is a multi-layer and multi-head attention based seq2seq architecture. As shown in the figure 1, it consists of two main components: one encoder and one decoder. The encoder encodes the sentence in normal English using a stack of 6 layers. Each layer can be further decomposed into two sub layers: one sub-layer is a 8 head self-attention layer

[2] https://github.com/louismartin/dress-data/raw/master/data-simplification.tar.bz2

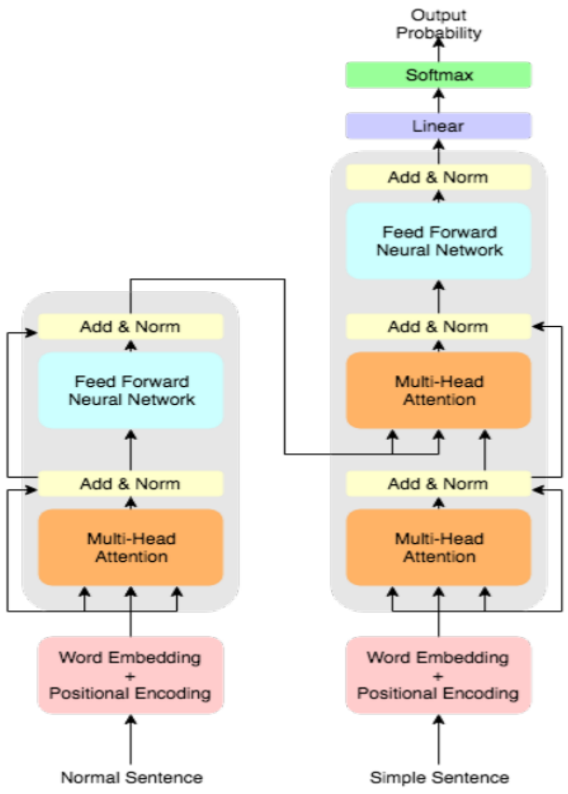


Figure 1: Architecture of Transformer Zhao et al.

and the other one is a fully connected feed-forward neural network. In the first layer, the multi-head self-attention sub-layer encodes the input sentence which is passed on to the fully connected feed-forward neural network which applies non-linear transformations on the encoded input. For every subsequent layer in the encoder, the multi-head self-attention sub-layer encodes the output of the previous layer and a fully connected neural network applies non-linear transformation to it.

The decoder on the other hand is responsible for generating simpler sentences given the encoded representation of input sentences by the encoder. Like the encoder, the decoder also consists of a stack of 6 identical layers. Here, in addition to two sub-layers present in the encoder, we have one additional sub-layer called encoder-decoder attention layer. This layer takes as input the output of multi-head self-attention sub-layer as well the output of the encoder and identifies relevant information from the encoder outputs. The model is trained to minimize the cross-entropy loss of the simple sentence,

$$CrossEntropyLoss(CEL) = -logP(O|I, \theta)$$

where θ represents all the parameters in the current model, O = sentence generated by the model and I = Input sentence in normal English.

Here we also use simplification rules froom the Simple PPDB to improve our model by incorporating paraphrasing knowledge from it into our seq2seq neural model. We implemented a method suggested in Fang and Stevens (2019) in their paper for their class project, of modifying loss function between predictions and targets as follows to incorporate simple PPDB into our model.

Let O be a sentence predicted by our model, T be its corresponding target, and N be the number of words in T. Let S be the number of words in T which have simplifications in the Simple PPDB. Then, the final loss value between O and T can be given as,

$$CL(O, T) = CEL(O, T) * (1 - \frac{S}{N}) * c$$

where $0 < c < 1$ is a hyperparameter, *CEL* is Cross Entropy Loss, *CL* is Confidence Loss.

The intuition here is that if the target T of the prediction O can be further simplified using the rules in simplePPDB, then we decrease the cross entropy loss between O and T by a multiplicative factor. This multiplicative factor is determined by the number of single word simplifications possible in the target sentence T. Thus we decrease the loss for a sentence whose target sentence in the corpus is of poor quality. We use this modified loss for training on only WikiLarge dataset since the other two dataset contains high quality data as compared to the WikiLarge dataset which contains several poorly constructed examples.

We experimented with two existing implementations of the transformer architecture. One was an openly available tensorflow implementation by Kyubyong Park [3] and the other was a library implementation by OpenNMT [4]. In our experiments with both the implementations we found that the OpenNMT's version of Transformer was much better optimized for training than Kyubyong Park's implementation. However, one downside for OpenNMT's implementation was its reduced flexibility

[3]https://github.com/Kyubyong/transformer

[4]OpenNMT is an open source ecosystem for neural machine translation and is licensed under the MIT license.

since it was a library implementation. As compared to this the Kyubyong Park's implementation offered a higher degree of flexibility to tweak the transformer architecture as per our needs. As a result, for training the transformer with negative log loss we used the default OpenNMT implementation while for incorporating the modified cross entropy loss with the simple PPDB rules we use Kyubyong Park's implementation of the transformer.

4.2 BERT-GPT2

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) is one of the recent state-of-the-art language model having the capability to generate deep bidirectional representations from both left and right contexts of text units. BERT has been trained on the BooksCorpus (800M words) and English Wikipedia (2,500M words). The pre-trained model can be fine-tuned on a variety of NLP tasks and has achieved state-of-the-art performance in many including Question Answering, Sentiment Classification, etc.

BERT is an encoder only architecture as it can only generate representations for text units. The task of text simplification requires language modeling as well as text generation. Thus, we need a sequence-to-sequence model with generative capabilities, and one of the best such models is GPT-2. GPT-2 (Radford et al., 2019) is a large transformer-based language model with 1.5 billion parameters in it's largest version, trained on a dataset of 8 million web pages. GPT-2 is trained with the objective function of predicting the next word, given all of the previous words within some text. The diversity of the dataset causes this goal to contain naturally occurring demonstrations of many tasks across diverse domains.

For the sentence simplification task we have used BERT as an encoder and GPT-2 as a decoder. BERT generates contextual representations for the complex sentences. Similar to an encoder-decoder architecture, these text representations of the complex sentence initialize the initial state of GPT-2. Then, GPT-2 generates the simple sentence word-by-word. The entire BERT-GPT2 architecture is trained on the cross entropy loss objective. For our implementation, we use the *'bert-base-uncased'* pre-trained model, which is the smallest pre-trained model released for BERT (trained on lower-cased English text)s and *'gpt2'*, which is the smallest pre-trained model released for GPT-2. Both the models being pre-trained on large English text, we presume that entire pipeline of BERT as encoder and GPT2 as decoder will help form a grammatically coherent sentence in the output. Due to time constraints we were able to evaluate this architecture only for WikiLarge dataset.

4.3 T5

T5 (Text to Text Transfer Transformer) (Colin Raffel, 2019) is a large text to text transformer model pre-trained in a multilingual setting. The encoder consists of a few blocks of which each block consists of a self-attention layer and then a small feed-forward neural network. Each layer's output is also layer normalized and includes a residual skip connection to farther output layers. The decoder has a similar architecture to the encoder with the addition of a self attention layer to the encoder.

T5 has been trained on the C4 (Colossal Clean Crawled Corpus) dataset. The dataset has been created by crawling contents on the web, which is the best technique to generate a large corpus, in this case having a size of 20 Terabytes. However, such a corpus has a lot of low quality data and hence the authors have applied several pre-processing techniques to use it as training data during the unsupervised phase. The model has been trained on 524,888 steps on unlabelled data from C4 with a maximum sequence length of 512 and a batch size of 128 sequences.

T5 uses a standard Transformer architecture (Vaswani et al., 2017). Having been pre-trained on an extremely large corpus of unlabelled data, T5 can be fine-tuned for many downstream NLP tasks. T5 has given state-of-the-art results on a wide range of tasks including question answering, general language understanding, machine translation among others.

The T5 model is available in 5 variants - Small, Base, Large, 3B and 11B consisting of 60 million, 220 million, 770 million, 3 Billion and 11 Billion parameters respectively. We decided to use the smallest variant of T5 i.e. T5 Small consisting 60 million parameter model for our sentence simplification task for two reasons. Firstly, with our resource constraints it would have taken an impractically large amount of time to fine-tune any other variant except for T5 Small, and secondly the size of our dataset was extremely small hence a larger model might have overfitted the data leading to a poor generalization. Here, we fine-tune T5 Small for our sentence simplification task using the negative log loss objective on all the three datasets, results for which are reported in the results section of this report.

4.4 mBART

mBART (Liu et al., 2020) is a Seq2Seq denoising auto-encoder pre-trained on a large scale monolingual corpora in multiple languages. It is the first known method to pre-train the complete sequence-to-sequence model primarily for machine translation tasks. mBART uses a standard seq2seq transformer architecture consisting of 12 layers of encoder and 12 layers of decoder each of which contains 16 head self-attention sublayers. It is pre-trained on a subset of 25 languages (CC25) extracted from the Common Crawl (CC) by Wenzek et al. (2019) corpus using the BART (Lewis et al., 2019) training objective. For each of the 25 languages, the input sentences are corrupted using a noise function and the model is trained to recover the corrupted sentences. Consecutive input sentences (upto 512 tokens each) in the training corpus are packed into one instance and then the Noise function is used to replace 35% of the words in each of those instances with a mask. Furthermore, order of the sentences within the instances are permuted which is then passed as an input to the encoder. The decoder on the other hand is passed the original text with offset of one position as the input. The model is then trained to reconstruct the original text given the noisy text. Models pre-trained this way can then be fine-tuned for a wide variety of machine translation tasks. Fine-tuning mBART achieves SOTA results on many low-resource (having parallel corpus with less than 1M sentence pairs) MT tasks.

Since, our sentence simplification task is based in a low resource setting with even the largest dataset that is available for simplification has only 296K sentence pairs, we hypothesize that using a large pre-trained model like mBART would help in achieving good results for the simplification task.

There two main variants of mBART that are openly available, one is the base model pre-trained on the CC25 dataset and the other one is a model further fine-tuned for the English-to-Romanian translation task. For our use case we used the base model pre-trained on CC25 dataset. The training objective we used for fine-tuning was negative log-likelihood. We used Adam optimizer with polynomially decaying learning rate initially set to 3e-05. Furthermore, to avoid overfitting to the training dataset we use dropout in both the self-attention layer as well the feed-forward layer of all the encoders and decoders. We set the feed-forward dropout and attention dropout rate to 0.3 and 0.1 as suggested in the paper.

5 Evaluation

5.1 SARI

SARI, an acronym for system output against references and input sentences, is an evaluation metric proposed by Xu et al. (2016) to measure the quality of text simplifications. SARI, measures the goodness of the words retained, dropped or added by our simplification model. Given a candidate sentence O (output of our model), human reference sentences (R) and the input sentence (I), we calculate the precision and recall of three operations: addition, retain and deletion.

For the addition operation, all the words present in the system output (O) and one of the human references (R), but not present in the input sentence (I) are rewarded i.e words that belong to $O \cap \overline{I} \cap R$ are rewarded. Precision and Recall are calculated for n-grams rather than for individual words. For addition operation for a single output sentence the n-gram precision and recall is calculated as:

$$p_{add}(n) = \frac{\#\text{ of n-grams in } (O \cap \overline{I} \cap R)}{\#\text{ of n-grams in } (O \cap \overline{I})}$$

$$r_{add}(n) = \frac{\#\text{of n-grams in } (O \cap \overline{I} \cap R)}{\#\text{of n-grams in } (R \cap \overline{I})}$$

Similar to the addition operation, for the retain operation all the words present in the system output (O), which are also present in the input (I) and one of the human references (R) are awarded. Here, the precision and recall for retaining a word is weighted depending on the number of references in which that word is retained. So, if out of r human references if only 2 of them have a word present in input(I) retained then precision and recall for that word will be multiplied by $\frac{2}{r}$. n-gram precision and recall for the retain operation can be given as,

$$p_{retain}(n) = \frac{\#\text{ of n-grams in } (O \cap I \cap R') * r_n}{\#\text{ of n-grams in } (O \cap I)}$$

$$r_{retain}(n) = \frac{\#\text{of n-grams in } (O \cap I \cap R') * r_n}{\#\text{of n-grams in } (R' \cap I)}.$$

where, r_n = fraction of references containing those n-grams & R' is a set such that R' $\subset$ R, in which the given n-grams have been retained.

Finally, all the words dropped in the system output (O) as well as dropped from one of the human references (R), but present in the input sentence (I) are rewarded. Here, only precision is measured since we don't want the model to be awarded too much for deletion as deletion hurts readability. Similar to the above case of the retain operation, precision for deletion is weighted depending on the number of references in which that word has been deleted. n-gram Precision for deletion can be given as,

$$p_{delete}(n) = \frac{\#\text{of n-grams in } (\overline{O} \cap I \cap \overline{R'}) * r_n}{\#\text{of n-grams in } (\overline{O} \cap I)}.$$

where r_n = fraction of references not containing those n-grams & R is a set such that R' $\subset$ R, in which the given n-grams have been dropped.

Now, average precision (P) and recall (R) is calculated for each operation for different orders of the n-gram. Here, the highest order n-gram we used is 4 which was the same value as suggested in their paper by Xu et al. (2016).

$$P_{operation} = \frac{1}{4} \sum_{i=1}^{i=4} p_{operation}(i)$$

$$R_{operation} = \frac{1}{4} \sum_{i=1}^{i=4} r_{operation}(i)$$

where operation $\in$ {add, retain, delete}.

For deletion we only calculate P_{delete}. From the average precision and recall, F-score for addition and retain operation is calculated as follows,

$$F_{operation} = \frac{2 * P_{operation} * R_{operation}}{P_{operation} + R_{operation}}$$

Based on the values calculated as above, SARI can be given as follows,

$$SARI = \frac{1}{3} * F_{add} + \frac{1}{3} * F_{retain} + \frac{1}{3} * P_{delete}$$

Xu et al. (2016) also report the experiments they carry out to measure how well does the SARI score correlate with the human judgement of simplification. They used human rated sentences [5] rated on a scale of 1-5 indicating varying degrees of simplification as the human judgement of simplification. They compared this with the SARI score of those sentences by plotting a graph of human scores vs SARI score and found that SARI score highly correlates with the human judgement of simplification. Hence, we incorporate SARI to evaluate the simplification accuracy of our model.

6 Analysis of Results

All the models mentioned in the approaches section above have been tested on the test set of the Turk Corpus. The test set of the Turk Corpus and the WikiLarge corpus have the same set of input sentences but different set of reference / target sentences. Here, we use reference sentences from both the WikiLarge corpus and the Turk Corpus to calculate a combined SARI score. Table 1 summarizes the average SARI scores obtained by different models on the test set of the TURK dataset. The following sections analyzes and discusses this results into more detail.

6.1 Analysis of the Best & the Worst performing Models

As shown in the Table 1, of all the models we tried the model with the highest SARI score was the T5 model fine-tuned on the WikiLarge dataset. The SARI score obtained of 0.44 is higher than the current state-of-the-art of 0.4187 as reported by Martin et al. (2019), and thus the highest ever achieved in sentence simplification on the WikiLarge Test dataset. In comparison to other models we accredit this success to the large amount of training data (296,402 sentence pairs) available in the WikiLarge dataset. The T5 model has been pre-trained as a text-to-text transformation task and hence is aptly suited for the sentence simplification task.

Also, we note that the worst SARI score of 0.16 was given by the Transformer model on the Turk Corpus (2,000 sentence pairs). We attribute this to two factors. Firstly, even though the Turk Corpus has very high quality data, the number of training examples are very less. Thus, the Transformer

[5]labelled by human annotators on Amazon's Mechanical Turk

Model	Dataset	SARI	No. of Steps
Transformer with Cross Entropy Loss	WikiLarge	0.39	50,000
	PWKP	0.36	5,000
	Turk Corpus	**0.16**	5,000
Transformer with Confidence Loss	WikiLarge	0.21	416835
T5	WikiLarge	**0.44**	2,47,200
	PWKP	0.30	3,91,200
	Turk Corpus	0.28	3,88,800
mBART	WikiLarge	0.31	1547
	PWKP	0.31	700
	Turk Corpus	0.26	598
BERT-GPT2	WikiLarge	0.32	500

Table 1: Comparison of SARI score of various models

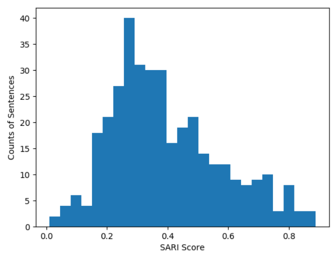
model did not have enough data to generalize well. Secondly, unlike the other models the Transformer model is trained from scratch and hence it could not learn well from low amount of data from Turk Corpus.

6.2 Performance Comparison of Datasets

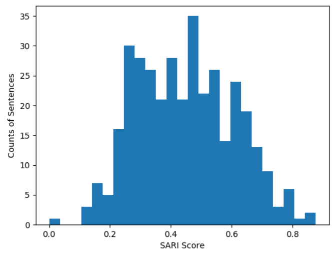
Out of the three different datasets we used, all models gave their best performance on the WikiLarge dataset. This result is expected since it contains the largest number of sentence pairs (2,96,402 sentence pairs). Next, the models gave great results on the PWKP corpus (1,01,419 sentence pairs), but the performance was lower than the performance on the WikiLarge dataset. All the models performed badly on the Turk Corpus (2,000 sentence pairs) which can be attributed to the fact that it contains the least amount of training examples. However, fine-tuning pre-trained models like T5, BERT-GPT2 and mBART gave higher SARI scores on this data as compared to the Transformer model because they can generalize better compared to the Transformer model trained from scratch.

6.3 Performance Comparison of Models

From the four models we evaluated T5 worked better than the others on WikiLarge and Turk Corpus. We attribute this to a few of the following reasons. Firstly, being pre-trained on extremely large sequence-to-sequence corpora, T5 is a highly sophisticated model as it already learns many low-level features for language encoding and generation. Secondly, since we are using the Small variant of T5 with only 60 million parameters even in a low resource setting ($\leq$ 1M sentence pairs in parallel corpus) such as ours, fine-tuning it does not



SARI Distribution for Transformer - WikiLarge



SARI Distribution for T5 - WikiLarge

Figure 2: SARI Score distributions on test set for top performing models (for all the distributions, please refer to Appendix A)

overfit the data as it is doing in mBART and BERT-GPT2. Here, however strangely the transformer outperforms T5 on the PWKP corpus. Transformers seem to work best in this case since it received enough amount of high quality data to create a good generalized model. On the other hand, T5 could not develop as good a model since the pre-trained model with a large number of parameters did not fine-tune considerably to the sentence pairs of the PWKP corpus.

mBART (610M parameters) and BERT-GPT2 (340M + 117M = 457M parameters) are extremely large models as compared to T5 Small and fine-

Model	**Dataset**	**% in 0.2-0.3**	**% in 0.3-0.4**	**% > 0.4**
Transformer with Cross Entropy Loss	WikiLarge	24.7	23.6	39.7
	PWKP	27.7	28.6	35.2
	Turk Corpus	27.7	1.9	0.5
Transformer with Confidence Loss	WikiLarge	35.0	6.6	6.5
T5	WikiLarge	16.4	20.8	58.3
	PWKP	31.3	23.0	21.1
	Turk Corpus	43.8	22.7	13.3
mBART	WikiLarge	50.0	19.4	18.0
	PWKP	46.8	29.3	14.9
	Turk Corpus	62.5	18.1	4.9
BERT-GPT2	WikiLarge	36.3	45.0	14.3

Table 2: Table shows a comparison of the percentages of test data giving high quality SARI scores. We consider a range of 0.2-0.3 SARI score to be moderate and a range of 0.3-0.4 decent and anything above 0.4 as exceptional, and thus summarize the density of results in these ranges to understand the quality of results.

tuning them requires an extremely large amount of data for the model to generalize well. Even though they were trained on the WikiLarge dataset of 296,402 training examples it was still insufficient to get good results from them since they overfitted the training set.

Transformer with confidence loss did not significantly outperform transformer with negative log loss as was intended. The goal of the confidence loss was to deal with poor quality sentences in the WikiLarge dataset. Many of the reference/target sentences were not really simple as compared to the input sentences. So, if a reference sentence in the training data can be further simplified using Simple PPDB rules we decrease the loss incurred from that example in order to punish it for its poor quality. One possible reason why results we got were not as expected, is we penalize an example only if the reference sentence consists of words that can be further simplified according to the rules in the PPDB. Here, we do not consider examples, where the reference sentences consists of phrases with length > 1, that can be further simplified according to the rules in PPDB. There may be many such examples and so our loss function is biased towards penalizing poorly constructed examples where references can be further simplified by simple word simplifications while not doing the same for references which can be simplified by phrase (length > 1) level simplifications.

6.4 Statistical Analysis

In this section, we try to compare the quality of SARI score performance of all the models. Table 2 shows a comparison of percentages of test data examples lying in specific ranges of SARI scores to understand the quality of text simplifications done by the models.

The range of SARI scores from *0.2-0.3*, are moderate to good simplifications. The second range of SARI scores from *0.3-0.4* are decent simplifications and anything above 0.4 are considered as exceptional simplifications performing close to gold standard simplifications.

For Transformer model with normal cross entropy objective, test outputs for model trained on WikiLarge and PWKP have the highest percentage of sentences about more than 35% of sentences with SARI scores > 0.4 i.e. in the exceptional range and about 48%-55% of sentences in the moderate and decent range. This is reflected in the average SARI score of these models in table 1. So, transformer performs quite well on these two datasets. On the other hand, for the Transformer model trained on Turk Corpus only about 30% sentences have SARI score above 0.2. So, this model performs really bad since most of its predictions i.e about 70% sentences have SARI score less than 0.2. For Transformer model trained on WikiLarge dataset with the proposed confidence loss, majority of its predicted sentences i.e. about 35% lies in the range of *0.2-0.3*. So, its predictions are decent but they are not upto the mark with the gold standard reference simplifications as well as the outputs of the Transformer trained with negative log loss objective on WikiLarge and PWKP.

In the case of T5 trained on WikiLarge, 95% of its predictions have SARI score more than 0.2 with

Model	**Output**
Complex Sentence	The Prime Minister stays in office only as long as he or she **retains** the support of the lower house .
Gold Standard Reference Sentence	The Prime Minister stays in office only as long as he or she **keeps** the support of the lower house .
Transformer - WikiLarge	The Prime Minister stays in office only as long as he or she **has** the support of the lower house .
Transformer - PWKP	The Prime Minister stays in office only as long as he **has** the support of the lower house .
Transformer - Turk Corpus	the slip was released in three formats for free .
Transformer with Modified Loss - WikiLarge	The **Governor-General** is the highest elected office in the world .
T5 - WikiLarge	The Prime Minister stays in office only as long as he or she **has** the support of the Lower House.
T5 - PWKP	The prime minister stays in office only as long as he or she **retains** the support of the lower house .
T5 - Turk Corpus	the prime minister , or she **keeps** the support of the lower house .
mBART - WikiLarge	The Prime Minister stays in office only as long as he or she retains the support of the lower house .
mBART - PWKP	the prime minister stays in office only as long as he or she retains the support of the lower house.
mBART - Turk Corpus	the prime minister stays in office only as long as he or she retains the support of the lower house .
BERT-GPT2 - WikiLarge	the prime minister stays in office only as long as he or she retains the support of the lower house.

Table 3: Outputs from different models on the same sentence comparing their translations

about 58.3% sentences having SARI well above 0.4. This is well reflected on its average SARI score of about 0.44 which as per our knowledge beats the current state-of-the-art of 0.41. Since majority of the mass lies in the range above 0.4, the quality of its predictions is exceptional and of all our models its outputs most closely matches that of the gold standard. For T5 trained on PWKP, 75% of its predictions lie above 0.2 and majority of this 75%, about 31.3%, lies in the range of *0.2-0.3* indicating most of its predictions are moderate to good. So, T5 trained on PWKP gives a mediocre performance as compared to that trained on WikiLarge. As compared to both these T5 trained on Turk Corpus performs quite average since 43.8% of its predictions are in the mediocre range of *0.2-0.3*. Average SARI score of 0.28 (Table 1) indicates this.

Training mBART on any of the three datasets, we find that around half of them in the case of WikiLarge and PWKP and more than half of them in the case of Turk Corpus are in the SARI score range of *0.2-0.3*. So, most of the predictions that mBART makes across all the three datasets is of mediocre quality which is reflected by its moderate average SARI score in table 1. Here, however it is noteworthy that in all the three models, more than 80% of its predictions have SARI score above 0.2, so the model did manage to learn to simplify the sentences but because of overfitting it could not generalize well as was expected.

Finally, the BERT-GPT2 model can simplify 45% of the sentences of the test set with a SARI score of 0.3 to 0.4. Also about 95% of its predictions have SARI score of more than 0.2. So, although this model performs quite decently but similar to mBART suffers from overfitting limiting its generalizability. For model detailed understanding of the distributions, refer to 3 in Appendix A.

6.5 Qualitative Analysis

In this section we analyze the outputs of various models from qualitative standpoint. Table 3 shows a sample input sentence from WikiLarge Corpus's test set along with its gold standard reference followed by the outputs of our various models. For the given sentence, the gold standard reference replaces the relatively complex word *retains* with a much simpler word *keeps*.

Here, the output of Transformer trained on WikiLarge and PWKP is remarkably good since it not only retains all of the information in the input sentence but also manages to replace the word *retains* with a much simpler word *has* which is even better than the gold standard. As opposed to these two models, Transformer trained on Turk Corpus performs quite poorly since it outputs quite irrelevant sentence which is neither simpler nor it preserves any information from the input sentence. This is in accordance with our quantitative analysis about this model in the Sections 6.3 and 6.4. Transformer trained with the confidence loss also outputs relatively irrelevant sentence. Here, however it is noteworthy that it somehow manages to understand the context of the sentence since it talks about the post of Governor-General which is a political post similar to the post of Prime-Minister in our input sentence.

In the case of T5, the one trained on WikiLarge like its Transformer counterpart performs exceedingly well by not only preserving the meaning of the input sentence but also replacing the complex word *retains* with a simpler version of it *has*. In contrast to this, T5 trained on PWKP, outputs the same sentence as its input without replacing the complex word *retains* with a simpler word as desired. T5 trained on Turk Corpus, does somehow manage to replace the complex word *retains* with a much simpler word *keeps* however the sentence it generates is incoherent and this severely affects its understanding.

mBART model for all the three datasets as well as the BERT-GPT2 model, as discussed in the above sections as well, clearly suffers from overfitting and outputs the same sentence as input.

7 Future Work

Apart from the approaches listed above we also tried round-trip translation technique for our simplification task. Machine translation models can be used to paraphrase text by translating it to an intermediate language and back (round-trip translation). This technique has worked well for the paraphrasing task, which is similar to text simplification. This is because of the fact that paraphrasing also deals with sequence-to-sequence generation in the same language. We experimented with an implementation from the Facebook AI Research (FAIR) group which uses a Mixture of Experts translation model Tianxiao Shen along with the round-trip translation. For our task, we use French as the intermediate language. FAIR also released pre-trained transformer models for the English-French translation and French-English translation, which we load for our sentence simplification architecture. This paraphrasing architecture generates multiple predictions (can be referred in 4 of Appendix A), which we propose to rank using the SARI evaluation metric and pick the one with the best SARI score. However,due to time constraints we were not able to fully evaluate how well this technique works for the sentence simplification task but in future it would be really interesting to see how it works.

Additionally, as reported in the results table 1 in the above section we were able to fine-tune mBART and BERT-GPT2 for fewer number of steps than the other models. This is because both BERT-GPT2 and mBART are very large models with millions of parameters and to fine-tune on top of them requires long GPU-hours. We had access to only one 12GB titanx GPU. As a reference, using this GPU for fine-tuning mBART on WikiLarge dataset consisting of 296K normal-simple sentence pairs it took us about 2 days. So our ability to try different hyper-parameters as well as further fine-tune both these models was extremely limited because of resource constraints. So, in future with access to appropriate amount of resources we would like to try different values of hyper-parameters as well as fine-tune these models with different objective functions by incorporating simplification rules from Simple PPDB to make them work better for the simplification task.

8 Conclusion

In this project, we evaluated various Seq2Seq models for the sentence simplification task. This report compares and contrasts their performance qualitatively and quantitatively. We found interesting text simplifications achieved by our models, like in one of the test example, the word ‘conceived’ was replaced by the phrase ‘thought of the idea’ by the T5 model. We found that pre-trained models like T5, mBART and BERT-GPT2 could achieve good grammatical coherence even under small fine-tuning dataset. Transformer model trained from scratch did not always preserve the grammatical structure. Also, a larger training dataset like WikiLarge helped to achieve better text simplifications than the other datasets. And finally, we achieved a new high SARI score using the T5 model which beats the current state-of-the-art model (Martin et al., 2019). This proves the suitability of pre-trained models like T5 to achieve great results in a new setting. It not only opens up new possibilities for challenging tasks in current NLP research, but also takes another step closer to the ‘one model for all’ ideology.

References

Adam Roberts Katherine Lee Sharan Narang Michael Matena Yanqi Zhou Wei Li Peter J. Liu Colin Raffel, Noam Shazeer. 2019. Exploring the limits of transfer learning with aunified text-to-text transformer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Fei Fang and Matthew Stevens. 2019. Sentence simplification with transformer-xl and paraphrase rules. In *Stanford Class project - Winter 2019*.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation.

Louis Martin, Benoît Sagot, Éric de la Clergerie, and Antoine Bordes. 2019. Controllable sentence simplification.

Ellie Pavlick and Chris Callison-Burch. 2016. Simple PPDB: A paraphrase database for simplification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 143–148, Berlin, Germany. Association for Computational Linguistics.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, Beijing, China. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Daqing He Saptono Andi Parmanto Bambang Sanqiang Zhao, Rui Meng. 2018. Integrating transformer and paraphrase rulesfor sentence simplification.

Michael Auli Marc’Aurelio Ranzato Tianxiao Shen, Myle Ott. 2019. Mixture models for diverse machine translation: Tricks of the trade.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2019. Ccnet: Extracting high quality monolingual datasets from web crawl data.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, Copenhagen, Denmark. Association for Computational Linguistics.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *EMNLP*.

Iryna Gurevych Zhemin Zhu, Delphine Bernhard. 2010. A monolingual tree-based translation model for sentence simplification.

Yiming Yang Jaime Carbonell Quoc V. Le Ruslan Salakhutdinov Zihang Dai, Zhilin Yang. 2019. Transformer-xl: Attentive language modelsbeyond a fixed-length context.

A Appendix

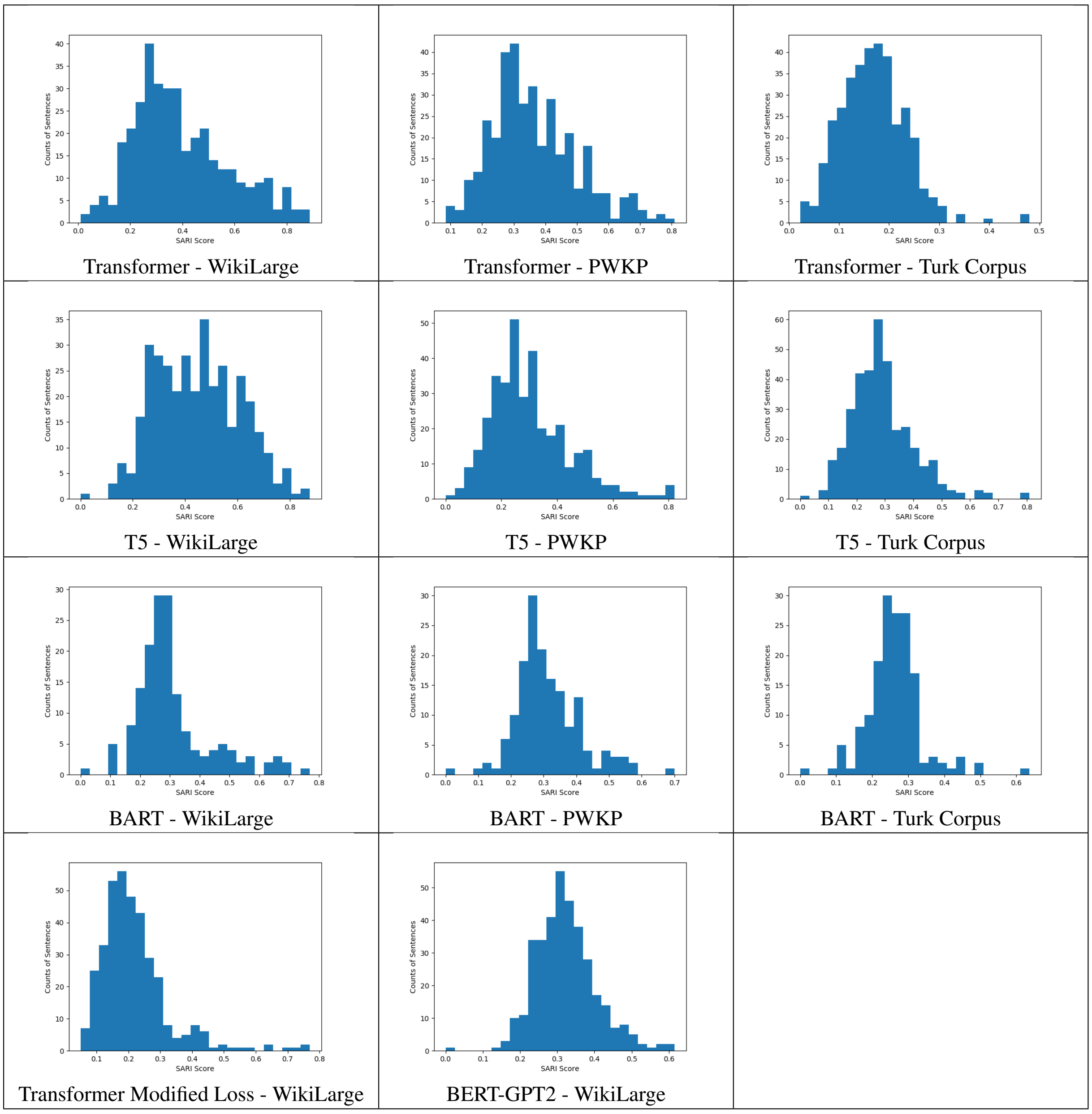


Figure 3: Histograms for distribution of SARI scores on test set by different models

	Sentences
Input Sentence	The Great Dark Spot is thought to represent a hole in the methane cloud deck of Neptune.
Model output using 10 experts	1. It is believed that the Great Dark Spot represents a hole in Neptune's cloudy methane deck. 2. The Great Dark Spot is believed to represent a hole in Neptune's massive methane cloud deck. 3. The Great Dark Spot is believed to represent a hole in Neptune's methane cloud deck. 4. Big Dark Spot is believed to represent a hole in Neptune's methane cloud deck. 5. The Great Dark Spot is believed to represent a hole in the cloudy methane deck of Neptune. 6. The Great Dark Spot is believed to represent the hole in Neptune's methane cloud deck. 7. The Great Dark Spot is believed to represent a hole in Neptune's methane cloud deck. 8. The Great Dark Spot is believed to represent a hole in Neptune's methane cloud deck. 9. The Great Dark Spot is thought to represent a hole in the Neptune methane cloud deck. 10. The Great Dark Spot is believed to represent a hole in the cloudy apron of Neptune methane.

Table 4: Output of round-trip translation technique