UNIVERSITY OF SOUTHAMPTON

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Text Simplification with Deep Neural Network Using Knowledge Transfer

by

Wei He

A thesis for the degree of
Doctor of Philosophy

June 2023

University of Southampton

Abstract

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Doctor of Philosophy

Text Simplification with Deep Neural Network Using Knowledge Transfer

by Wei He

Text simplification aims to rephrase complex text into simpler text, where the text we are mainly considering is the English text sentences. Transfer learning from pre-trained text embeddings and models has recently shown great success on a range of natural language processing tasks and is, therefore, a focus method for our work.

This thesis's first focus is to avoid using parallel corpus with sentence pairs. We propose an unsupervised method to overcome the need for parallel data and similarity constraint loss for preserving the original meaning. Moreover, an asymmetric denoising technique is adopted better to learn various features from sentences with different complexity. The results demonstrate that the denoising method can improve the performance, and the content similarity constraint can help preserve the content in our unsupervised method.

The second focus of this thesis is to define a novel approach to refining the existing noisy parallel datasets available for text simplification. After refining the dataset, our approach involves fine-tuning a pre-trained language model with a new proposed tuning strategy and decoding with a task-specific strategy. Our data refining method can generate a better dataset for the text simplification task, and the proposed fine-tuning strategy will accelerate model convergence. Moreover, the decoding strategy can greatly improve the model's performance.

The third focus of this thesis is to propose a prompting-based method without model fine-tuning. The proposed method transfers the text simplification task to the text denoising task with adaptive prompts. Our decoding vocabulary constraint technology also makes the output sentence simplicity controllable. The extensive experiments show that our proposed methodology can achieve state-of-the-art results considering many of the automatic evaluation metrics.

Contents

List of Figures

List of Tables

Declaration of Authorship

I declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as: Wei He, Katayoun Farrahi, and Adam Prugel-Bennett. Text simplification using pre-trained language models without fine-tuning. In *Submitted to The 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*, 2023

Signed:..... Date:.....

Acknowledgements

I would like to express my sincere gratitude to both my supervisors, Dr Katayoun Farrahi and Prof Adam Prugel-Bennett. Without their valuable guidance and feedback, none of the featured work would have been possible. I am also grateful to the Electronics & Computer Science Department Doctoral Training Partnership (ECS DTP), for providing me with the financial support for my research studies.

This thesis is dedicated to my parents.

Definitions and Abbreviations

Mathematics

x	Input sentence
$f_{prompt}(\cdot)$	Prompt function
I	System input
O	System output
R	Reference
$O \cap \bar{I}$	The output which is not in the Input
$\#g(\cdot)$	The binary indicator of the occurrence
V	Vocabulary set
w	Weight vector
$\exp()$	Exponential function
e	Euler's number
$p(\boldsymbol{x} \mid \boldsymbol{y})$	The conditional probability distribution of x given y
$\boldsymbol{X} = \left\{\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(n)}\right\}$	Input dataset
θ	Model parameters
$X \sim p(\cdot)$	The distribution of the random variable X is $p(\cdot)$
$\mathbb{E}$	Expectation
$\mathcal{L}(\cdot)$	Loss function
$\boldsymbol{T}$	Sentence style transformation
$< \cdot, \cdot >$	Dot product
$\| \cdot \| * \| \cdot \|$	Cross product
$Cosine(\cdot, \cdot)$	Cosine similarity function
$\text{noise}(\cdot)$	Noise function
$Lev(\cdot, \cdot)$	Edit distance (Levenshtein distance) function

Models

AE	Autoencoder
AAE	Adversarial autoencoder
BART	Bidirectional and Auto-Regressive Transformers
BERT	Bidirectional Encoder Representations from Transformers

CNN	Convolutional neural network
GAN	Generative adversarial network
GPT	Generative Pre-Training
GloVe	Global Vectors for Word Representation
GRU	Gated recurrent unit
LSTM	Long short-term memory
RNN	Recurrent neural network
SBert	Sentence-Bert
VAE	Variational autoencoder

Chapter 1

Introduction to Automatic Text Simplification

Text simplification (TS) is a natural language processing (NLP) approach of reducing the linguistic complexity of text to improve its readability. TS has the potential to assist a wide range of people in reading, such as those with dyslexia, non-native speakers, children, and non-experts in specialized areas (Alva-Manchego et al., 2020c). It will rewrite the original text using simpler words and syntactic structures to generate a new, more easily understandable text. It focuses on preserving the original content and meaning as much as possible while reducing the linguistic complexity of the text. In this thesis, we consider the context of text to be at the sentence level.

We can see many applications in our daily life embedding TS as an important component (Al-Thanyyan and Azmi, 2021), such as the online writing assistant Grammarly [1]. Some content-based applications are also built with TS techniques, such as Simple Wikipedia [2] and the Newsela online education platform [3]. Moreover, in some natural language processing (NLP) tasks, TS

[1] https://www.grammarly.com/
[2] https://simple.wikipedia.org/
[3] https://newsela.com/

can be employed in data augmentation and other pre-processing and post-processing stages (Siddharthan, 2014). It can also help other NLP tasks, such as text summarization (Vanderwende et al., 2007) and machine translation (Tyagi et al., 2015; Štajner and Popović, 2019), achieve better performance (Al-Thanyyan and Azmi, 2021).

TS is similar to other NLP tasks, such as machine translation and text summarization. These tasks can all be viewed as text sequence-to-sequence (Seq2Seq) tasks and therefore similar techniques can be shared among them. For example, the increasing application of translation methods to TS makes it a "monolingual translation". However, there are many differences between TS and other NLP tasks. For example, text summarization will significantly reduce the original's length and content, but TS does not aim to reduce the content and may generate longer output for easier understanding.

Studies before the data-driven time are commonly built to simplify text in explicit phases: 1. to replace complex words with simpler equivalents (lexical simplification), 2. to adjust complex sentence structure (syntactic simplification), and 3. to paraphrase the text (Al-Thanyyan and Azmi, 2021). However, identifying the complex parts of the sentence is difficult and differs from one sentence to another. Recently, deep learning-based methods make the automatic text simplification task easier as the model will learn to conduct the above phases implicitly. In this thesis, we will focus on deep neural network transfer learning for automatic text simplification. We will use the pre-trained embeddings, pre-trained language models (PLMs), and fine-tuned models from other tasks as part of our methodology.

Sequence-to-Sequence models and their variations have dominated recent deep-learning-based text simplification systems. Most supervised models are trained on two kinds of widely used parallel corpora datasets: Wikipedia dataset (Xu et al., 2016a) and Newsela (Xu et al., 2015). Parallel datasets for TS normally contain pairs of sentences with a complex sentence called the source sentence and a simplified counterpart called the target sentence,

Source	Plays and comic puppet theater loosely based on this legend were popular throughout Germany in the 16th century , often reducing Faust and Mephistopheles to figures of vulgar fun.
Target	Some of the plays and comic puppet theater from around the 16th century make up their own versions of the story. They often show Faust as a figure of vulgar fun.
Source	Admission to Tsinghua is extremely competitive.
Target	Entrance to Tsinghua is very very difficult.
Source	The Suprematists also made architectural models in the 1920s which offered a different conception of socialist buildings to those developed in constructivist architecture.
Target	The suprematists also made architectural models in the 1920s which offered a different conception of socialist buildings to those developed in constructivist architecture.
Source	This quantitative measure indicates how much of a particular drug or other substance (inhibitor) is needed to inhibit a given biological process (or component of a process, i.e. an enzyme, cell, cell receptor or microorganism) by half.
Target	This quantitative measure indicates how much of a drug or other substance is needed to inhibit a biological process by half.

TABLE 1.1: Examples of the parallel TS dataset Wikilarge with pairs of source and target sentences.

such as examples in Table 1.1. The Wikilarge (Xu et al., 2016a) dataset has been considered the benchmark for training and evaluating text simplification systems. However, the dataset has automatic sentence alignment errors, massive inadequate simplifications, and poor generalization. It means that the scarcity of high-quality parallel data is the main limitation of supervised methods. We deal with this problem by adopting an unsupervised method in Chapter 3, refining the dataset in Chapter 4, and developing a zero-shot method in Chapter 5.

With the boom of large-scale, pre-trained language models (PLMs), the pre-train and fine-tuning paradigm has dominated various downstream NLP tasks recently (Devlin et al., 2018; Lewis et al., 2019). Consequently, there has been an increasing interest in works and methods based on pre-trained language models in TS (Martin et al., 2020c; Lu et al., 2021; Martin et al., 2020e). Our work is also not an exception; in Chapter 4, our method follows this

paradigm by fine-tuning a Seq2Seq pre-train language model BART (Lewis et al., 2019) , whose architecture naturally fits to our task, with elaborately refined parallel sentences. The fine-tuning process will also need massive computing resources and deployment experiments as the scale of PLMs continue to surge to billions of parameters (e.g. GPT-3 (Brown et al., 2020b)). Moreover, fully fine-tuning may harm natural language understanding of original PLMs and has been shown to be unnecessary (Brown et al., 2020b). To alleviate these issues, we use PLMs by modifying TS tasks with prompts to language model tasks without updating the parameters in Chapter 5. Our prompting-based method achieves state-of-the-art results considering many metrics without fine-tuning.

1.1 List of Contributions

In summary, in Chapter 2, we review the development of text simplification with deep learning neural networks and the related techniques in this thesis. We cover "Before data-driven text simplification methods" (Section 2.2.1), deep learning based methods (Section 2.2.2.4), and Pre-trained model (PTM) based methods (Section 2.2.2.2).

The list of novel contributions in **Chapter 3** are as follows.

- We propose an adversarial auto-encoder network for unsupervised text simplification.
- We propose a similarity loss between simple and complex sentence latent representation, which can effectively preserve the original content from the simplification.
- We design a specific asymmetric denoising technique that can effectively assist our unsupervised simplification method. It can allow our model to learn various features from different sentence complexities.

The list of novel contributions in **Chapter 4** are as follows. These contributions are being submitted to a conference for publication.

- We propose a parallel data refining method based on sentence embedding similarity for TS task. The method is effective and can be easily transferred to other related tasks.
- We propose a new fine-tuning strategy, which can speed up the fine-tuning process.
- We propose a new decoding strategy for TS; it can boost the model performance and be easily extended to other related tasks.

The list of novel contributions in **Chapter 5** are as follows. These contributions are currently under review at The 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)[4].

- We propose a prompt-based method in cooperation with pre-trained models without fine-tuning.
- Our method circumvents the data-scarce problem, and it is easy to implement.
- The proposed method is state-of-the-art in many evaluation metrics.

Chapter 6 concludes our work and discusses open problems and potential future work directions.

[4]https://2023.emnlp.org/

Chapter 2

Background and Related Work

In this chapter, I present some background information and provide a literature review on text simplification. I cover the previous works in text simplification and discuss some of the methods I consider in this thesis. I begin by giving a detailed background introduction in Section 2.1, including text simplification task-specific evaluations and datasets. Second, Section 2.2 categorizes recent text simplification methods. Finally, other related methods are additionally reviewed in Section 2.3, and related training-free techniques are reviewed in Section 2.3.4.

2.1 Background

Text simplification is to make the original sentence easier to understand while preserving its original meaning. In this section, I first introduce the evaluation metrics for the text simplification task in Section 2.1.1. Then in Section 2.1.2, I introduce the widely used datasets for the TS task. Finally, I introduce the fundamental decoding strategy of our methods in Section 2.1.3.

2.1.1 Evaluation

2.1.1.1 Human Evaluation

Human evaluation is considered the ideal method for evaluating simplification quality. It usually asks experts to evaluate the system outputs in terms of grammaticality, meaning preservation, and simplicity with Likert scales (1-5 or 1-3). However, human evaluation is not easy to obtain, and it is a subjective measure. Moreover, experts may differ from one another in some views, which makes the results inconsistent in some cases for different evaluators.

2.1.1.2 Automatic Evaluation

Like other text generation tasks, automatic evaluation metrics are more widely used in TS. It can also be divided into reference-less and reference-based metrics. This section describes five widely used automatic metrics: SARI, BLEU, FKGL, Lexical Complexity Score, and Exact Match score. I use the *Easier Automatic Sentence Simplification Evaluation* (EASSE) framework[1] to calculate the metrics above. I also introduce factual consistency to measure content preservation performance. It is worth noting that multiple sentence references are necessary in order to calculate reference-based metrics. It is difficult to optimize the reference-based metrics while training because the training sets in use do not have sufficient references.

Reference-based Metrics Like related Seq2Seq tasks, the primary automatic metrics, such as SARI (system output against references and against the input sentence), for TS are based on human generated references of each source sentence. I describe these metrics in more detail.

BLEU

[1] https://github.com/feralvam/easse

BLEU (Bilingual Evaluation Understudy) is a precision-oriented metric that has been widely used to evaluate machine translation systems (Papineni et al., 2002). It calculates the matches between a system's generation and references of 1-to-n grams. The definition of BLEU is as follows:

$$BLEU = BP \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.1}$$

where BP is the brevity penalty for the scene that candidates length c is less than the total length of references r. It is defined as

$$BP = e^{1-\frac{r}{c}}$$

p_n is the match precision at n-gram, and w_n is the positive weight in which all N weights sum to 1.

The more sentence references available, the more n-grams of input will be counted. This means that the BLEU score favours outputs close to the input rather than simplified with many changes. BLEU is an metric for text-to-text generation, despite it being controversially not well suited for evaluating simplicity in the lexical point of view (Xu et al., 2016a) and penalizing simpler sentences (Sulem et al., 2018).

SARI

SARI (system output against references and against the input sentence) is a text simplification benchmark metric introduced by Xu et al. (2016a), which has been considered the most important metric in TS. It measures the efforts of simplicity by comparing the model's output to references and the original sentence based on three aspects of word operations: add, delete and keep. SARI shows a high correlation with human evaluation results (Xu et al., 2016a). Overall, the SARI score can be represented as follows:

$$SARI = \frac{1}{3}F_{add} + \frac{1}{3}F_{keep} + \frac{1}{3}P_{del} \tag{2.2}$$

where

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

$$P_{\text{operation}} = \frac{1}{k} \sum_{n=[1,\ldots,k]} p_{\text{operation}}(n)$$

$$R_{\text{operation}} = \frac{1}{k} \sum_{n=[1,\ldots,k]} r_{\text{operation}}(n)$$

where

$$p_{\text{operation}} \in [p_{del}, p_{keep}, p_{add}] \text{ and } \mathrm{r}_{\text{operation}} \in [r_{del}, r_{keep}, r_{add}]$$

and k is the highest n-gram order and is normally set to 4.

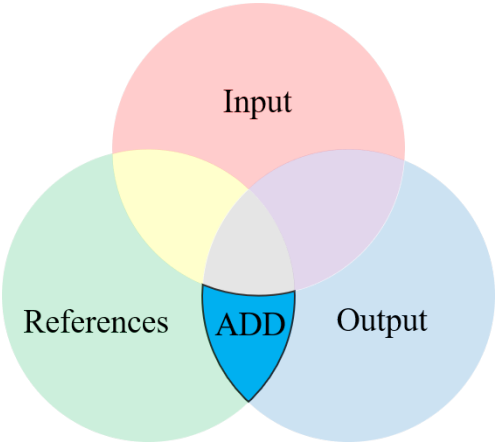
The addition operations for n-gram precision $p(n)$ and recall $r(n)$ are defined as follows:

$$p_{\text{add}}(n) = \frac{Valid\ Add}{\sum_{g \in O} \#_g(O \cap \bar{I})} \tag{2.3}$$

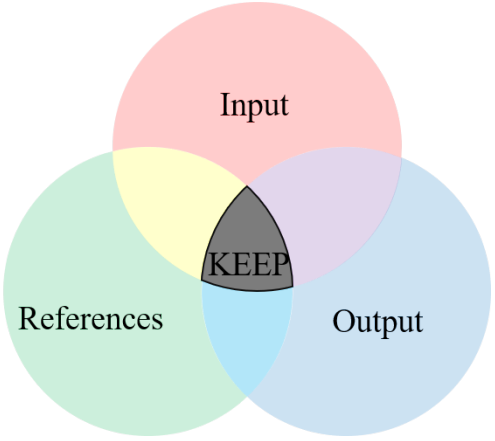
$$r_{\text{add}}(n) = \frac{Valid\ Add}{\sum_{g \in O} \#_g(R \cap \bar{I})} \tag{2.4}$$

It defines $O \cap \bar{I} \cap R$ for output O was not in the input I but appeared in any of the references R. *Valid Add* is defined as $\sum_{g \in O} \min\left(\#_g(O \cap \bar{I}), \#_g(R)\right)$, which is illustrated in 2.1a, where $\#g(\cdot)$ is a binary indicator of the occurrence of n-grams g in a given set and

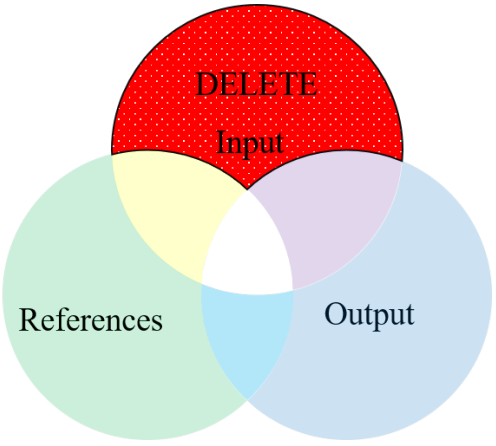
$$\#_g(O \cap \bar{I}) = \max\left(\#_g(O) - \#_g(I), 0\right)$$



(A) Valid Add of SARI



(B) Valid Keep of SARI



(C) Valid Delete of SARI

FIGURE 2.1: Key operations of SARI definition. As the figures illustrate, there are many overlap n-grams among Input, system output and references. Highlights represent *valid* operations in SARI score calculation.

$$\#_g(R \cap \bar{I}) = \max\left(\#_g(R) - \#_g(I), 0\right).$$

It rewards output that is not in the input but in the references. Some references are not necessarily complete simplifications. That is, some complex words in references may directly copy from the input without any simplification. SARI will not reward these words even if they are in references. In this way, SARI can calibrate those references having excessive copying words.

For the keep operation, it rewards words that are kept in both the outputs and references and defines:

$$p_{\text{keep}}(n) = \frac{Valid\ Keep}{\sum_{g \in I} \#_g(I \cap O)} \tag{2.5}$$

$$r_{\text{keep}}(n) = \frac{Valid\ Keep}{\sum_{g \in I} \#_g\left(I \cap R'\right)} \tag{2.6}$$

where R' is the weighted R. The weight is calculated by n/r. n is the number of n-gram occurrences out of the total r references. And

$$\#_g(I \cap O) = \min\left(\#_g(I), \#_g(O)\right)$$

$$\#_g\left(I \cap R'\right) = \min\left(\#_g(I), \#_g(R)/r\right).$$

Where *Valid Keep* is defined as $\sum_{g \in I} \min\left(\#_g(I \cap O), \#_g I \cap R'\right)\Big)$ and it is illustrated in 2.1b. It takes n-grams that are unnecessary to be simplified into account.

For the delete operation, SARI only uses precision for prevent over-deleting. It uses the following operation:

$$p_{\text{del}}(n) = \frac{Valid\ Delete}{\sum_{g \in I} \#_g(I \cap \bar{O})} \tag{2.7}$$

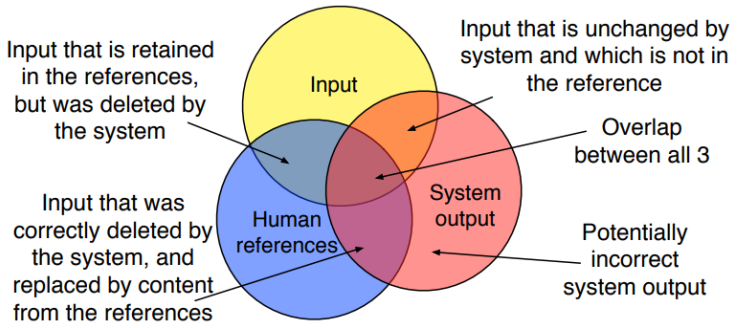


FIGURE 2.2: SARI illustration from Xu et al. (2016a)

where *Valid Delete* is defined as $\sum_{g \in I} \min\left(\#_g(I \cap \bar{O}), \#_g(I \cap \overline{R'})\right)$ and illustrated in 2.1c, and

$$\#_g(I \cap \bar{O}) = \max\left(\#_g(I) - \#_g(O), 0\right)$$

$$\#_g(I \cap \overline{R'}) = \max\left(\#_g(I) - \#_g(R)/r, 0\right)$$

In this way, words that are kept by mistake by human editors are compensated by weighting n-gram counts in $\overline{R'}$.

SARI considers the differences between input, output and references by taking both precision and recall into account. It use multiple human references to capture simplification operations in various ways. BLEU does not take recall into account and ignores differences between the input and the references. It is more suitable for evaluating the results' meaning preservation and grammaticality. SARI demonstrate correlation with simplicity scores rated by humans, especially when with multiple references, it will further improve the correlations (Xu et al., 2016b). It is also demonstrated that SARI is better suited for evaluating the simplicity of outputs via lexical paraphrasing (Alva-Manchego et al., 2021).

Reference-free Metrics **FKGL** (Flesch-Kincaid Grade Level) was designed for the U.S. Navy research in 1975 (Kincaid et al., 1975). It was initially used

for assessing the difficulty of technical manuals by the army. It is now used to estimate the readability of texts. The formula to calculate Flesch-Kincaid Grade Level is as follows:

$$0.39\left(\frac{\text{total words}}{\text{total sentences}}\right)+11.8\left(\frac{\text{total syllables}}{\text{total words}}\right)-15.59 \tag{2.8}$$

It shows that the number of words and syllables indicates the difficulty of a sentence. The score corresponds with the grade level of the US education system. A higher score indicates lower readability. For example, an FKGL score of 9 means that the text needs a reader with 9-grade level education to understand it. Despite the popularity in practice, one shortage of this evaluation metric is that short sentences could get low scores, even if they are ungrammatical or preserve meaning poorly (Wubben et al., 2012).

Lexical Complexity Score The lexical complexity (LC) score is computed by taking each word log-ranks of a sentence in a frequency table. The frequency table rates 37,058 generally known English words and 2896 two-word expressions, which has been proposed by Brysbaert et al. (2014).

Exact Match Score The exact match (EM) score shows the proportion of exact matches, i.e., the number of sentences that have been directly copied from the source without any modification.

Factual Consistency The factual consistency is used in summarization tasks to determine if a summary agrees with the facts in the source document. It will report factual accuracy as a precision measure. I use a model-based evaluation, factCC (Kryściński et al., 2019), to measure factual consistency as a proxy to measure the content preservation, higher factual accuracy stands for better content preservation performance. The factCC model is based on a BERT-based approach to train models that can identify factual consistent sentences.

2.1.2 Datasets

Lack of sufficient training data has always been a problem in the neural TS field. The data for training a neural TS model normally needs parallel sentences. In other words, each sample should consist of a pair of sentences, including a source and a target. However, generating simple counterparts of complex texts usually requires human interference, which can be pretty costly. Some training datasets are introduced in this section, and evaluation datasets will also be listed in the following sections.

2.1.2.1 Training Datasets

The training sets are typically built in two ways: first, aligning sentences from corresponding websites, such as Wikipedia and Simple Wikipedia[2]; second, gathering text from children or language education books. Both ways normally require human expertise involvement.

WikiLarge WikiLarge is a widely used parallel dataset for supervised training that was compiled by Zhang and Lapata (2017). The training set of WiKiLarge combines three datasets:

- Parallel Wikipedia Simplification (WikiSmall) (Zhu et al., 2010)
- Aligned sentence pairs from Kauchak (2013)
- Aligned sentence pairs from Woodsend and Lapata (2011).

The WikiSmall was treated as a benchmark for training and evaluating text simplification models. It consists of parallel sentences from English Wikipedia and Simple English Wikipedia. Simple English Wikipedia is an online encyclopedia for children and adults learning English. Its articles contain fewer

[2]https://simple.wikipedia.org/

complex words than English Wikipedia, and its grammar is simpler. WiKi-Large has 296,402 sentence pairs, including 2000 pairs used as a validation set and 359 pairs used as a test set, both of them are derived from TurkCorpus Xu et al. (2016a). The summary statistics are described in Table 2.1.

Newsela Newsela corpus [3] was first introduced by Xu et al. (2015). The authors argued that the Newsela corpus has better simplification than simple-Wikipedia-based datasets. Newsela is a platform providing reading materials for education. It organized professional editors to produce these simplifications for children of different grade levels. Zhang and Lapata (2017) made new alignments by removing some "too similar" sentence pairs to make the source and target distinct to each other. In the end, the newly aligned dataset comprises 94,208 pairs of sentences for training, 1,129 pairs for development, and 1,076 pairs for testing. The test set contains one simplification reference per complex sentence. However, the publisher does not allow researchers to share splits of the data publicly. So it limits its reproducibility and comparison among models.

TABLE 2.1: Training datasets summary in number of pairs

	Train set	Validation set	Test set
WikiLarge	289,043	2,000	359
Newsela	94,208	1,129	1,076

2.1.2.2 Evaluation Datasets

SARI has been regarded as one of the most important evaluation metrics in recent studies. As a reference-based evaluation, SARI needs multiple references with various simplifications for each result sentence. There are two widely used expert-generated reference sets for TS evaluation, TurkCorpus,

[3] https://newsela.com/data/

and ASSET. Both share the WikiLarge test set as the original but provide different 1-to-many references per original sentence (see Table 2.2).

TurkCorpus TurkCorpus is first introduced together with SARI in Xu et al. (2016a). It has 8 simplification references for each original sentence, and each of the references is generated by different native speakers to ensure their variety. However, this dataset is considered mostly rewriting sentences by lexical paraphrasing (Alva-Manchego et al., 2020b).

ASSET To make the references more varied, Alva-Manchego et al. (2020b) releases a new evaluation set with 10 new-generated references for each original sentence. It is claimed that the new rewriting can better capture features of simplicity. It has also been widely used in recent studies.

TABLE 2.2: Evaluation Datasets Summary

Name	Instance Number	Num of References per Original
TurkCorpus	359	8
ASSET	359	10

2.1.3 Decoding Strategy

The raw output of a decoder at each time step is an array of probability for each token. Finding the most likely output sequence involves searching all the possible output tokens. An intuitive way called greedy search is selecting one best candidate as an output for each time step in practice. However, the best candidate for the current time step could be a sub-optimal choice for the full sentence. I use the commonly used beam search technique in our decoding process. The beam search algorithm selects a number of alternatives for an input sequence at each time step based on conditional probability. The only parameter is the beam width, B, which determines the number of alternatives. Common beam width values are 5 or 10, and beam search with beam

width 1 is the greedy search. Generally speaking, larger beam widths result in better performance, but it will decrease the decoding speed.

2.2 Text Simplification Methods

Text simplification has developed for many years from rule-based methods to data-driven methods. I begin by outlining the major approaches to text simplification.

2.2.1 Before Data-Driven Text Simplification Methods

Early studies of TS covered a lot of hand-crafted feature engineering. Chandrasekar et al. (1996) were the first to develop a two-stage simplification preprocessing step for a parser. Many rule-based works were proposed in the early years. Devlin (1998) proposed a framework that uses Kucera-Francis written frequency (Kucera et al., 1967) rank synonyms from the semantic thesaurus, such as WordNet (Miller, 1998), to do lexical simplification by identifying the most common synonym. Carroll et al. (1998) combine some rule-based natural language processing tools with building a text simplification system to assist aphasic readers. Another notable work learns to rewrite rules from annotated corpora as a pre-processing step to assist other natural language applications (Chandrasekar and Srinivas, 1997). A tool for helping writers remove ambiguity and complexity based on grammar rules was proposed by naming EasyEnglish (Bernth, 1997). Beigman Klebanov et al. (2004) eases the task of accessing factual information by addressing the text simplification problem. Furthermore, a text simplification framework applies transformation rules from Extensible Markup Language files to a dependency representation (Siddharthan, 2011).

2.2.2 Data-Driven Text Simplification Methods

2.2.2.1 Monolingual Machine Translation based Methods

With the boom of machine translation methods, a popular way to simplify text is to treat it as monolingual machine translation, with the original and simplified pairs as source and target sentences. In early work on this subject, Wubben et al. (2012) proposed a PBMT-R (Phrase-Based Machine Translation with dissimilarity-based Re-ranking) method for careful phrase-based paraphrasing, also discussing the application of text readability metrics such as Flesch-Kincaid grade level (Kincaid et al., 1975) for evaluating the text simplification system performance. Coster and Kauchak (2011) predefined complex word alignments for adopting the text simplification task. Narayan and Gardent (2014) proposed a hybrid method to derive simple sentences from complex ones by combining monolingual and deep semantic machine translation. Xu et al. (2016a) employed large-scale paraphrases from bilingual texts and small parts of manual simplifications to conduct an in-depth adaption of traditional statistical machine translation. In addition, the authors also proposed an automatic metric SARI for evaluating text simplification. Radford et al. (2018) is the first to use a popular phrase-based machine translation system Moses (Koehn et al., 2007) for text simplification tasks without adaptations. Nisioi et al. (2017) is the first simplification model based on an attended encoder-decoder machine translation model provided by OpenNMT system (Klein et al., 2017).

2.2.2.2 Pre-trained Model Fine-tuning Methods

Pre-training a language model on vast corpora will obtain good word representations for downstream tasks. Fine-tuning pre-trained models (PTMs) has achieved outstanding performance in TS, just like in other NLP tasks (Qiu et al., 2020). ACCESS (Martin et al., 2020a) fine-tunes the BART model (Lewis et al., 2019) with pre-defined prefixes. The main issue of training the Seq2Seq

text simplification model is the lack of high-quality parallel data. Some methods also introduce pre-generated pseudo-parallel sentences to augment the training data (Martin et al., 2020c; Lu et al., 2021), which achieved excellent unsupervised results. I propose an approach in Chapter 5 that requires no training data but only customized prompts for each input and obtains better results.

2.2.2.3 Methods without Parallel Datasets

The above-mentioned Seq2Seq methods achieve excellent performance in simplification tasks in specific parallel datasets such as Wikilarge and Newsela, but the training datasets they use are not only too small to overcome overfitting but also of low quality with lots of noise. Therefore, many researchers tend to build unsupervised methods. In terms of unsupervised methods, lexical simplification was performed by Narayan and Gardent (2015) and Paetzold and Specia (2016). They replaced complicated words with simpler synonyms, which is not considering Grammar and Syntax simplification. Štajner and Nisioi (2018) proposed using reduced vocabulary and copy mechanism to improve datasets to obtain a better result for both in-domain and cross-domain text simplification. Surya et al. (2018) utilized two parts of adversarial training to restrict similar attention distribution between simple and complex sentences. Zhao et al. (2020) adopt the back-translation framework for unsupervised text simplification. The authors also used a denoising auto-encoder for simplification, in which the reinforcement learning algorithms are used for promoting the back-translation.

2.2.2.4 Other Seq2Seq Deep Learning Methods

Although many recent studies treat text simplification as a monolingual translation and pre-trained model (PTM) fine-tuning has achieved good performance, other Seq2Seq deep learning models have also had an impact in the

TS field. The Neural Semantic Encoders by Vu et al. (2018) is a method that proposes an extension of this architecture by using augmented memory. Guo et al. (2018) introduced multi-task learning with related auxiliary tasks of entailment and paraphrase generation in this architecture. A transformer (Vaswani et al., 2017) based model developed by Zhao et al. (2018b) integrated external paraphrase knowledge; the authors claim it could utilize real-world simplification rules. To avoid whole sentence directly copying and to make the output more diverse when applying generic Seq2Seq simplification models, Kriz et al. (2019) first incorporated content word complexities and secondly generated a re-ranking system for generated candidate simplifications, which improved the automatic evaluation results.

Text simplification is often said to be very similar to text summarization. However, summarization aims to generate a shorter version of the source; simplification generates a more readable output and may have longer text. Both of these tasks often apply conditional training with Seq2Seq models (Kikuchi et al., 2016; Fan et al., 2017). In addition, a Seq2Seq model trained with a deep reinforcement learning framework named Dress was proposed by Zhang and Lapata (2017) outperforms competitive simplification systems.

Edit-based methods (Omelianchuk et al., 2021; Dong et al., 2019) try to modify the sentence with explicit edit operations. CROSS (Mallinson and Lapata, 2019) adds indicator features to word embeddings and templates to sentences for lexical and syntactic constraints. All the above Seq2Seq methods need a large amount of high-quality parallel data, which is an open problem for Seq2Seq training in the TS field. Other methods also introduce Generative Adversarial Network (GAN) to circumvent the problem (Surya et al., 2019).

2.3 Other Related Work

2.3.1 Denoising Method

Adding noise into the auto-encoder input layer has been shown to be effective in creating robust latent representations, as demonstrated by denoising auto-encoders. Vincent et al. (2008) first proposed a denoising auto-encoder to initialize deep architectures for robust image representations learning. Poole et al. (2014) proposed a single-layer denoising framework and showed that some types of noise improved the performance. A denoising adversarial auto-encoder was proposed by Creswell and Bharath (2018) where the authors used it for generative image modelling. Zhao et al. (2020) showed that noise in text, such as word dropout, shuffle and replacement, can be useful for text simplification. In contrast to the previous works that focused on continuous image data, Shen et al. (2019) demonstrated that input noises are particularly useful for discrete text modelling using powerful sequence networks because they encourage the preservation of data structures in latent space representations. In Chapter 3, our work takes advantage of the input noise attributes to build a text simplification system.

2.3.2 Latent Representation

Many unsupervised learning methods aim to learn data representations that enable manipulating variations of underlying latent factors. Sohn et al. (2015) proposed a conditional deep generative model with Gaussian latent variables. Given an input sentence, another method combines deep generative models with sequence-to-sequence models to generate paraphrases (Gupta et al., 2018). Previous studies also adopt an auto-encoder architecture with style discriminators in text style transfer work to learn disentangled representations (Shen et al., 2017; Yang et al., 2018). In our study, I propose an unsupervised method with adversarial learning in Chapter 3.

2.3.3 Pre-trained Models

Over the last few years, more and more models pre-trained on language modelling tasks, starting from ELMo (Le Quéré et al., 2018), OpenAI GPT (Radford et al., 2018), and BERT (Devlin et al., 2018), have achieved impressive results on various natural language processing tasks. Most of the pre-trained models are based on the "Transformer" architecture (Vaswani et al., 2017), which relies on attention mechanisms, and does not have an explicit conception of word order other than labeling each word with a positional embedding. Seq2Seq PTMs have also been proposed with reconstruction objectives (Lewis et al., 2019; Raffel et al., 2019). Their architecture naturally fits text-to-text tasks. The size of the PTMs have become increasingly larger, making the fine-tuning costs very large (Wang et al., 2022a). Thus, a lot of huge PTMs have been applied with the zero-shot and few-shot learning strategies recently (Brown et al., 2020b; Wei et al., 2021). The methods I build on in Chapters 4 and 5 employ PTMs.

2.3.4 Training-Free Techniques

2.3.4.1 Prompt Engineering

Prompting is a new way of using PTMs with or without fine-tuning (Liu et al., 2021a). Prompts can be roughly categorised into two types: cloze and prefix. The cloze prompt creates slots to remind the language model to fill in a particular location. Pre-defined cloze prompts are commonly used in few-shot learning settings on text classification and text generation tasks (Schick and Schütze, 2021a,b). The prefix prompt is used as the additional input before the original. Li and Liang (2021) and Lester et al. (2021) propose prefix-tuning methods, which keep language model parameters frozen, but optimize the prefix vector for different tasks. They only need to learn a very small portion of model parameters to achieve comparable performance to traditional fine-tuning methods learned on all the parameters. Gao et al. (2021) use T5 (Raffel

et al., 2020) automatically generating task-specific prompts to help fine-tune, which outperforms vanilla fine-tuning. I propose a new prompting method in Chapter 5 for the TS task.

2.3.4.2 Tuning-Free Prediction

Tuning-free in-context learning is becoming popular due to the prevalence and success of GPT-3, making the model predict an answer only given the task description. LAMA (Petroni et al., 2019a) analyses a wide range of pre-trained language models (PLMs) to conclude that the original BERT contains relational knowledge that matches database-based NLP methods. A suite of human-designed templates for testing language model understanding via language modelling is introduced in Ettinger (2020). Jiang et al. (2020) systematically uses an automatic method to generate prompts used in retrieving factual knowledge from language models, resulting in better performance than manually designed prompts. Our methods in Chapter 5 are also tuning-free methods.

2.3.4.3 Constrained Answer Spaces

In most cases in the literature, the answer space is the entire set of tokens. In prompting methods, tasks with limited label space, such as text classification (Yin et al., 2019), or entity recognition (Cui et al., 2021a), often constrain the possible outputs. Gao et al. (2021) prune the search space by selecting the top k vocabulary words based on the conditional likelihood of the initial language model. In Chapter 4 and 5, our models construct pruned search spaces by removing complex vocabulary words for text generation.

Chapter 3

Text Simplification with Adversarial Neural Networks

In recent years, autoencoder-based generative models have achieved good performance in natural language processing (Bowman et al., 2015; Hu et al., 2017; Shen et al., 2017; Zhao et al., 2018a). They project sentences as latent representation vectors and then transform text using simple calculations on the latent vector. The critical component is finding the geometry of latent representations and then capturing underlying sentence syntax and semantics. Many text simplification studies are also based on this architecture (Surya et al., 2018; Martin et al., 2020b; Zhao et al., 2020). However, none of them consider preserving content while simplifying. Some results generated by these previous works introduce incorrect content or result in lost key information making the results useless or harmful in practice, particularly in professional fields (Zhang et al., 2018). In this chapter, I consider the key idea of content preservation in the text simplification work. Furthermore, it has been discovered that additional guidance, such as denoising, leads to improved detection of sentence representation alignments (Shen et al., 2019). An asymmetric denoising method is added to the model.

3.1 Latent Space for Text Simplification

Learning representations of data is a key objective for fine control over the underlying latent factors of variation. In some cases, these latent factors are given or can be learned through observation of samples from the data distribution (Shen et al., 2017). These latent factors can often be learned as a transformation from source to target. However, in some problems, access to parallel datasets is limited. Studies have focused on learning mappings between two data domains in image style transfer and machine translation. Although they achieved great success in the visual domain (Isola et al., 2017), further research still needs to be done for natural language generation-related problems. Because in all of these related problems, the source sentence's content must be preserved but generated with desired presentation constraints in the target. Many studies formulate the problem into an encoder-decoder-based framework in the text style transfer domain that the encoder maps the text into a style-independent latent representation, and the decoder generates text based on this latent representation plus a different style variable (Fu et al., 2017). In other words, many researches have focused on disentangling style and content from a sentence to control its representation (Shen et al., 2017; Hu et al., 2017).

In the work, I treat text simplification as a special case of text style transfer, i.e., transferring the complex style of text to a simple style text. Thus, the first step is also finding the latent representation. Sentence latent representation preserves the underlying structure in latent space, including semantics and syntax. In other words, similar sentences tend to locate close together in latent space (Sutskever et al., 2014). This idea is schematically illustrated in Fig 3.1 for a 2D (two-dimensional) latent space.

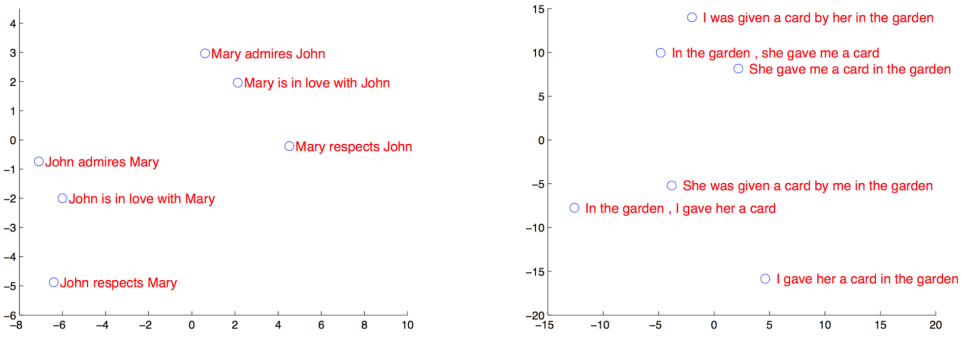


FIGURE 3.1: Sentence latent representations in 2D from Sutskever et al. (2014)

Intuitively, text simplification, which aims to map complex sentences to a simpler domain, could be viewed as finding complexity-independent content and complexity representations. However, these two aspects interact in subtle ways in natural language sentences. A good context latent representation tends to maximally preserve the meaning of the text while removing as much linguistic complexity as possible. The goal is to explore a transformation to map complex sentences' latent representation to simple sentences. A complexity-independent content vector is obtained and then decoded to a simple representation. If we can achieve the general transformation, we can achieve a general simplification method by mapping a sentence to the latent space and then using the transformation to get a simple latent representation for decoding a simple counterpart.

3.2 Adversarial Unsupervised Model with Text Denoising

It requires large-scale high-quality parallel corpus to train a TS systems in recent Seq2Seq ways. However, the most commonly used dataset, WikiLarge, is not large enough and contains a lot of noise. Although we lack paired data at the sentence level, it is easy to access abundant unlabeled data in

domains: a set of sentences in a complex domain and a set of sentences in a simple domain. In this Chapter, an unsupervised text simplification system is designed to overcome the shortage of sentence parallel datasets. I explore a method to learn the transformation from the complex domain into simple domains without sentence-level complex-simple pairs. It is based on the idea that semantic meaning can have different sentence renderings and that latent relationships exist between them (Liu et al., 2018).

Following Shen et al. (2017), I use the adversarial autoencoder (Makhzani et al., 2015) to learn the attention weighted representation for the simple sentence generation. We learn an encoder that maps a complex sentence to a latent representation and then reconstructs it by the simple sentence decoder. The study tries to find the transformations between latent representations of pairs of sentences.

The proposed system consists of three parts: 1. the encoder-decoder part, 2. the adversarial part, 3. the sentence similarity measurement part. The core of the architecture is based on two encoder-decoder models that share an identical encoder for both the complex and simple involved, similarly to Ha et al. (2016) and Artetxe et al. (2017). The shared encoder aims to generate latent representations of the input text with different complexity. In other words, $\mathbf{z}_c$ for complex sentence representation and $\mathbf{z}_s$ for simple sentence representation into a same distribution; then, each complexity-dependent decoder renders them to the corresponding complexity to conduct reconstruction.

The architecture overview is illustrated in Fig. 3.2. The encoder $\mathbf{E}$ will generate sentence representations for all types of sentences, and the simple sentence decoder $\mathbf{G}_s$ and complex sentence decoder $\mathbf{G}_c$ are responsible for reconstructing sentences with corresponding complexity. The objective is to feed the latent representation of complex sentence $\mathbf{z}_c$ to the simple sentence decoder $\mathbf{G}_s$ to generate simple sentences. If the complex sentences can be

reconstructed from $\mathbf{z}_c$, it contains the intact semantic meaning of the original complex sentences. $\mathbf{G}_s$ will add weights to $\mathbf{z}_c$ by choosing which hidden states to attend to generate a simple sentence. Intuitively, an encoder-decoder system's decoder only works well when its input comes from a distribution very close to the one induced by its encoder. In the system, that means I should make the distribution of latent representation $G_s(\mathbf{z}_c)$ close to the distribution of $G_s(\mathbf{z}_s)$ for the G_s to generate simple style sentences.

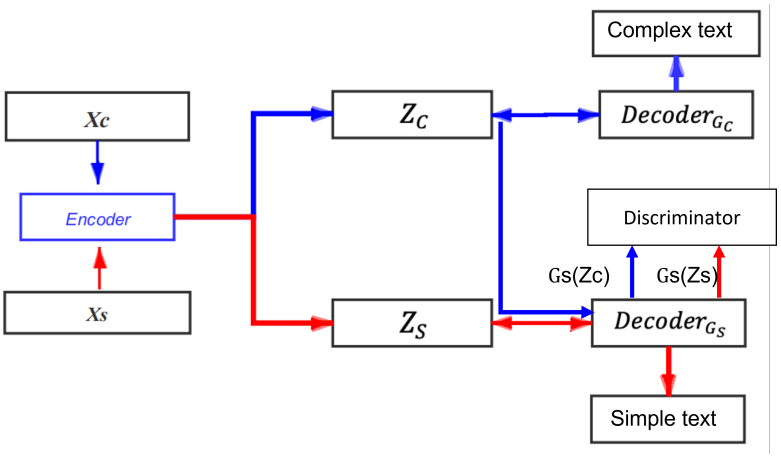


FIGURE 3.2: The Architecture of The Model

This is achieved by introducing an adversarial loss function. I train a discriminator **D** to classify between the latent representations of complex sentences and simple sentences. The discriminator part is just analogous to GANs (Generative Adversarial Networks).

One common issue in the simplification task is that it learns to copy the entire sentence with minimal or even without change during training. Especially in the pure autoencoder architecture, it will easily learn to only copy every input word one by one without any useful structures in the data. A denoising autoencoder can force the decoder to leverage external attribute information (Lample et al., 2017). Because the noise corrupts the input, the decoder has to use external features to reconstruct the sentence instead of a direct copy. Shen et al. (2019) proved that denoising the adversarial autoencoder model

would map similar sequences to similar latent representations. It will help the decoder $\mathbf{G}_s$ find the simple sentence distribution from the latent space.

A novel asymmetric denoising technique was employed to model simple and complex sentences separately, which helps the simplification system to learn distinguishable latent representations from sentences with different complexity. The denoising process will help the system learn lexical simplification potentially. Moreover, I propose a content similarity constraint that helps the system preserve the complex sentences' content when the decoder generates simplified ones.

3.3 Model Description

The model is based on an encode-decode architecture with an adversarial part. Eq. (3.1) is the basic loss function. The details of each part are described in the following subsections.

$$\min_{E,G} \max_{D} \mathcal{L}_{\text{rec}} \left(\theta_E, \theta_G\right) - \lambda \mathcal{L}_{\text{adv}} \left(\theta_G, \theta_D\right) \tag{3.1}$$

3.3.1 Autoencoder

In the model, encoder **E** is responsible for extracting features from x to generate latent representations $\mathbf{z}$ that can be used for the new sentences sequentially generating one word each time.

The loss in this part could be recognised as reconstruction loss from both decoder parts. I define $\mathbf{X}_s$ and $\mathbf{X}_c$ as datasets of simple and complex sentences, respectively. I first use the encoder E to encode x_c and x_s to get the latent representation $E(x_c) = \mathbf{z}_c$ and $E(x_s) = \mathbf{z}_s$. I denote θ_E, θ_{G_s}, θ_{G_c} as parameters of the encoder, simple sentence decoder and the complex sentence decoder, respectively. The reconstruction loss consists of both $\mathbf{E} - \mathbf{G}_c$ and $\mathbf{E} - \mathbf{G}_s$ parts. Then it can be defined as follows:

$$\mathrm{L}_{rec}\left(\theta_E, \theta_{G_s}, \theta_{G_c}\right) = \mathbb{E}_{X_s \sim \mathcal{S}}\left[-\log P_{E-G_s}\left(X_s|\mathbf{z}_s\right)\right] + \mathbb{E}_{X_c \sim \mathcal{C}}\left[-\log P_{E-G_c}\left(X_c|\mathbf{z}_c\right)\right] \quad (3.2)$$

3.3.2 Adversarial Training

To better generate sentences with the desired complexity, I introduce additional supervision with a discriminator **D** that maximizes the complexity classifier's accuracy. We could not backpropagate the gradients if adversarial training were performed on the discrete samples generated by $\mathbf{G}_s$. Although reinforcement learning can be adapted to address this issue, training with these methods can be unstable (Zhang and Lapata, 2017). In the method, I use the attention weighted latent representation, which contains the output's information and is smoothly distributed, as the input of **D**. Attention weights are generated from decoders. So the attention weighted complex sentences representation for $\mathbf{G}_s$ is denoted as $\mathbf{G}_s(\mathbf{z}_c)$ and simple sentences representation is $\mathbf{G}_s(\mathbf{z}_s)$. That is, the input to the discriminator **D** is either $\mathbf{G}_s(\mathbf{z}_s)$ or $\mathbf{G}_s(\mathbf{z}_c)$.

D generates a binary output, and if $\mathbf{G}_s(\mathbf{z})$ is close to a simple sentence representation in the dataset, the output is 1; otherwise, 0. It is used to lead the latent representation $\mathbf{z}_c$ to reconstruct complex sentences well, in which $\mathbf{z}_c$ can generate simple sentences by $\mathbf{G}_s$. Eventually, $\mathbf{E} - \mathbf{G}_s$ learns to produce simple-like latent representations from complex input sentences. The discriminator loss is shown as follows:

$$\begin{aligned} \mathcal{L}_{adv,D}(\theta_D) &= -\mathbb{E}_{X_s \sim \mathcal{S}}[\log(D(G_s(\mathbf{z}_s)))] - \mathbb{E}_{X_c \sim \mathcal{C}}[\log(1 - D(G_s(\mathbf{z}_c)))]. \\ \mathcal{L}_{adv,E}(\theta_E, \theta_{G_s}) &= -\mathbb{E}_{X_c \sim \mathcal{C}}[\log(D(G_s(\mathbf{z}_s)))]. \end{aligned} \quad (3.3)$$

3.3.3 Content Preservation Constraint

The framework is composed of a shared encoder and a pair of decoders, as shown in Fig.3.2, $\mathbf{G}_s$ is crucially assisted by discrimination-based losses. Without any constraint, $\mathbf{G}_s$ easily simplifies sentences too much to lose important original information. To preserve the meaning, I build a content similarity constraint loss between sentence representations. Motivated by the text similarity measurement in Siamese networks (Neculoiu et al., 2016), I measure the similarity between complex sentences attention weighted representation for reconstruction $G_c(\mathbf{z}_c)$ and representation for simplification $G_s(\mathbf{z}_c)$ by using their cosine similarity. It is shown in Eq. (3.4), where $\mathbf{x}_1$ and $\mathbf{x}_2$ are vectors. The content preservation loss is given in Eq. (3.5). To measure representations with different sequence lengths, H_c^c is defined to the average of $G_c(\mathbf{z}_c)$, while H_c^s is defined to the average of $G_s(\mathbf{z}_c)$. By adjusting the importance of this metric in the loss function, we can control the degree to which we maintain the sentence content.

$$Cosine(\mathbf{x}_1, \mathbf{x}_2) = \frac{\langle \mathbf{x}_1, \mathbf{x}_2 \rangle}{\|\mathbf{x}_1\| * \|\mathbf{x}_2\|} \tag{3.4}$$

$$\mathcal{L}_{sim,G_s}(\theta_E, \theta_{G_s}) = Cosine(H_c^c, H_c^s). \tag{3.5}$$

3.3.4 Asymmetric Denoising

Word dropout and shuffle are commonly used denoising strategies that have been shown to critically impact unsupervised machine translation systems (Lample et al., 2017). However, in text simplification systems, symmetric noise will not be very effective because the features I want to use in the simple sentence are different from those in the complex sentence. So I will propose asymmetric noise for simple and complex corpus.

I propose three kinds of noise in the **simple** part:

Simple-Substitution: Simple sentences are supposed to with low complexity words and structures. In other words, complex words could be recognised as

noise. I use the Simple PPDB (Paraphrase Database) (Pavlick and Callison-Burch, 2016) to replace simple words with complex ones. Simple PPDB contains 4.5 million pairs of simplified and complex expressions. Every pair follows a simplification rule with a score of confidence. Some examples are given in Table 3.1.

TABLE 3.1: Examples of the Simple PPDB

Score	Rules
0.57	tiring → tired
0.80	weary → tired
0.84	fatigued → tired
0.96	completely exhaust → tired

To some extent, substitution simulates the lexical simplification process. It could help the model learn words from simplified sentences. Moreover, simple words could be encouraged to be generated from the shared latent space for the decoder.

Additive: Additive noise adds additional words to the simple input. Additive noise for compression of the sentences in Fevry and Phang (2018) causes incomplete but true summaries of sentences. The model will remove words from the corrupt inputs and produce logical sentences. I randomly select a bigrams subsequence from a sentence and then insert the subsequence into the complex input.

Simple-Shuffle: Word shuffling is a widely used technique in the denoising method. It has been proven that word shuffling helps the model to learn useful structure in sentences (Lample et al., 2017). To make the additive words evenly distributed in the noised simple sentence, I concatenate the complex sentence and the additive subsequence and completely shuffle the bigrams, keeping all word pairs together.

I also propose three kinds of noise for **complex** sentences.

Complex-Substitution is also performed for complex sentences. Here, I normally use the rules in Simple PPDB to replace the complex words with simpler versions. Besides, I applied another two noising methods.

Drop: Word dropping discards several words from the sentences. During the reconstruction, the decoder has to recover the removed words through the context. Mapping from simple to complex usually includes sentence extensions, which need the decoder to generate extra words. I only delete the "frequent word" with the probability of 0.6 because words with a lower frequency usually contain more semantic information. I define "frequent word" as words that appear 100 times in thecorpus. A similar approach has also been used in unsupervised language generation (Freitag and Roy, 2018).

Complex-Shuffle: Similar to (Lample et al., 2017), I only slightly shuffle the complex sentences instead of the complete shuffle process for simple sentences. Because the sentences have longer and more complex words, it is hard for the decoder to reconstruct the sentences with the complete shuffled inputs.

Motivated by the BERT (Devlin et al., 2018) masking strategy, I only make the noise mentioned above 80% of the time, randomly replace a token 10% of the time, and keep the token unchanged 10% of the time. I introduce a denoising term in theloss function, as shown below

$$\mathcal{L}_{\text{denoi}} = -\mathbb{E}_{X_s \sim \mathcal{S}}\left[\log P_{E-G_s}\left(X_s \mid \text{noise}_s\left(X_s\right)\right)\right] - \mathbb{E}_{X_c \sim \mathcal{C}}\left[\log P_{E-G_c}\left(X_c \mid \text{noise}_c\left(X_c\right)\right)\right]. \tag{3.6}$$

3.3.5 Employ Pre-trained Embeddings

Many works have shown that using pre-trained embeddings is beneficial for NLP tasks, and there is no exception in TS task. Thus, instead of random initial embeddings (not good at capturing synonymy relations (Tissier

et al., 2017)) which are crucial for the simplification task, I use the pre-trained *GloVe* embeddings as the initial embeddings (Pennington et al., 2014). I obtain *GloVe* embeddings from the official website [1]. In the framework, the encoder and the decoders share the same initial word embeddings. In order to maintain consistency on different sides, I freeze embeddings on each side in the training progress.

3.4 Training Details

Algorithm 1 gives the overall training process. It produces two basic results: 1. $\mathbf{E} - \mathbf{G_c}$ works as an autoencoder for sentence reconstruction, 2. $\mathbf{E} - \mathbf{G_s}$ works to simplify its input.

In the initialization phase, I train the encoder $\mathbf{E}$ and two decoders $\mathbf{G_c}$ and $\mathbf{G_s}$ with denoising loss $\mathcal{L}_{\text{denoi}}$ and reconstruction loss $\mathcal{L}_{\text{rec}}$. Then I train the discriminator with $\mathcal{L}_{adv,D}$. At this stage $\mathcal{L}_{adv,D}$ is not used to update the encoder. Initialization allows two decoders and the discriminator to learn independently of each other. In the adversarial phase, $\mathcal{L}_{adv,G_s}$ and $\mathcal{L}_{rec}$ also participate in training encoder $\mathbf{E}$ and the two decoders. At this stage, $\mathcal{L}_{sim}$ is introduced for preserving the content of the sentence.

In the training process, asymmetric denoising is introduced for diversification. It encourages input for the decoder $\mathbf{G_s}$ to be different from input for the decoder $\mathbf{G_c}$. It helps distinguish between latent representations $\mathbf{z}_c$ and $\mathbf{z}_s$.

3.5 Experiments

I describe the experimental settings in this section and then analyze the results.

[1] https://nlp.stanford.edu/projects/glove/

Algorithm 1 Unsupervised simplification algorithm using denoising, reconstruction, adversarial and similarity losses.

Input: unlabeled simple dataset X_s, unlabeled complex dataset X_c.
Initialization phase:
repeat:
  Update $\theta_E, \theta_{G_s}, \theta_{G_c}$ using $\mathcal{L}_{\text{denoi}}$
  Update θ_D using $\mathcal{L}_{adv,D}$
until specified number of steps are completed
Training phase:
repeat:
  Update $\theta_E, \theta_{G_s}, \theta_{G_c}$ using $\mathcal{L}_{\text{denoi}}$
  Update $\theta_E, \theta_{G_s}, \theta_{G_c}$ using $\mathcal{L}_{adv,G_s}$
  Update θ_D using $\mathcal{L}_{adv,D}$
  Update θ_E, θ_{G_s} using $\mathcal{L}_{sim}$
until specified number of steps are completed

3.5.1 Datasets Description

For training the model, I use the unlabeled dataset *Wiki*720*k* of simple and complex sentences provided by Surya et al. (2018). It partitioned the standard en-wikipedia dump into simple and complex groups based on FE score (Flesch, 1948). Sentences with FE scores greater than 70 are considered simple, and sentences with FE scores under 10 are categorized as complex. The dataset statistics are shown in Table 3.2. The validation set (2000 sentences) and test set (359 sentences) are TurkCorpus (Xu et al., 2016a), which have 8 reference sentences for each source sentence.

TABLE 3.2: Number of sentences in statistics with average words per sentence, average FE score and FE score range select for building the *Wiki*720*k* training set.

Category	Sents	Avg. Words	Avg. FE	FE-range
Simple	720*k*	18.23	76.67	$74.9 - 79.16$
Complex	720*k*	35.03	7.26	$5.66 - 9.93$

I use the Newsela test set as the second test set. I follow Zhang and Lapata (2017) to generate the new test set alignment for the ablation study.

3.5.2 Hyperparameter Settings

Both the encoder and decoders are built with bi-directional GRU (Gated recurrent unit) (Cho et al., 2014) architectures with two layers each and the global attention (Luong et al., 2015) is used in our models. The discriminator is a CNN-based classifier analogous to (Kim, 2014) with filters size from 1 to 5. Based on the computing device capacity and practical experience, hyper-parameters are empirically selected as follows: The model chooses a hidden size of 600 and an embedding size of 300. The batch size is 36 and the beam search size is 10. Learning rates are 0.0001 for updating $\theta_E, \theta_{G_s}, \theta_{G_o}$, and 0.00005 for updating θ_D and $\mathcal{L}_{\text{sim}}$. It takes 6000 steps in batches for the initialization.The experiments were executed on an 11 GB GPU.

3.5.3 Comparison Methods

I consider three unsupervised sentence simplification methods and one unsupervised lexical simplification method as the main baselines. For further comparison, I also introduce two supervised and one semi-supervised method.

UNMT is a monolingual unsupervised translation method proposed by Artetxe et al. (2017). It uses back-translation and denoising techniques.

UNTS (Surya et al., 2018) is an unsupervised neural text simplification method by using adversarial training.

LIGHTLS (Glavaš and Štajner, 2015) makes use of the most recent word vector representations for lexical simplification.

SBMT (Xu et al., 2016a) is a supervised method that is a statistical machine translation with optimizing for text simplification.

ACCESS (Martin et al., 2020b) is a Transformer (Vaswani et al., 2017) based Sequence-to-Sequence model that adapts a discrete parametrization mechanism to an explicit control simplification system.

XLM-un and **XLM-semi** are two methods that are adaptions of XLM framework in unsupervised and semi-supervised ways, respectively (Conneau and Lample, 2019). XLM-semi takes advantage of 5000 parallel sentences as part of the training dataset.

In addition, I also report results for three other baselines: (1) The **Back-T(e-c-e)** is that it adopts the back-translation technique first to translate the test set to Chinese and then translate it back to English by using Google Translation[2]. (2) The **Truncation** baseline is used to truncate the source sentences by keeping the first 80% of the words as the simplification results. This baseline is a good comparison by standard text simplification metrics (Martin et al., 2020d). (3) **Reference** is the reference of the TurkCorpus (Xu et al., 2016a) test set; I choose the first human-generated reference.

3.5.4 Results and Analysis

Table 3.3 shows the test results of the model using several evaluation scores along with the previous state-of-the-art supervised and unsupervised studies' results.

Supervised method ACCESS achieves the best results as shown through the SARI score of 41.38 on TurkCorpus. The method achieves 37.10 in SARI and 78.09 in BLEU. I obtain a better SARI score on TurkCorpus than all the unsupervised comparison methods. As a machine translation adaption method, I observe that XLM-semi often directly copies the source sentences to the output, obtaining the highest exact match score of 0.76. Even the XLM-un has the second-highest exact score at 0.30. This is why XLM achieves a higher BLEU score but a lower SARI score, as shown in Table 3.3. Our method obtains the best score of 8.01 in lexical complexity measurement. Our method has an advantage in lexical simplification because it introduced substitution noise, making the denoising operation inherently apply lexical simplification.

[2]https://translate.google.co.uk/

LIGHTS substitutes complex words using some rules. This often results in high BLEU and factCC scores, because the lexical simplification method only substitutes complex words without any other changes, which leads to limited simplification. Compared to the three generative unsupervised methods, XLM-un, UNTS, and UNMT, the method achieves a higher BLEU score and factual consistency accuracy, indicating that my method can better preserve the source's content. The back-translation method Back-T(e-c-e) obtains the highest factual consistency score, motivating us to incorporate back-translation to conduct content preservation in the future work. I also present some example results in Table 3.4.

TABLE 3.3: Comparison of automatic evaluation metrics in different methods. The su, unsu and semi stand for supervised, unsupervised, and semi-supervised, respectively.

Name	Categories	SARI	BLEU	FKGL	EM	LC	factCC
SBMT	su	38.59	73.62	7.95	0.10	8.03	88.58%
ACCESS	su	**41.38**	76.36	7.29	0.04	7.94	88.30%
XLM-semi	semi	28.30	**94.83**	9.75	0.76	8.19	-
XLM-un	un	35.63	76.93	7.74	0.30	8.15	-
UNTS	un	35.29	76.44	7.60	0.21	8.02	85.79%
UNMT	un	33.72	70.84	8.97	0.14	8.13	85.52%
Our_method	un	**37.10**	**78.09**	8.02	0.17	**8.01**	**86.07**%
LIGHTLS	-	34.96	80.40	9.63	0.20	8.03	93.31%
Identity baseline	-	33.80	83.54	10.02	1.00	8.34	-
Back-T(e-c-e)	-	39.02	54.64	9.19	**0.03**	8.23	**96.94**%

I also illustrate the average word length in each sentence, as measured by the number of characters, and the average sentence length, as measured by the number of words, of the results generated using different methods, as shown in Figure 3.3 and 3.4, respectively. Compared to the source sentences, the reference sentences, on average, have shorter word lengths and sentence lengths. Our method and other comparison methods all reduce both lengths to varying degrees, though the method tends to reduce both measures slightly more.

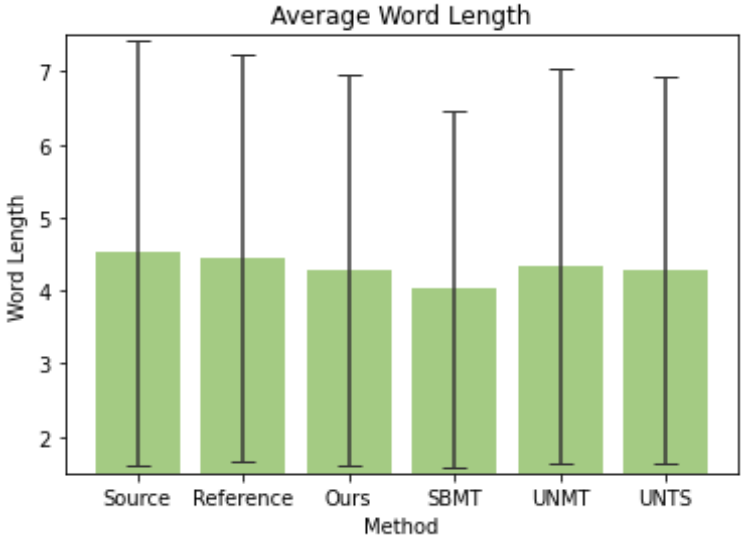


FIGURE 3.3: The word length averaged over all of the generated sentences by number of characters for the different approaches

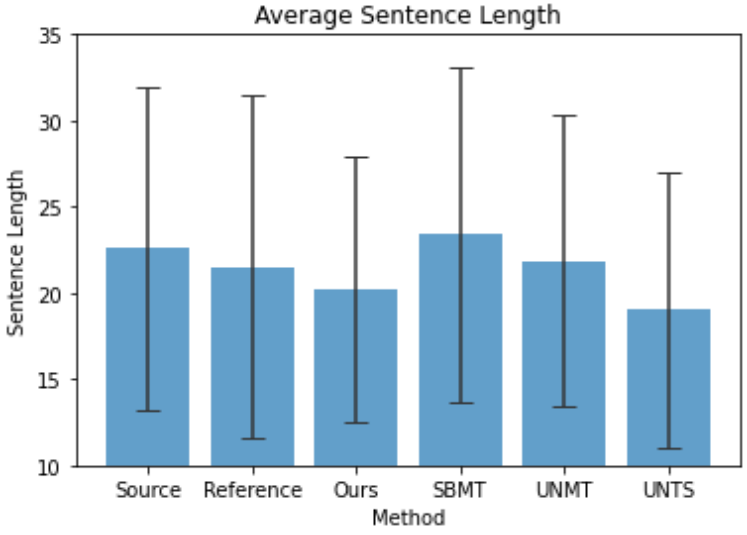


FIGURE 3.4: The sentence length averaged over all of the generated sentences for the different approaches. Our method produces relatively short sentences on average in comparison to other approaches.

A more detailed illustration of the compression ratio is shown in Figure 3.5, where I plot the histogram over the generated cases. As it shows, there are not only compressed sentences but also lengthened sentences in reference. That is the difference between text simplification and sentence compression or text summarization, reducing text length more often. However, most of the results focus on sentences with a constant or shorter length. Our training set is divided by Flesch Readability Ease (FE) (Flesch, 1948). Shorter sentences are considered to be simpler by FE. Therefore, it is believed that my model tends to generate shorter sentences due to being trained on this training set.

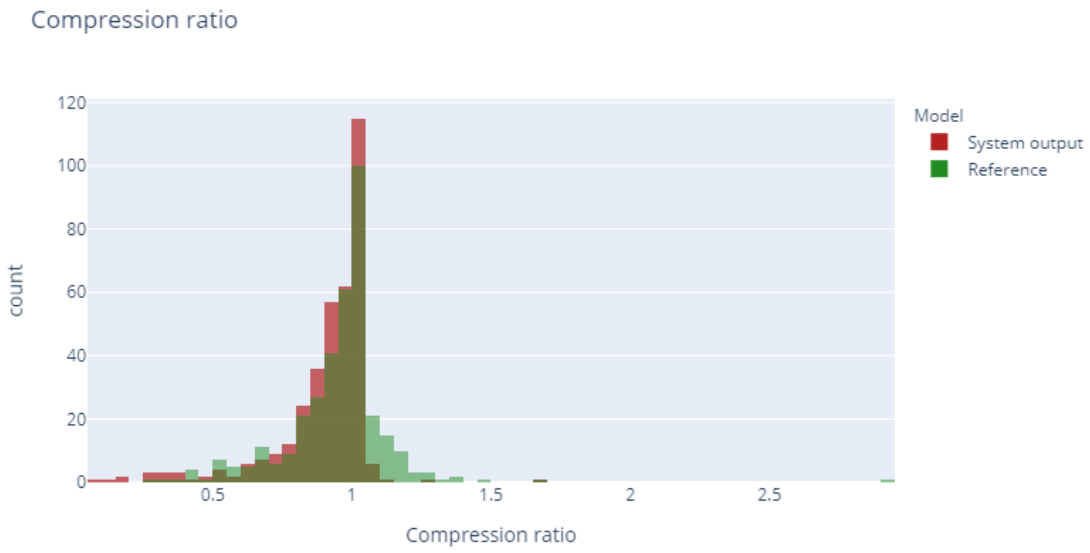


FIGURE 3.5: Distribution of the compression ratios between the original sentence and the output sentence. System output (red) is the model output, and Reference (green) is the human-generated Reference as described in Section 3.5.3. The brown part is the overlap of these two distributions. The count is the number of sentences.

3.6 Ablation Studies

In this section, I explore the contribution of the proposed architecture's different components using the SARI metric. I first explore the effects of various noise methods on the model and demonstrate that the content preservation

Source	Alessandro (" Sandro ") Mazzola (born 8 November 1942) is an Italian former football player.
Reference	Alessandro Mazzola is an Italian former football player.
SBMT	Alessandro ("Sandro") Mazzola (born 8 November 1942) is an Italian former football player.
XLM-semi	Alessandro (" Sandro ") **Mazzola (born 8 November 1942) is an Italian former** football player .
UNTS	Alessandro **" (")** Mazzola (born 8 November 1942) is **an** former football Player.
UNMT	Alessandro **" (Sandro ")** (born 8 November) is **an former Italian** football Player.
Ours	Alessandro **" (Sandro ")** is **a** former player .
Source	They are culturally akin to the coastal peoples of Papua New Guinea.
Reference	They are **culturally similar** to the coastal **people** of Papua New Guinea.
SBMT	They are culturally **close** to the coastal people of Papua New Guinea.
UNTS	They are **been akin** to the **Coastal** peoples of Papua New Guinea.
UNMT	They are **likely akin** to the **Coastal** peoples of Papua New Guinea.
Ours	They are **likely** to the coastal peoples of Papua New Guinea.
Source	This was absorbed into battalions being formed for XI International Brigade.
Reference	This was **added to battalions** being formed for XI International Brigade .
SBMT	This was **taken up under camps being set up for XI the Brigade.**
UNTS	This was **absorbed into brigades** being formed for **Xi Brigade.**
UNMT	This was absorbed into battalions being formed for **Xi** International Brigade.
Ours	This was **merged into regiment** being formed for XI International Brigade.

TABLE 3.4: Example results on Turkcorpus test dataset.

Source	I came to **recognise various** signs of a bad paper.
XLM-semi	I came to **recognise various** signs of a bad paper.
Ours	I came to **find many** signs of a bad paper.
Source	There has been a kind of inflationary process at work: nowadays anyone applying for a research post has to have published **twice the number of papers that would have been required for the same post** only 10 years ago.
XLM-semi	There has been a kind of inflationary process at work: nowadays anyone applying for a research post has to have published **twice the number of papers that would have been required for the same post** only 10 years ago.
Ours	There has been a kind of inflationary process at work: where anyone for a research post has to have published **twice the number of** only 10 years ago.

TABLE 3.5: Example Results on Randomly Selected No-target English Sentences.

is working; then, I further explore the effect of different components in the model.

I report these model scores trained with different part combinations on the TurkCorpus test set. The efficiencies of content preservation loss and two types of noise are tested in three situations: 1. **Sym.** : symmetric noise only. 2. **Asy.** : asymmetric noise only. 3. **Asy.** + **Pres.** : asymmetric noise with content preservation

Figure 3.6 shows the variation of SARI on the test set over steps using unsupervised training. The model with only symmetric noise (Sym.) has low scores but the fastest convergence rate during the training process. The proposed model with the asymmetric noise and content preservation (Asy.+Pres.) has the slowest convergence rate. However, the highest SARI score indicates that the content preservation improves the model's performance.

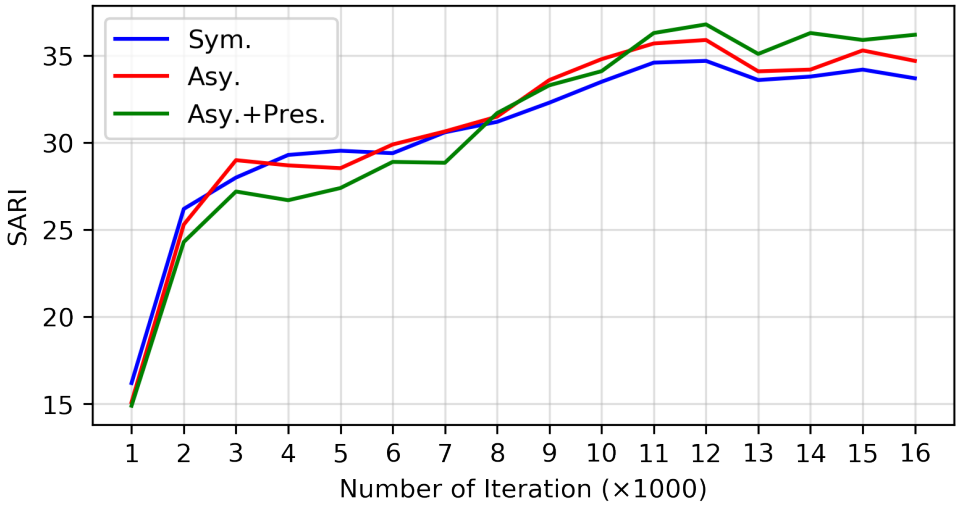


FIGURE 3.6: Variation in the SARI score in TurkCorpus over different types of noise in the architecture, where Sym. is symmetric noise only, Asy. is asymmetric noise only, and Asy. + Pres. is asymmetric noise with content preservation. As the number of iterations increases, the Asy. + Pres. performs the best, indicating the intent preservation is improving the model performance

The model results with different noise components are reported in Table 3.6. The noise types follow the definitions in Section 3.3.4. As we can see, the

asymmetric noise can improve the result.

Noise Type	Newsela	TurkCorpus
Additive & Shuffle	32.86	35.45
+Drop	33.54	36.21
+Substitution	33.92	36.80

TABLE 3.6: SARI scores with the various noise strategies on the two test datasets. Additive & Shuffle stands for Additive, Simple-Shuffle, and Complex-Shuffle noises as defined in Section 3.3.4; Drop stands for Drop noise; Substitution stands for Simple-Substitution and Complex-Substitution noises.

Data Name	SARI	BLEU	FKGL	EM	LC	Factcc
Autoencoder	31.01	92.74	9.54	0.61	8.30	93.42%
Auto + GAN	34.73	83.15	9.12	0.18	8.18	86.35%
Auto + denoising	33.80	85.41	9.32	0.25	8.22	89.69%
Auto + denoising + sim	32.57	90.32	9.14	0.49	8.23	91.56%
Our_method	**37.10**	78.09	**8.02**	0.17	**8.01**	86.07%

TABLE 3.7: Comparison of automatic evaluation metrics in the method with different components. Auto stands for autoencoder, denoising stands for the proposed asymmetric denoising, sim stands for content similarity constraint.

$\mathbf{E} - \mathbf{G}_s$ also illustrate the efficacy of different components in the model as shown in Table 3.7. Pure autoencoder can well reconstruct the sentences with exact match score 0.61 and BLEU score 92.74. *Auto + denoising* has lower BLEU and Exact match score compared to pure autoencoder. This means the proposed noises affect the reconstruction to some extent, but they can help get a higher SARI score.

We can see that adding the content similarity constraint helps the system preserve the meaning from the original sentences, by getting higher BLEU from 85.41 to 90.32 and factual consistency accuracy from 89.69% to 91.56% in comparison to the model without this loss.

Note that, during the training, by using the *Auto + GAN* strategy, the mode collapse problem appears, which is a common problem with GAN training. The mode kept generating the same sentence regardless of the input, and

optimization failed to progress. Because in the normal text GAN training without any constraint, finding a single point is sufficient to fool the discriminator.

3.7 Failure Case Analysis

Compared to text simplification performed by a human, two kinds of issues commonly arise in automatic text simplification systems: the misinterpretation and even introduction of incorrect information, particularly with professional phrases, and the handling of previously unseen words. These failure cases also occur in the results, as described in more detail below.

In the professional field, for example, in Table 3.8, the method substitutes 'inoperable abdominal cancer' with 'lung cancer'. These are two different cancers, though lung cancer is more common. This mistake is critical as it introduces incorrect information to the output. Our system needs improvement to tackle the issue of introducing false information, particularly in the context of specialized domain simplification, such as the medical domain, which can be an application of my work. Another example below shows a $\langle unk \rangle$ token for an out-of-vocabulary word. It is widespread in most text generation tasks to handle unseen or rare words. Much research has shown that using the subword method can effectively deal with this issue (Kudo, 2018). I will introduce this kind of approach in the further study to improve the performance of my model.

As discussed in Section 3.5.4, the system tends to keep or shorten sentence length because of the training dataset I use. On the other hand, most simplification methods, including ours, rarely do sentence splitting.

Source	He was diagnosed with **inoperable abdominal cancer** in April 1999 .
Reference	He was diagnosed with **abdominal cancer** in April 1999 .
XLM-semi	He was diagnosed with **inoperable abdominal cancer** in April 1999 .
Ours	He was diagnosed with **lung cancer** in April 1999 .
Source	The Britannica was primarily a Scottish enterprise , as symbolised by its thistle logo , the floral emblem of Scotland.
Reference	Its logo , which is the floral emblem of scotland , shows that the Britannica was a scottish business.
UNTS	The Britannica was primarily a Scottish enterprise, as Symbolised by its Thistle logo, the chancel shield of Scotland.
Ours	The Britannica was mostly a Scottish company , as symbolised by its $\langle unk \rangle$ logo , the stained emblem of Scotland.

TABLE 3.8: Example failure cases.

3.8 Discussion

Automatic text simplification systems are mainly evaluated on data. However, the standard dataset used for evaluation, TurkCorpus, is mostly limited to lexical paraphrasing. Many other simplification operations, such as changing the sentence's syntactic structure and removing complicated redundant information, is rarely considered in TurkCorpus. The Newsela corpus applies multiple rewriting transformations, but it needs a licence to get access to it and is restricted in redistribution. On the other hand, no metrics show a strong correlation between human evaluation (Alva-Manchego et al., 2020b) and the metrics I use to evaluate the systems in the literature. The above issues motivate us to develop better metrics for guiding us to conduct more complicated simplification operations.

In the framework, the discriminator training part relies on the Wiki720k dataset. It is partitioned by the Flesch Readability Ease score, which is not the perfect partition for text similarity. It will make discriminator performance unreliable. Besides, how to measure a sentence's simplicity is still an open problem.

3.9 Conclusion

In this Chapter, I first demonstrate that there is no general linear transformation from complex to simple sentences in the learned latent space in the benchmark dataset. Then I propose a content preservation asymmetric denoising unsupervised text simplification approach. I adopt an autoencoder architecture to perform unsupervised text simplification. A novel asymmetric denoising technique was employed to model simple and complex sentences separately, which helps the simplification system to learn latent representation and features from the sentence with different complexity. Moreover, the content similarity constraint helps the system preserve the content from the original sentences when the decoder generates simplified ones. The automatic evaluation shows that the system can perform competitively compared to other unsupervised methods. The ablation study demonstrates that the proposed denoising method can efficiently improve the system performance compared with the symmetric denoising method. On the other hand, the BLEU scores and factual consistency accuracy results show that the similarity constraint technique can significantly preserve the meanings between the original and generated simple sentences.

Chapter 4

Continue-Fine-Tuning with Refined Datasets and Decoding Strategy for Text Simplification

4.1 Introduction

Sequence-to-sequence modeling is naturally fit for tasks with a source sequence and a target sequence, such as text simplification. Training a Seq2Seq model usually heavily relies on the quality of the task-specific parallel datasets. Although the recent emerging dedicated pre-trained models provide a new paradigm to train a model (i.e., fine-tune the pre-trained model with a smaller dataset for the objective task), the widely used datasets of TS also have noise and error (Vásquez-Rodríguez et al., 2021).

4.1.1 Backgrounds

Wikipedia-based datasets, such as WikiLarge, dominate model training in recent deep learning based text simplification studies (Martin et al., 2020b). However, these datasets have many errors like sentence pair misalignments,

noise sentences, and inaccurate and limited variations of simplifications (See Table 4.1). These errors negatively contribute to the training model (Vásquez-Rodríguez et al., 2021). Thus the lack of high-quality data has been a main problem in the TS field (Xu et al., 2015; Alva-Manchego et al., 2020c).

For this reason, many studies tend to find alternative datasets, such as Newsela (Xu et al., 2015). However, these alternatives require permission to get access or are insufficient in data size to train a good deep learning model (See Table 2.1). Some other methods use text mining techniques to automatically collect paraphrases to create a large training corpus for TS, then train the model on the new-built dataset (Martin et al., 2020d; Omelianchuk et al., 2021). On the other hand, some other studies explored unsupervised methods to deal with this problem. However, most of these attempts at unsupervised learning are elaborated with complicated architectures and perform far worse than supervised methods.

4.1.2 My Methods

In this chapter, I start by exploring two sentence similarity methods for cleaning the widely used WikiLarge dataset to build a refined WikiLarge in Section 4.2.1. The motivation is that the source and the target sentence in a pair should have consistent semantic meanings for TS. Furthermore, most errors are on the target side in the WikiLarge dataset. Comparing the source and target sentence similarity makes it easy to filter out misalignment, noises, and copies, which will remove most error pairs. In this way, I can build a cleaned WikiLarge for model training. Note that my method only removes the error pairs instead of correcting them. Therefore, the refined dataset will have a smaller size than the original dataset. The idea can also be extended to other tasks for cleaning parallel datasets, such as paraphrasing and text style transfer.

Recently, substantial works have shown that pre-trained models (PTMs) on the large corpus can learn universal language representations, which are beneficial for downstream NLP tasks and can avoid training a new model from scratch (Qiu et al., 2020). Fine-tuning a PTM has become a paradigm, which achieves state-of-the-art results in many fields (Church et al., 2021). Fine-tuning a model with only a relatively small dataset can outperform models trained from scratch with a large dataset. I fine-tune PTMs with my refined dataset. As Seq2Seq models naturally fit the text simplification task in structure, the pre-trained model BART (Lewis et al., 2019), is employed in my research.

I also propose a fine-tuning strategy which I call continue-fine-tuning to assist my model training. More details are given in Section 4.2.3. I demonstrate that fine-tuning a model which is pre-tuned with another similar task will boost the performance and training speed. Furthermore, a new decoding strategy exclusive for TS is also be explored.

My contributions in this chapter are as follows. 1) I first propose a method using BERT sentence similarity to refine the WikiLarge dataset, which can improve the dataset's quality. 2) I propose the continue-fine-tuning strategy with the refined dataset, which speeds up model fine-tuning and achieves good results. 3) I propose a new decoding strategy for simple text generation.

4.2 Methodology

I first refine the WikiLarge dataset to create subsets of this dataset using different sentence similarity thresholds. I then feed these datasets to train powerful pre-trained Seq2Seq models with my proposed strategy, as described in more detail next.

1	Source	They take up oxygen in the lungs or gills and release it while squeezing through the body 's capillaries.
	Target	Red blood cells are very large in number ; in women, there are 4.8 million red blood cells per microliter of blood.
2	Source	It is by far the longest of the Pauline epistles, and is considered his "most important theological legacy".
	Target	Here, the letter is addressed to the early Church in Rome.
3	Source	The elk, or wapiti (Cervus canadensis), is one of the largest species of deer in the world and one of the largest mammals in North America and eastern Asia.
	Target	It lives in Asia and eastern Europe.
4	Source	46 people perished in the accident, of whom 41 were senior year pupils of the Geschwister-Scholl-Schule in Radevormwald.
	Target	The dispatcher had seen what happened and tried to hold the train back with emergency signals, but he failed and the train disappeared behind a curve.
5	Source	Many Major League alumni have called Northern League teams home in an effort get back to the Majors.
	Target	Catskill Cougars -LRB-/O2000/O-RRB-
6	Source	The Greater Berlin Act was passed by the Prussian parliament on 27 April 1920 and came into effect on 1 October of the same year.
	Target	Pankow
7	Source	Because fronts are three-dimensional phenomena, frontal shear can be observed at any altitude between surface and tropopause, and therefore be seen both horizontally and vertically.
	Target	Low Level Jets.
8	Source	On July 11, 2007, the first new episode of Danny Phantom was aired on the Nicktoons Network.
	Target	On July 11, 2007, The first new episode of Danny Phantom was aired on the Nicktoons Network.

TABLE 4.1: Examples of error pairs with misalignment, noise or exact copy in the WikiLarge dataset. Examples 1 to 4 demonstrate misalignments, as the source and target are unrelated. Target sentences in Examples 5 to 7 are examples of noise in the target. Example 8 represents an example of a copy where the source and target are identical.

4.2.1 WikiLarge Dataset Cleaning

As a Wikipedia-based dataset, WikiLarge is regarded as the most widely used training dataset for text simplification. Many researchers (Mallinson et al., 2020; Martin et al., 2019; Omelianchuk et al., 2021) work on this dataset, despite there being many misaligned and noisy sentence pairs (see Table 4.1). In order to filter out some of these error pairs, I explore two methods to measure the similarity between the source and target sentence.

The first is an explicit method that compares the token edit distance between the source and target. It is motivated by the observation that a sentence's simplification should have a small token edit distance against the original (See examples in Table 4.2). In contrast, noise and misalignments show a substantial difference explicitly against their source sentences (See examples in Table 4.1).

The second proposed method is an implicit method (or model-based method) based on measuring sentence embedding similarity (SES) by using the Sentence-BERT (*SBERT*) (Reimers and Gurevych, 2019b). Intuitively, noise or misalignment targets differ from their source sentences in terms of SES score, while exactly copy pairs will have the inner SES score as 1.

4.2.1.1 Token Edit Distance Method

Edit distance traditionally quantifies character-level changes from one sequence to another (Navarro, 2001). For two sequences a and b with lengths i and j respectively, the edit distance can be defined in Eq. 4.1. In this work, following the settings in Vásquez-Rodríguez et al. (2021), I compute the number of changes between the original and simplified sentences through the token edit distance at the token level. To make the results comparable across text, I divide the number of changes by the original text length and obtain values between 100% (no changes) to 0% (completely different sentence).

1	Source	There is manuscript evidence that Austen continued to work on these pieces as late as the period 1809, and that her niece and nephew, Anna and James Edward Austen, made further additions as late as 1814.
	target	There is some proof that Austen continued to work on these pieces later in life. Her nephew and niece, James Edward and Anna Austen, may have made further additions to her work in around 1814.
2	Source	When Japan earned another race on the F1 schedule ten years later, it went to Suzuka instead.
	target	When Japan was added back to the F1 schedule ten years later, it went to Suzuka instead.
3	Source	It is by far the longest of the Pauline epistles, and is considered his "most important theological legacy".
	target	Here, the letter is addressed to the early Church in Rome.
4	Source	A very wide covered footbridge joins all platforms at their western ends but does not provide entry to or egress from the station.
	target	A covered footbridge connects the platforms at their western end. The footbridge does not provide entry to or exit from the station.
5	Source	Matilda died of a fever at Hedingham Castle, Essex, England and is buried at Faversham Abbey, which was founded by her and her husband.
	target	She was buried in Faversham Abbey.
6	Source	On Christmas morning he leaves Max alone, tied up in a room in an old abandoned apartment that had almost been completely burnt down.
	target	At night he leaves Max alone, tied up in a room in the apartment of an old lady on vacation.
7	Source	It originally aired on the Fox network in the United States on January 31, 1991.
	target	It first started on the Fox network in the United States on January 31, 1991.
8	Source	A shoe is an item of footwear evolved at first to protect the human foot and later, additionally, as an item of decoration in itself.
	target	A shoe is also an item of clothing.

TABLE 4.2: Examples of correct pairs in WikiLarge dataset. These examples also demonstrate that the degree of simplification varies across examples.

$$\text{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \text{lev}_{a,b}(i-1,j) + 1 \\ \text{lev}_{a,b}(i,j-1) + 1 \\ \text{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases} \tag{4.1}$$

4.2.1.2 Sentence BERT Similarity

SBERT similarity is a measure of how similar pre-trained model context sentence embeddings are. It is usually obtained by calculating the similarity of sentence embeddings. Previous methods typically transform word embeddings by a mean pooling operation to create semantic representations of the input sequence. The pooling operation takes the mean of all token embeddings and compresses them into a single vector space to create a sentence vector. They then take sentence and calculate the respective similarity between different sequences using the cosine similarity metric.

As a very successful pre-trained model in many NLP tasks, BERT can encode a sentence's meaning into densely packed contextual word embeddings. It is revealed that BERT-based text embeddings are useful for computing semantic similarity (Reimers and Gurevych, 2019b). I explore calculating the cosine similarity of two embeddings in a pair. The architecture is illustrated in Figure 4.1. *SBERT* is a siamese network architecture that can derive fixed-sized vectors for input sentences. *SBERT* adds a mean pooling operation to the output of BERT to derive a fixed-sized sentence embedding. The semantic similarities of sentences can be found by calculating fixed-sized sentence embeddings with similarity measures like cosine-similarity or Manhatten / Euclidean distance. The similarity represents the correlation of a pair of sentences. I use similarity to filter out sentence pairs with less correlation.

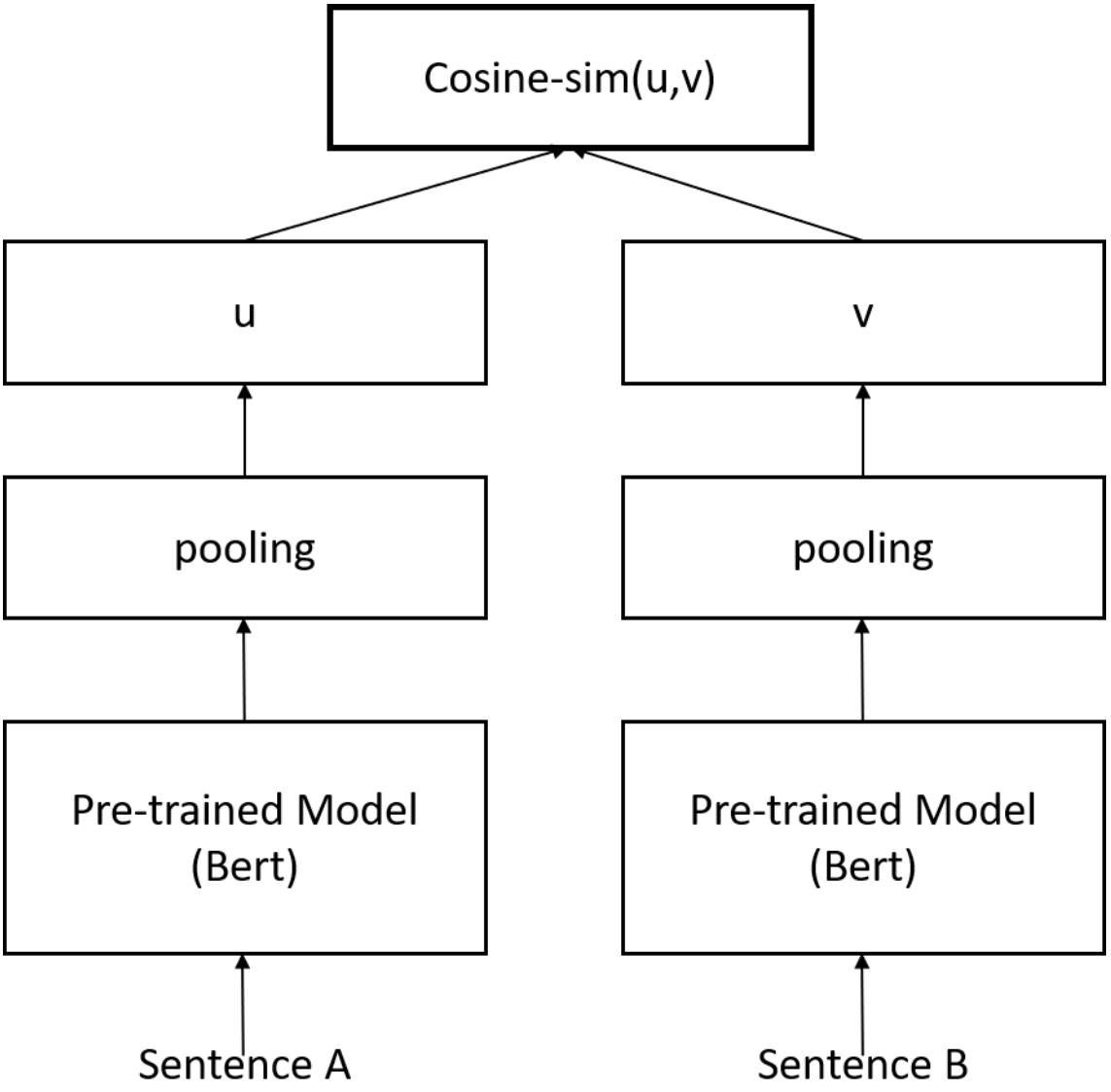


FIGURE 4.1: The structure of calculating the sentence similarity uses Sentence BERT embedding with cosine similarity. u and v are sentence embeddings. Note that the pre-trained models are fixed, and the process has no parameter update.

4.2.2 Model

Previous research has compared different model structures and results in that encoder-decoder outperform encoder-only, and decoder-only architectures (Raffel et al., 2019). My method uses the BART encoder-decoder denoising language model to conduct fine-tuning. I also choose well-fine-tuned summarization models to conduct continue-fine-tuning.

4.2.2.1 BART Pre-Trained Model

BART (Bidirectional and Auto-Regressive Transformer) is a Transformer-based pre-trained Seq2Seq autoencoder. It combines the bidirectional Encoder (BERT-like) with an auto-regressive decoder (GPT-like) into one Seq2Seq model (See figure 4.2). It is trained by reconstructing sentences from spans of text that are replaced with a single mask token. It has been proven that fine-tuning BART on various task-specific datasets will obtain good results on a wide range of text-to-text downstream tasks (Lewis et al., 2019). I fine-tune BART with the datasets mentioned above. The result shows that my data cleaning method can efficiently remove misalignments and noise data pairs to improve the model's performance.

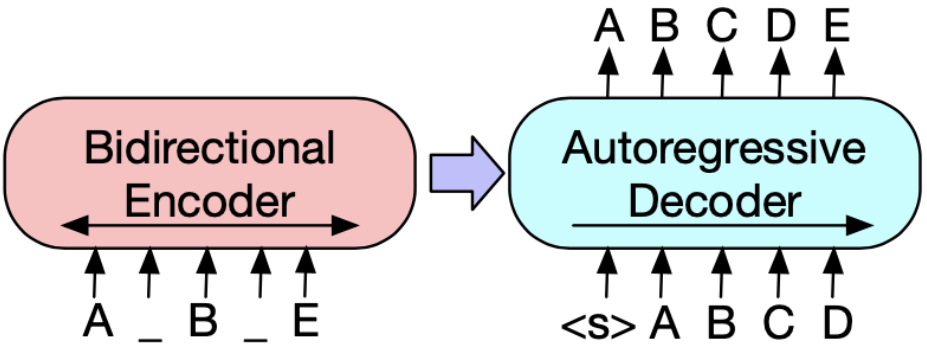


FIGURE 4.2: The structure of BART model from Lewis et al. (2019)

4.2.3 Fine-Tuning from Scratch?

Previous fine-tuning methods normally fine-tune a pre-trained model with a new task-specific objective (Qiu et al., 2020). However, the pre-trained and the fine-tuning objective can be very different from each other. For example, the pre-training objective of the BART model is sentence denoising, which is different from downstream tasks such as Translation and Question Answering. It will increase the difficulty of fine-tuning convergence. I propose a continue-fine-tuning method, which will further fine-tune a well-fine-tuned model with similar objectives. Experimental results in Section 4.4.2.2 show that this technique decreases the training time and makes the model converge easier. I choose a well-fine-tuned summarization model as the initial model for TS fine-tuning because the summarization task and TS task are closely related to each other (i.e. similar) (Zaman et al., 2020).

4.2.4 Decoding Method

The decoding strategy for traditional text-to-text methods is decoding text by maximizing the likelihood with beam search. Basically, the decoding text quality relies on the features of the training corpus. This method avoids using the TS task-specific training corpus, so I propose a tailored searching space (TSS) for text simplification, which can effectively control the lexical simplicity of the output.

4.2.4.1 Generation with a Tailored Searching Space

Same as the separation in Section 5.2.3.2, the answer space can be separated into two sub-spaces based on word frequency. The pre-trained model is trained on the whole vocabulary V, but on prediction, I only use $V^{(s)}$, the high-frequency-word and named entities subset of V, as illustrated in Figure 5.2. Let $p' = \sum_{y \in V^{(s)}} P(y_i \mid y_{1:i-1}, x)$.

In the implementation, I set the candidate words from the low-frequency subset as 0 probability (excluding named entities detected in Section 5.2.3.1), forcing the generation search to only occur on the high-frequency word space:

$$P'\left(y_i \mid y_{1:i-1}, x\right) = \begin{cases} P\left(y_i \mid y_{1:i-1}, x\right) / p' & \text{if } y_i \in V^{(s)} \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

In this way, I use a high-frequency-word-only subset of the original vocabulary when generating text. It is based on the findings that high-frequency words are easier to understand than low-frequency ones (Hu et al., 2022).

4.2.4.2 Comparing with Other Sampling Strategies

Top-k sampling and Nucleus sampling have recently become popular sampling procedures (Holtzman et al., 2019). Although TSS samples from truncated neural language model distributions, some differences exist. First, the TSS works on directed generation, in which the output is constrained by the input instead of open-ended generation. Second, TSS chooses the vocabulary subset $V^{(s)}$ based on word frequency, which is a proxy of lexical simplicity, while the other two strategies are based on candidate probability.

4.3 Experiments

I report the implementation details and experimental settings in this section. The evaluation metrics and testing data have already been introduced in Chapter 2, Section 2.1.1 and 2.1.2.

4.3.1 Data Cleaning

I follow the settings in Vásquez-Rodríguez et al. (2021) to calculate the edit distance.

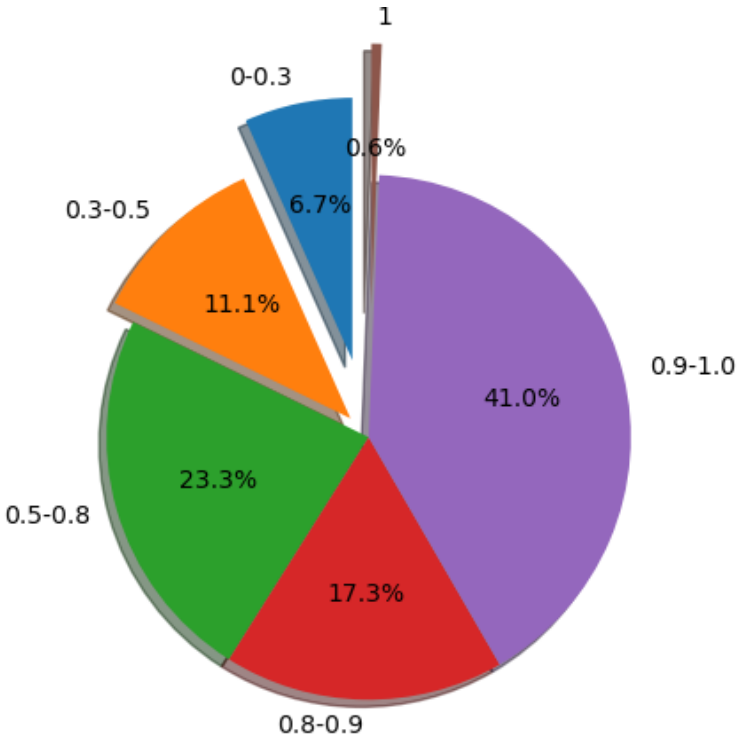


FIGURE 4.3: The proportion of WikiLarge sentence pairs in terms of different ranges of *SBERT* similarity scores

For obtaining the sentence embeddings, I choose a fine-tuned model, *'all − mpnet − base − v2'*, which is a good performance model in the *SBERT* pre-trained model repository [1]. I first feed every sentence into the *SBERT* model to get sentence embeddings of each sentence. Then I apply cosine similarities to calculate the similarities of two sentences in each pair. Then, I use the resulting score to measure each pair's similarity. I take the sentence-transformers library [2] to implement this procedure. Figure 4.3 illustrates the proportion of WikiLarge sentence pairs in terms of different ranges of BERT sentence similarity. The score represents a pair's similarity between

[1] https://www.sbert.net/docs/pretrained_models.html
[2] https://www.sbert.net/

the source and target. I use the *SBERT* similarity to filter out sentence pairs with low similarity.

4.3.2 Model Fine-tuning Details

I implement my model training with the Transformers library (Wolf et al., 2019). For continue-fine-tuning, I choose a well fine-tuned model '$bart-large-cnn-samsum$' from the HuggingFace model repository [3]. The learning rate is set to $5e-5$. Other hyper-parameters follow the model pre-fine-tuning settings. In the sentence decoding settings, the word frequency list from the *GloVe* embedding vocabulary [4] is used to estimate a word's complexity. My results are evaluated by the text simplification evaluation tool *easse* with a wide range of metrics. In order to make a fair comparison across different fine-tuning strategies, I train (or fine-tune) each proposed model with a same academic budget (4 GeForce RTX 2080 Ti GPU cards in my case).

4.3.3 Comparison Methods

Two supervised methods are reported for comparison baselines in my experiments. The first is ACCESS (Martin et al., 2020b) which uses a transformer-based Seq2Seq model for training from scratch; the second is a translation-based method SBMT (Xu et al., 2016a). Both comparison methods are trained with the WikiLarge dataset without augmentation or supplementary data. This setting is considered for a fair comparison.

4.4 Results and Analysis

I first analyse the data cleaning result and then the model fine-tuning result. Finally, I analyse the impact of my proposed decoding strategy on the result.

[3] https://huggingface.co/philschmid/bart-large-cnn-samsum
[4] https://nlp.stanford.edu/projects/glove/

4.4.1 Data Cleaning Results

Sentence similarity	Percentage	Filtered-out pairs	Remaining pairs
Edit distance	15%	84,451	211,951
	10%	29,640	266,762
	5%	14,821	281,581
	1%	2,965	293,437
SBERT similarity	40%	99,902	196,500
	30%	76,986	219,416
	20%	59,280	237,122
	15%	44,460	251,942
Full WikiLarge	0%	0	296,402

TABLE 4.3: Statistics of WikiLarge dataset refinements. I first rank sentence pairs according to sentence similarity and then filter out the top least similar pairs by different percentages.

Data cleaning aims to filter out error pairs and retain the correct pairs. In my method, the error pairs are defined by sentence pair inner similarity scores, which is a hyper-parameter. Different separations are summarised in Table 4.3 and Figure 4.3. In Figure 4.3, I observe that 6.7% of total pairs have very low similarity scores (range from 0 to 0.3) in terms of *SBERT* similarity. There are also 11.1% total pairs with a relatively low score in the range $[0.3 - 0.5)$. The rest portions are 23.3% in range $[0.5, 0.8)$, 17.3% in range $[0.8, 0.9)$ and 41.0% in range $[0.9 - 1.0)$. It is worth noting that there are 0.6% of total pairs having exactly the same meanings as their source, which are an exact copy of the source. This portion is regarded as no action and can be filtered out to refine the training data.

On the other hand, from Table 4.3 I can see that the WikiLarge dataset can have various separations by different sentence similarity metrics. For example, the number of pairs at the 15% lowest similarity is $84,451$ in terms of Edit-distance, while in terms of *SBERT* similarity, the number goes to $44,460$.

4.4.2 Model Performance

The result of model training with different data filtering-out strategies is reported in the following sections.

TABLE 4.4: Comparison of automatic evaluation metrics of different methods in TurkCorpus dataset. Training dataset *Edit_5%* means the dataset is built by filtering out 5% least similar pairs of WikiLarge by Edit distance, while *BERT_sim_5%* means doing the same thing by *SBERT* similarity. Bold fonts highlight the best results.

Name	Training dataset	SARI↑ ASSET	SARI↑ TurkCorpus	FKGL↓	EM↓	LC↓
Training from scratch method						
SBMT	WikiLarge Full	37.11	39.56	7.95	0.10	8.03
ACCESS	WikiLarge Full	40.13	**41.38**	7.29	**0.04**	7.94
Fine-tuning method						
BART_large	WikiLarge Full	37.30	39.06	8.35	0.20	8.19
BART_large	Edit_5%	38.02	39.65	7.66	0.15	8.15
BART_large	Edit_10%	38.59	39.62	7.95	0.17	8.14
BART_large	Edit_15%	38.56	39.10	8.11	0.24	8.21
BART_large	Edit_50%	38.75	39.40	8.65	0.15	8.17
BART_large	BERT_sim_5%	38.12	39.76	7.86	0.15	8.15
BART_large	BERT_sim_10%	38.50	39.30	7.51	0.11	8.15
BART_large	BERT_sim_15%	**38.91**	39.83	7.45	0.17	8.14
BART_large	BERT_sim_50%	38.25	38.77	7.65	0.15	8.17
BART_SUM	BERT_sim_15%	38.20	**40.08**	7.75	0.14	8.19
BART_SUM + Decoding	BERT_sim_15%	**41.75**	39.71	**6.84**	0.12	**7.44**

4.4.2.1 Fine-tuning with Refined Dataset

In Table 4.4, it is evident that fine-tuning methods are better than the training-from-scratch methods in terms of ASSET's SARI score, FKGL score, and LC score, even though they are training with the same dataset WikiLarge.

The best fine-tuning SARI score of TurkCorpus 40.08 goes to the BART_SUM model fine-tuning with WikiLarge *BERT_sim_15_%* which is the continue-fine-tuning in my work. The best comprehensive performance goes to the above model with the new decoding strategy with a SARI score of ASSET at 41.75, FKGL at 6.84, and LC at 7.44. Example results are shown in Table 4.5.

Overall, models fine-tuned with the refined WikiLarge are better than the models trained with full-WikiLarge considering the presented metrics in Table 4.4. It reveals that my data refining method can effectively clean the WikiLarge dataset and be used to better follow-up training results.

Source	In architectural decoration Small pieces of colored and iridescent shell have been used to create mosaics and inlays, which have been used to decorate walls, furniture and boxes.
Simplification	Small pieces of colored and shiny shell have been used to decorate walls, furniture and boxes.
Source	He advocates applying a user-centered design process in product development cycles and also works towards popularizing interaction design as a mainstream discipline.
Simplification	He favors a user-centered design process and works towards bringing interaction design into mainstream popularity.
Source	He is also a member of another Jungiery boyband 183 Club.
Simplification	He is also a member 183 Club.
Source	Each version of the License is given a distinguishing version number.
Simplification	Each license is given a number.

TABLE 4.5: Examples of simplification results generated by my best model.

4.4.2.2 Results of Continue-fine-tuning

The Continue-fine-tuning strategy tends to reduce the training time and enables the model to converge easier. Results are reported in Table 4.6. I can see that the $BART-large-cnn-samsum$ model obtains model convergence results in SARI at 40.08 by using only 12 epochs training, and with further training, the model tends to over-fit. Figure 4.4 also illustrates this trend. In contrast, the original $BART-large$ needs to train 27 epochs to get a model convergence result with the same dataset, even if they have the same model architecture. The reason is that the text summarization task is similar to the text simplification task, so the summarization task pre-fine-tuning learned knowledge could be shared with the TS task to help its model convergence. Future work could explore other pre-fine-tuning models of TS's related tasks.

TABLE 4.6: Comparison of the fine-tuning result of different initial models in the TurkCorpus test dataset. All models are trained on a refined Wiki-Large dataset *BERT_sim_15%*

Initial Model	SARI↑	Training epoch	FKGL↓	LC↓
BART-large	38.59	12	7.95	8.03
	39.83	27	7.45	8.14
T5_base	38.76	27	8.19	8.16
BART-sum	40.08	12	8.35	8.19
	39.76	20	7.66	8.15
	39.30	27	7.51	8.15

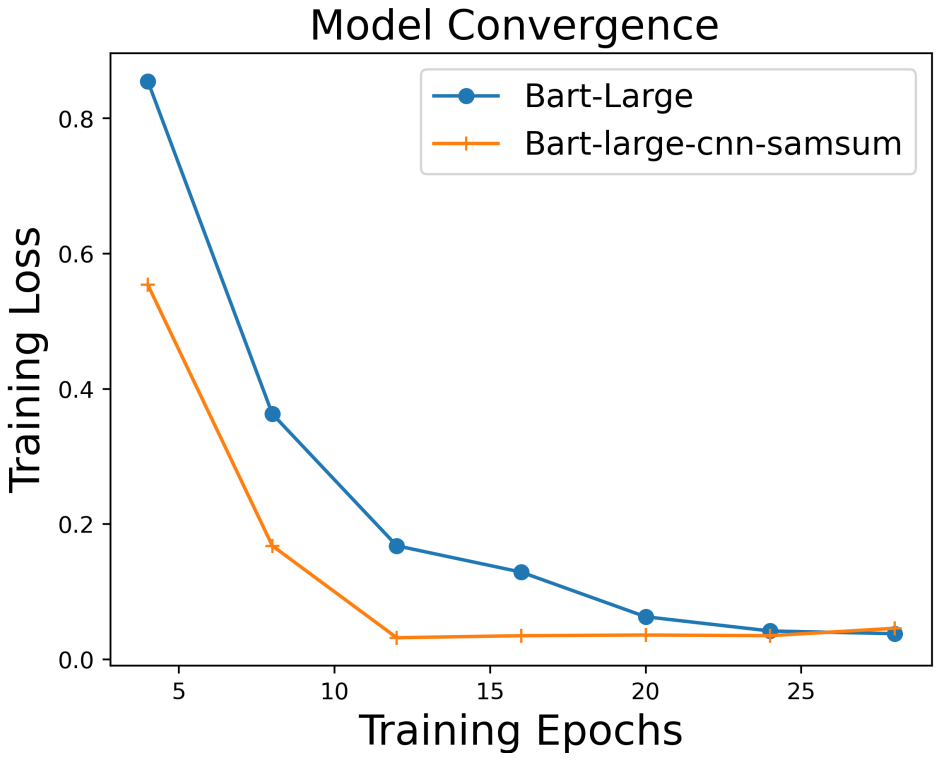


FIGURE 4.4: The Model convergence trends with the increasing of training epochs.

4.4.2.3 Results with Decoding Strategy

The traditional decoding strategy chooses the best candidates with the maximum likelihood. My strategy gives priority to high-frequency candidates when decoding. Table 4.7 illustrates that the vocabulary size of the decoding search space can significantly affect the model performance.

TABLE 4.7: Comparison of the fine-tuning result of different Decoding strategies in TurkCorpus dataset. The Decoding searching space is divided by the vocabulary sizes of a word frequency list.

Vocabulary Size	SARI↑		FKGL↓	LC↓	*SBERT* similarity↑
	ASSET	TurkCorpus			
BART-large-cnn-samsum + BERT_sim_15					
1000	42.12	37.38	**6.08**	**7.05**	85.83%
2000	**42.16**	38.74	6.49	7.30	88.90%
3000	41.75	39.71	6.84	7.44	90.56%
5000	39.76	**39.84**	7.44	7.67	93.58%
8000	38.12	39.35	7.80	7.85	95.33%
15000	38.08	38.61	8.28	8.03	96.52%
Full	39.40	39.94	8.65	8.17	**97.80**%

4.5 Discussion

My method refines the widely used text simplification dataset WikiLarge. Experimental results show that my data refining method can improve the dataset's quality. However, there are also some limitations to my methods. First, my methods only filter out dissimilar pairs and retain similar ones, while a target sentence similar to the source cannot necessarily be considered a proper simplification. It means there will be some mistakes in my refined dataset. The second is that even though the refined dataset has a higher quality, the pair numbers of the refined dataset are too small as a training set for a good transformer-based Seq2Seq model (See statistics in Table 4.3).

In addition, the best proportion of pairs that should be filtered out for the downstream task is uncertain. These issues motivate us to explore a method without a large amount of parallel data or to find a way to collect more high-quality parallel sentences.

4.6 Conclusion

In this chapter, I propose a parallel dataset refining method by *SBERT* similarity for the text simplification task. The result shows that the refining method is good for the purpose of filtering errors, which is beneficial to model training. I also propose a continue-fine-tuning strategy to help the model converge faster. Furthermore, I use a task-specific decoding strategy to boost my model performance.

Chapter 5

Text Simplification Using Pre-Trained Language Models without Fine-tuning

5.1 Introduction

Recently, there has been a paradigm shift in how large language models are used. Rather than fine-tune these models for downstream tasks, increasingly, researchers are using different strategies to make language models solve specific tasks (Liu et al., 2021a). One prominent example of this is prompting methods, where a prompt is provided to a pre-trained language model to encourage it to fulfil a particular task. The advantages of these approaches include: 1) the language capacity of the model is not down-graded by fine-tuning it to a much smaller downstream dataset; 2) the approach can be switched to a new version of a language model as it becomes available allowing easy benefit from new and improved models; 3) there is considerable saving in computation and time as no training is required. This latter point is critical when considering massive language models such as GPT-3 (Brown

et al., 2020a) where fine-tuning is impractical for many practitioners due to the vast computational resources required.

In this chapter, a new framework is proposed to conduct text simplification based on pre-trained Seq2Seq language models without retraining them. It consists of a dedicated prompting method for TS and a novel decoding strategy. Three different approaches are also explored for text simplification using pre-trained language models without fine-tuning. The first method is a back-translation method that results in paraphrases by translating the sentence to another language and then back to English using *Google translate* [1]. Several languages are considered for back-translation. It shows that performance depends on the intermediate language, but all methods achieve a reasonable level of performance. The second method is to use zero-shot in-context learning with GPT-3. This is a classic human-designed prompting method, where I instruct GPT-3 to simplify the sentence. Again, this achieves a very respectable level of performance using standard text simplification metrics. The last method is a zero-shot method that adopts the masking network BART (Lewis et al., 2020). This method masks words and requires the network to reconstruct the sentence. I use BART by masking complex words while keeping simple words and named entities. I can improve the preservation of meaning by concatenating a paraphrase obtained using back-translation via *Google translate*. The advantage of the proposed methods are that they do not require fine-tuning. Therefore, this approach can be quickly adapted to a new version of the pre-trained language model and does not require considerable training time. Another significant advantage of my methodology is that I can change the size of the vocabulary used by the model to control the complexity of the sentences generated.

The ability to adapt the level of simplification to match the reading level of the user is not a focus of previous work. In part, this is due to the standard metric of SARI used to evaluate text simplification, treating the problem as a

[1] https://translate.google.com/

single-objective problem rather than the multi-objective problem that it is -— I need to trade-off between simplicity and accuracy. The level of performance of all three approaches proposed and extensively evaluated in this work is near state-of-the-art to fine-tuned models though the final method achieves state-of-the-art SARI scores.

The contributions of this chapter are 1) I present an investigation of the use of pre-trained language models without fine-tuning and demonstrate excellent performance on text simplification; 2) I propose an adaptive prompt for each input suitable for moderately-sized pre-trained language models; 3) I propose a novel method that makes sentence simplicity controllable and obtains state-of-the-art FKGL scores.

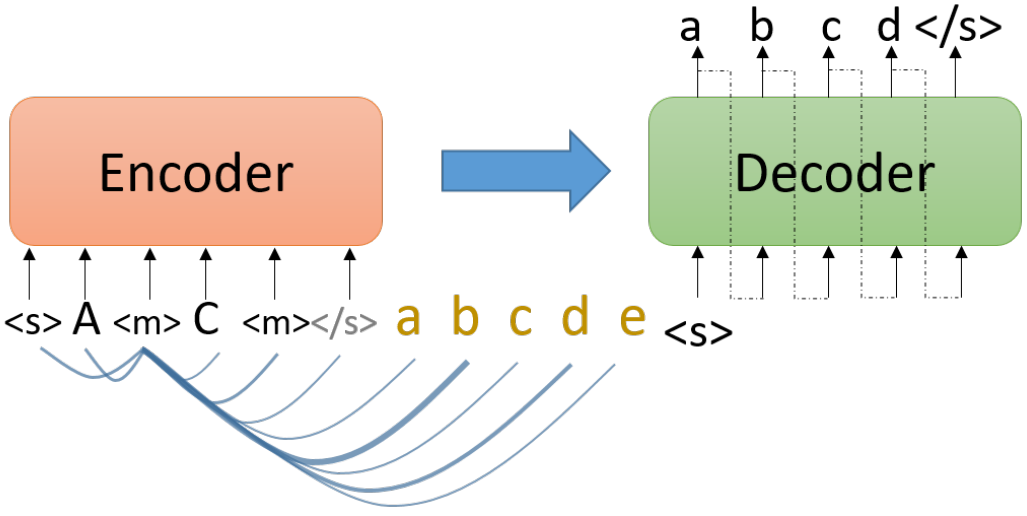


FIGURE 5.1: The structure of this method. Inputs to the encoder have been corrupted by replacing difficult tokens with mask tokens ($\langle m \rangle$ in this figure). The paraphrase (the golden sequence following $\langle /s \rangle$) is concatenated as a context prompt to help the corrupted sentence attend semantic meaning of the origin sentence.

5.2 Overview

The main issue of text simplification is the lack of high-quality parallel training data (Al-Thanyyan and Azmi, 2021). This hinders supervised training for

text simplification from scratch as well as fine-tuning pre-trained models. my method applies implicit task descriptions as part of the input to circumvent this issue by reformulating the TS task to a denoising language model task.

The main idea of this method is motivated by the findings that PLMs tend to generate high-frequency tokens (Jiang et al., 2019; Gu et al., 2020; Meister et al., 2020), which are easier to understand than low-frequency ones (Hu et al., 2022). This allows us to make use of PLMs without fine-tuning. The core logic of this method is corrupting a sentence by replacing complex parts with prompting tokens, which will trigger a pre-trained denoising language model to reconstruct the corrupted sentence to its simple counterpart (See Figure 5.1). I generate an input self-adaptive prompt to enable the model to focus on specific parts.

Moreover, for meaning preservation, I add a pre-generated paraphrase $\tilde{\mathbf{x}}$ as the context prompt. In this way, the template is used to transfer the TS task as the sentence reconstruction task. The reconstruction processing, which will modify sentence structures while generating a new sentence with simple words, is different from mere text infilling (Petroni et al., 2019b; Cui et al., 2021b). I describe each component in more detail.

5.2.1 Simplification Framework

The simplification framework consists of two parts: 1) a noising or prompting function $f(\boldsymbol{x})$ which given a sentence $\boldsymbol{x}$ returns a prompt or noisy version $\hat{\boldsymbol{x}} = f(\boldsymbol{x})$ and 2) a language model $P_L(\boldsymbol{x}' \mid \hat{\boldsymbol{x}})$ which given an input sentence $\hat{\boldsymbol{x}}$ defines a probability over sentences $\boldsymbol{x}'$. The simplified sentences are then given by

$$\boldsymbol{x}_{\text{simp}} = \underset{\substack{\boldsymbol{x}'=(x'_1,x'_2,\ldots x'_n) \\ x_i \subset V}}{\operatorname{argmax}} P_L\left(\boldsymbol{x}' \mid f(\boldsymbol{x})\right) \tag{5.1}$$

where V is some vocabulary set. Within this framework, the language model L, the prompting function f and the vocabulary V can be changed. As usual for language models, it is easy to find the most probable next word. I use beam search to obtain a good approximation to the most probable sentence.

In this chapter, I explore four different language models: GPT-3, BART-Base, BART-Large, and BART-Large-CNN, which is a fine-tuned version of BART-Large on CNN-DM (Nallapati et al., 2016) an abstractive text summarisation dataset (Lewis et al., 2020). I consider a range of different prompting/noising functions $f(\boldsymbol{x})$, which I outline in more detail below. I also consider different vocabulary sets where I select the N most commonly occurring words for various N.

5.2.2 Noising/Prompting Functions

The most important part of the model is the choice of denoising or prompting functions $f(\boldsymbol{x})$. That is, given a sentence $\boldsymbol{x}$ that I want to simplify, $f(\boldsymbol{x})$ outputs a sentence (i.e. template) that is input into the language model. The simplest case is to use the original sentence, $\boldsymbol{x}$. For the BART-Based models, I also use a noisy version of the original sentence. I consider two noising functions: $f_r(\boldsymbol{x})$ where I replace randomly chosen words with a mask token "$\langle mask \rangle$", and $f_n(\boldsymbol{x})$ where I retain named entities and common words, but mask less common words. I also consider appending sentences to the initial sentence. The language model reconstruction of these appended sentence are ignored, but they are used to aid the preservation of semantic meaning by providing context to the first sentence. I consider appending the original sentence, $\boldsymbol{x}$, but also a paraphrase $f_p(\boldsymbol{x})$ obtained by back-translation of $\boldsymbol{x}$. I experiment with six different prompting functions from B_0 to B_5 formalised as follows:

B_0**:** $f(\boldsymbol{x}) = \boldsymbol{x}$
B_1**:** $f(\boldsymbol{x}) = f_n(\boldsymbol{x})$

B_2: $f(\boldsymbol{x}) = f_n(\boldsymbol{x}) + \langle /s \rangle + \boldsymbol{x}$

B_3: $f(\boldsymbol{x}) = f_n(\boldsymbol{x}) + \langle /s \rangle + f_p(\boldsymbol{x})$

B_4: $f(\boldsymbol{x}) = \boldsymbol{x} + \langle /s \rangle + f_p(\boldsymbol{x})$

B_5: $f(\boldsymbol{x}) = f_r(\boldsymbol{x}) + \langle /s \rangle + f_p(\boldsymbol{x})$

where $+$ denotes concatenation and $\langle /s \rangle$ is the end-of-sentence token. All the above definitions are demonstrated in Table 5.1.

For GPT-3, I consider two human interpretable prompts to guide the model to simplify sentences: $\boldsymbol{p}_1$ = "Summarize this for a second-grade student: " and $\boldsymbol{p}_2$ = "Simplify this for easy reading: "

G_1: $f(\boldsymbol{x}) = \boldsymbol{p}_1 + \boldsymbol{x}$

G_2: $f(\boldsymbol{x}) = \boldsymbol{p}_2 + \boldsymbol{x}$.

Examples of all of the sentences (templates) are given in Table 5.1.

5.2.3 Template Building

In this method, an input $\boldsymbol{x}$ is modified using templates into a textual sequence denoted as $f(\boldsymbol{x})$. For each input $\boldsymbol{x}$, there is a customized prompt adapted to it. I explain how to build the templates in this section.

5.2.3.1 Named Entities Keeping

Complex sentences usually contain named entities. However, most named entities are recognised as low-frequency words but might be the key to a sentence's meaning (Nadeau and Sekine, 2007). Therefore, this method uses the Spacy (Partalidou et al., 2019) tool with the "en_core_web_trf" pipeline to detect named entities as a preprocessing step in order to keep them in model input $\hat{\mathbf{x}}$. This step also reduces the reconstruction complexity of the language model.

Templates	Examples
B_0	His next work, Saturday, follows an especially eventful day in the life of a successful neurosurgeon.
B_1	His next work $\langle mask \rangle$ Saturday $\langle mask \rangle$ day in the life of a $\langle mask \rangle$
B_2	His next work $\langle mask \rangle$ Saturday $\langle mask \rangle$ day in the life of a $\langle mask \rangle$ $\langle /s \rangle$ His next work, Saturday, follows an especially eventful day in the life of a successful neurosurgeon.
B_3	His next work $\langle mask \rangle$ Saturday $\langle mask \rangle$ day in the life of a $\langle mask \rangle$ $\langle /s \rangle$ His next work on Saturday marks a particularly pivotal day in the life of a successful neurosurgeon.
B_4	His next work, Saturday, follows an especially eventful day in the life of a successful neurosurgeon. $\langle /s \rangle$ His next work on Saturday marks a particularly pivotal day in the life of a successful neurosurgeon.
B_5	His $\langle mask \rangle$ $\langle mask \rangle$, Saturday , $\langle mask \rangle$ an especially eventful day $\langle mask \rangle$ $\langle mask \rangle$ life of a successful $\langle mask \rangle$. $\langle /s \rangle$ His next work on Saturday marks a particularly pivotal day in the life of a successful neurosurgeon.
G_1	**Summarize this for a second-grade student:** His next work, Saturday, follows an especially eventful day in the life of a successful neurosurgeon.
G_2	**Simplify this for easy reading:** His next work, Saturday, follows an especially eventful day in the life of a successful neurosurgeon.

TABLE 5.1: Examples of the different proposed templates. Templates for this method: B_0. original input. B_1. complex masked input. B_2. concatenate **complex masked input**, $\langle /s \rangle$ and **original input**. B_3. concatenate **complex masked input**, $\langle /s \rangle$ and an **input paraphrase**. B_4. concatenate **original input**, $\langle /s \rangle$ and an **input paraphrase**. B_5. concatenate **random masked input**, $\langle /s \rangle$ and an **input paraphrase**. Templates for GPT-3 zero-shot learning method: G_1: prompt of "**Summarize this for a second-grade student:**" concatenated to input. G_2: prompt of "**Simplify this for easy reading:**" concatenated to input.

5.2.3.2 Sentence Corruption

Previous studies create external prompts as an additional part of the input (Liu et al., 2021b; Shin et al., 2020; Schick and Schütze, 2021a). In this method, I first detect complex text and then replace a span of difficult text with a $\langle mask \rangle$ token, as shown in Table 5.1. The $\langle mask \rangle$ token works as an indicator to guide the denoising language model to locate which part needs to be

reconstructed. I also propose a random masking method in comparison. The efficiency of various approaches is discussed in Section 5.4.1.1.

Complex Text Masking In this section, I explain the noise function $f_n(\cdot)$ used to corrupt the input $\boldsymbol{x}$ in this method. Most complex sentences only have some difficult parts. It often occurs that high frequency words are easier and low frequency words are harder. I use a word frequency ranking list [2] as the threshold to separate high and low-frequency words. In this way, a span of difficult words are replaced with a single $\langle mask \rangle$ token, which can be detected by denoising language model. The threshold is a hyperparameter in this method.

Random Words Masking Motivated by BART's masked sentence reconstruction pre-training objective, I propose a method $f_r(\cdot)$ to randomly mask a portion of the input sentence to trigger the model to do sentence reconstruction. This is a baseline comparison method in contrast to the complex text masking method. I also use the $\langle mask \rangle$ to replace the randomly chosen part of a sentence for subsequent processing.

5.2.3.3 Paraphrasing Context as Prefix Prompt

The original meaning of a heavily corrupted sentence is hard to reconstruct without additional information. I use a paraphrase of the original sentence as additional context to help the language model reconstruct the masked sentence (See Table 5.1). In this way, this method finds a pattern to make the LM focus on reconstructing the corrupted sentence considering the paraphrase.

Back-translation is a simple way of generating paraphrases (Prabhumoye et al., 2018), whereby, I translate English sentences to another language, and

[2] https://www.wordfrequency.info/

then back to English. I use the *Google translate* API (application programming interface) to conduct back-translation. I choose 10 languages in 8 different language families to increase the variety of the back-translation results. I test the performance of different back-translation operations in the TurkCorpus dataset (Xu et al., 2016a), and the results are shown in Table 5.2. This method does not require training and naturally preserves semantic meaning. Thus these results can also work as comparison results. Furthermore, Based on the feature that back-translation tend to use more common representation to generate a sentence from an original complex text, I use back-translation to generate paraphrases of complex sentences. I compared back-translation with different languages and find out that back-translation with languages in different language families tend to generate varied sentences which is beneficial for dealing with text simplification data deficiency. I also find that back-translate a sentence with a language from a different language family tend to obtain a more various paraphrases.

In Table 5.2, I show that English-Thai-English back-translation gets the highest SARI score in the TurkCorpus dataset and the lowest FKGL score, implying it can get the best simplification results. We, therefore, use English-Thai-English back-translation to get the paraphrases $f_p(\cdot)$ in the templates.

Language	SARI	BLEU	FKGL	Sen-Sim
Arabic	39.02	66.70	9.82	96.38%
Bengali	38.17	58.50	9.57	93.96%
French	38.30	76.51	9.70	**97.66**%
Hindi	38.94	71.91	9.65	96.70%
Japanese	38.89	58.77	8.84	96.08%
Somali	38.00	57.72	9.24	93.42%
Telugu	39.05	64.45	9.33	95.64%
Thai	**39.21**	59.47	**8.32**	95.52%
Chinese	39.02	54.64	9.19	96.83%
Esperanto	37.92	**78.34**	9.70	97.46%

TABLE 5.2: Back-translation based text simplification results on the TurkCorpus dataset. I observe that English-Thai-English performs best on both SARI and FKGL.

5.2.4 Generation with Simple-Word Beam Search

Beam search is the default decoding strategy in many generation methods. In this section, I propose a simple-word beam search method for text simplification, which can effectively control the lexical simplicity.

It is intuitive that high-frequency words are simpler and low-frequency words are harder. I take the word frequency list table [3], which ranks words by their frequency calculated from large amounts of corpus. It can be separated as two sub-spaces based on word frequency.

The pre-trained model is trained on the full vocabulary, but on prediction, I only use the simple-word subset of full vocabulary as illustrated in Figure 5.2. In my implementation, I set the candidate words from the low-frequency subset as 0 probability, forcing the beam searching only on high-frequency word space.

In this way, I use a simple-word only subset of the original vocabulary when predicting. This is based on the intuition that difficult expressions can be replaced by simple ones (excluding Name Entities). That is why I keep Name Entities in my template.

5.3 Experimental Results

In the experiments, I use the BART-Large model and set the difficult vocabulary masking threshold to 1250. For random masking in B_5, I set the masking rate to 0.15. The model is implemented using HuggingFace's transformers library (Wolf et al., 2019). I choose the validation set of ASSET for model selection and hyper-parameter tuning.

[3] https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists

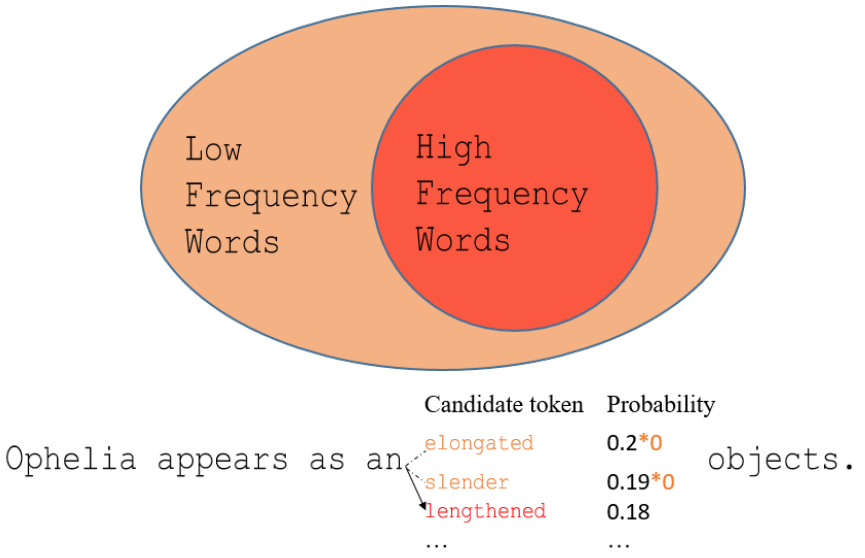


FIGURE 5.2: Beam searching only in the high-frequency word space. I multiply 0 with probabilities of words from the low-frequency space to avoid searching for less frequent (i.e. complex) words.

5.3.1 Comparison Methods

As this method was built without fine-tuning or labels, the main comparison methods are GPT-3 zero-shot predictions, and the best Back-translation result (Thai) from Section 5.2.3.3. I also choose two state-of-the-art unsupervised training methods (MUSS (Martin et al., 2020c), Trans-SS (Lu et al., 2021)) as well as one supervised method (ACCESS (Martin et al., 2020a)) for comparison.

5.3.2 Evaluation Metric

I calculate the SARI (Xu et al., 2016a) score of two evaluation datasets, namely ASSET (Alva-Manchego et al., 2020a) and TurkCorpus, to evaluate the text simplification performance. ASSET is considered better than TurkCorpus, because ASSET has more references with more abstractive transformations (Alva-Manchego et al., 2020a). It is also the primary evaluation set to calculate the SARI score in the results.

Other reference-free automatic evaluation metrics used are FKGL (Kincaid et al., 1975) and lexical complexity (LC) score to estimate the simplicity of the generated outputs. All the above metrics are implemented in the EASSE framework (Alva-Manchego et al., 2019).

Previous studies have stated that BLEU is not suitable for evaluating text simplification (Sulem et al., 2018) and BERTScore (Zhang et al., 2020) is a better substitution (Scialom et al., 2021). Therefore, the results present BERTScore F1-score of references and system output instead of BLEU.

Meaning preservation is an important evaluation metric for text simplification, which previous research has neglected. I propose comparing sentence similarity (Sen-Sim) between original and system outputs to evaluate the meaning preservation. For implementation, I use the best average performance pre-trained model *all-mpnet-base-v2* [4] of Sentence-BERT (Reimers and Gurevych, 2019a) to obtain the sentence similarity results.

5.3.3 Main Results

Table 5.3 summarises the results on the different context prompts considering the various templates proposed. I note that the B_1 template only masks input tokens, which often removes the semantic meaning of the sentence resulting in a low Sen-Sim score of 76.94% (with the BART-Large model). The B_2 template (which concatenates a masked sentence with the original sentence) results in the highest Sen-Sim score of 96.93%, but with low SARI scores on both evaluation datasets as well as a high FKGL score of 8.40. The BART-Large model with the B_3 template can generate a good output with a 41.53 SARI score on ASSET.

[4] models listed in website https://www.sbert.net/index.html

Name	ASSET		TurkCorpus		FKGL↓	LC↓	Sen-Sim↑
	SARI↑	BERTscore↑	SARI↑	BERTscore↑			
Original test set	20.73	0.993	26.29	0.959	10.01	8.34	100%
Training Method							
ACCESS	40.13	0.895	41.38	0.901	7.29	7.94	94.81%
MUSS	42.65	–	40.85	–	8.79	–	–
Trans-SS	42.69	0.938	41.97	0.929	8.97	8.13	95.19%
Tuning-Free Method							
GPT-3 + G_1	40.77	0.907	39.72	0.902	8.46	8.20	93.34%
GPT-3 + G_2	42.23	0.790	37.75	0.794	7.15	8.10	81.65%
Back-Translation(Thai)	41.41	**0.912**	39.21	0.900	8.33	8.24	95.52%
BART-Large + B_1	30.23	0.781	36.65	0.776	7.85	7.65	76.94%
+ B_2	33.48	0.820	35.16	0.800	8.40	8.30	96.93%
+ B_3	41.53	0.742	38.82	0.734	6.90	8.16	91.92%
BART-Base + B_3	40.04	0.882	39.36	0.874	8.63	8.10	90.68%
BART-Large-CNN + B_0	32.92	0.932	35.78	0.941	7.61	8.30	96.82%
+ B_3	42.55	0.881	38.92	0.882	7.09	8.23	93.58%
+ B_5	41.37	0.889	39.28	0.892	7.62	8.27	92.66%
+ B_5 + V2000	42.52	0.788	36.75	0.784	5.40	7.61	84.03%
+ B_4	35.17	0.930	37.16	0.941	8.14	8.31	**97.59**%
+ B_4 + V2000	**43.24**	0.848	39.00	0.858	**5.36**	**7.55**	88.32%
+V5000	41.31	0.904	**39.79**	**0.920**	6.37	7.84	92.37%

TABLE 5.3: Summary of the overall findings, comparing the proposed approaches to the unsupervised methods (MUSS and Trans-SS) as well as the supervised method (ACCESS) on the ASSET and TurkCorpus. The best performing method uses BART-Large-CNN +B_4+V2000 and beats the state-of-the-art SARI on ASSET as well as having far lower FKGL and lower LC.

Wang et al. (2022b) demonstrates that with multi-task fine-tuning, a masked language encoder-decoder pre-trained model performs best in zero-shot generalization. BART-Large-CNN with original inputs only $+B_0$ does not perform well. However, when it is fed with the adaptive context prompt template $+B_4$ with the 2000 vocabulary constraint answer space, it obtains the best SARI score at 43.24 on ASSET and the best FKGL score of 5.36 and LC score of 7.55. Comparing BART-Large-CNN results with complex words masking $+B_3$ and with random words masking $+B_5$, $+B_3$ obtains better results. This is because $+B_3$ can in fact locate the complex words.

The Back-translation (Thai) paraphrasing result obtains high SARI scores and BERTscore, but the FKGL and LC scores are high as well. It means that the paraphrasing method cannot simplify the input very much. The best result is not only better than the GPT-3 fix prompt zero-shot methods, which is a significantly larger model than the chosen BART-Large model, but also outperforms the training-based comparison methods.

The results show that with the vocabulary constraint, this method can achieve much lower FKGL and LC scores than others, demonstrating that I can make the output sentences simpler with the vocabulary constraint. Table 5.4 shows that with different answer space choices, the output sentence is controllable in terms of simplicity. Figure 5.2 illustrates the variation in results with respect to the chosen vocabulary size of the answer space. I also observe that in this method, hyper-parameters make a significant difference.

5.3.4 Human Evaluation

I follow the evaluation setups in Kumar et al. (2020) to measure the Adequacy, Simplicity and Fluency of resulting sentences on a five-point Likert scale. The human evaluation was conducted on the ASSET dataset. Three post-graduate students are recruited as evaluators (one native English speaker and two non-native fluent English speakers). Forty sentences are randomly selected from

Size	ASSET		TurkCorpus		FKGL	Sen-Sim
	SARI	BLEU	SARI	BLEU		
BART-Large-CNN + B_4						
1000	**43.3**	42.7	38.0	39.6	**5.0**	85.5%
2000	43.2	52.2	39.0	49.5	5.4	88.3%
3000	42.4	56.7	39.3	54.6	5.7	90.1%
5000	41.3	64.4	**39.8**	63.3	6.4	92.4%
8000	40.0	68.6	39.3	68.8	6.9	94.0%
15000	38.8	73.9	38.6	74.7	7.4	95.5%
30000	37.1	**77.4**	38.1	**78.8**	7.8	**96.8**%

TABLE 5.4: An evaluation of the effects of vocabulary size in beam search for sentence generation of model BART-Large-CNN + Template B_4. Size 1000 means I only choose the top 1000 high-frequency words to generate sentences.

the dataset, and each evaluator is given the same simplified results of the sentences from different models for rating.

The results are reported in Table 5.5. The results show that my best method achieves the best average score at 4.23, and my best method obtains the highest Simplicity score at 4.27, which is consistent with the result of automatic evaluation. The Back-translation results receive the highest Adequacy score for its direct round-trip translating function, which can preserve the content well. GPT-3 $+G2$ performs the best in Fluency at 4.55 because the model is huge and trained with enormous data. My method achieves the second best at 4.50. Text simplification needs to find a balance between these three measures. Although my method does not obtain the highest score in every human evaluation measurement, the average score of my method reaches the highest, which confirms my method's effectiveness, even compared with training-based methods.

Method	Simplicity	Adequacy	Fluency	Avg
Trans-SS	3.24	4.10	3.90	3.75
GPT-3 + G_2	4.10	3.20	**4.55**	3.95
Back-T	3.33	**4.50**	4.40	4.08
My-best	**4.27**	3.92	4.50	**4.23**

TABLE 5.5: Human evaluation on ASSET. Simplicity, Adequacy, Fluency, and their average(Avg) score are reported in this table based on 1-5 Likert scale. Back-T is the back-translation(Thai) results and My-best is the BART-Large-CNN + B_4 + $V2000$ results.

5.4 Ablation Study

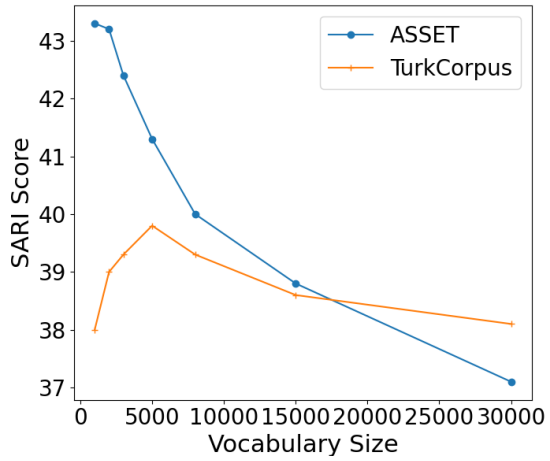
5.4.1 Sentence Masking Rate

I consider two ways to corrupt the sentence via masking as follows.

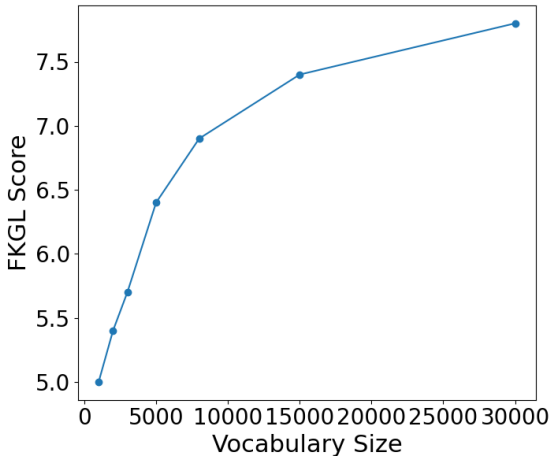
5.4.1.1 Masking with Different Thresholds

Sentences are corrupted sentences by replacing complex parts with a $\langle mask \rangle$ token. In template B_3, the complex parts are recognized by those beyond the simple word thresholds. Table 5.6 reports the results of this method with different threshold selections, Figure 5.3 also illustrates the result changes in different metrics.

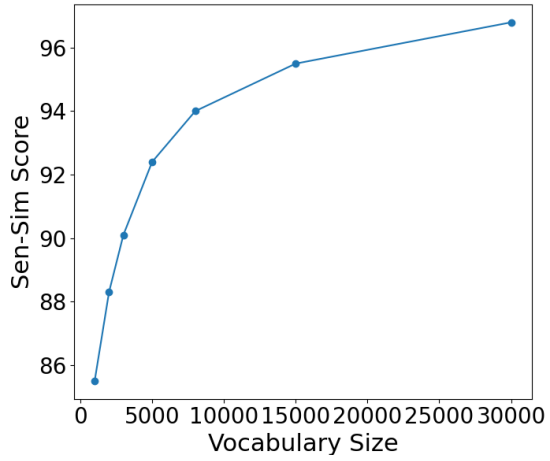
The results show that with a large threshold, this method will mask a small portion of words in B_3, then the model only fills in a few individual words, which leads to a higher SARI score on TurkCorpus in which most references only have lexical paraphrasing (Alva-Manchego et al., 2020a). On the contrary, with a small threshold, the model has to reconstruct the heavily corrupted sentences, which leads to a more abstractive generation and a higher SARI score on ASSET.



(A) SARI scores of two evaluation sets



(B) FKGL scores



(C) Sen-Sim scores

FIGURE 5.3: Results in Different Simple Vocabulary Sizes

Threshold	SARI↑ ASSET	SARI↑ TurkCorpus	FKGL↓
BART-Large-CNN + B_3			
200	**42.62**	38.81	**6.96**
450	42.33	38.66	7.19
850	42.45	38.90	7.05
1250	42.55	38.92	7.09
3000	42.07	39.60	7.56
5000	41.27	**39.89**	7.78

TABLE 5.6: Results of different corrupted sentences with various simple word thresholds.

5.4.1.2 Masking Randomly with Different Rates

Original BART models randomly mask 15% tokens of input for pre-training. Recent research indicates that increasing the masking rate to 40% can obtain better performance for masked language models (Wettig et al., 2022). I report results of different random masking rates of the BART-Large-CNN model with template B_5 in Table 5.7. The results show that masking 40% tokens in B_5 template also outperforms other masking rate settings in the SARI score of the ASSET. Results in Table 5.7 also indicate that excessive masking will blur the intention of the template to reconstruct a sentence, degrading the final results.

Masking Rates	SARI↑ ASSET	SARI↑ TurkCorpus	FKGL↓
BART-Large-CNN + B_5			
10%	40.93	**39.12**	7.51
15%	41.37	39.28	7.60
20%	41.26	38.83	7.52
40%	**42.33**	39.00	7.08
50%	42.23	38.70	6.90
80%	41.76	36.53	**5.78**

TABLE 5.7: Comparison of automatic evaluation metrics with different random masking rates.

5.4.2 Vocabulary Size of Answer Space

I report results on constraining the simple vocabulary size when searching in answer space in Table 5.4. The results show that with a small vocabulary

size, the model can generate very simple sentences (low FKGL score), but this slightly harms the semantic meanings (lower Sen-Sim score). By increasing the vocabulary size, the complexity of generation increases, as seen by the FKGL score and the SARI and Sen-Sim scores increasing. I can therefore adjust the answer space size to balance the simplicity and meaning preservation of the reconstruction.

5.4.3 Different Pre-trained Models

Now I consider the different pre-trained models with different training objectives. T5 (Raffel et al., 2019) is a text-to-text model pre-trained with the text infilling objective, but it only generates discrete predictive words instead of the whole sentence. I choose T5 with template B_3 (See Table 5.1) all other settings follows Section 5.3. The results reported in Table 5.8 indicate that the BART models get better results than the T5 model. This indicates that the pre-training objective matters a lot. On the other hand, the BART-Large model achieves better results than the BART-Base model, which indicates that model size also matters with my method. Examples are shown in Table 5.9

Model	SARI↑		FKGL↓
	TurkCorpus	ASSET	
BART-Base	**39.71**	40.49	7.10
BART-Large	38.82	**41.53**	**6.90**
T5	38.59	39.27	7.56

TABLE 5.8: Comparison of automatic evaluation metrics of different pre-trained language models.

5.5 Discussion

5.5.1 Prompt Interpretability

An ideal interpretable prompt should include easy-to-understand tokens that clearly describe the task and explicitly lead the model to solve problems. This method explicitly reformulates a text simplification problem into a denoising

Source	He is also a member of another Jungiery boyband 183 Club.
BART-Base	He is also a member of boyband 183 Club.
BART-Large	He is also a member of 183 Club.
T5	He is also a 183 Club member.
Source	It is derived from Voice of America (VoA) Special English.
BART-Base	It comes from Voice of America VoA Corps Special English.
BART-Large	It comes from Voice of America Special English.
T5	This is from VOA Learning English.
Source	The incident has been the subject of numerous reports as to ethics in scholarship.
BART-Base	The incident has been the subject of many reports as to ethics in scholarship.
BART-Large	The incident has been the subject of many reports on scholarship ethics.
T5	The incident has been the subject of many academic ethics reports.

TABLE 5.9: Examples of simplification results generated by different pre-trained models.

problem with specific difficult text corruption and generated context. Furthermore, the generation space is also reduced into a visible simple words subspace. These advantages make this method more interpretable than previous fine-tuning methods.

5.5.2 Answer Space Constraint

Constraining the answer space will avoid using difficult-word candidates when generating text. However, those candidates may have a higher probability of the original pre-trained language model, as illustrated in Figure 5.2. The smaller the search space I use, the more suboptimal candidates will be accumulated. When the original language model's candidates are insufficient, constraining its answer space (i.e. blocking some candidates) will harm the performance. Thus, constraining the answer space may not be suitable for small language models with a small amount of training datasets. This is a topic for future research.

5.5.3 Limitations of Suboptimal Discrete Prompts

The templates in this method are built based on intuition, though this method obtains a good result in text simplification. They are discrete and can be considered suboptimal. The vocabulary size of the simple answer space also faces the same suboptimal issue in this method.

5.5.4 Limitations of Generation Diversity

A key component of this method is the constrained answer space. It can effectively reduce the text complexity of the system output. However, narrowing the answer space also harms the diversity of the text generation.

5.5.5 Limitations of Complex Text Detection

my method can only detect complex text at the lexical level rather than the phrase level. It may have a disadvantage because some idioms or complex structures that can be considered difficult will be kept in the model input. One solution is that I can use a more advanced phrase database, such as Simple PPDB used in Section 3.3.4 Chapter 3.

my current method sacrifices flexibility but to a limited extent. There is a trade-off between using high-frequency words and sentence simplification. A sentence with rare words can hardly be treated as simple. This method is not a simple lexical substitution. It is to replace difficult text as $\langle mask \rangle$ to trigger the pre-trained language model to reconstruct the sentence. Although this might not always be the case, there is a strong prima facie case that using frequently used words are likely to improve comprehension. Using frequently used words makes this approach very straightforward to implement, and can be seen as an advantage of this approach.

5.6 Conclusion

This chapter proposes a zero-shot method using a moderately-sized denoising pre-trained model with adaptive prefix context prompts for text simplification. It uses word frequency to detect difficult words and replace them with $\langle mask \rangle$ tokens to trigger the denoising language model to reconstruct the sentence. It also constrains the appearance of low-frequency words in the answer space to simplify further. The results show that I can control the simplicity of output sentences by controlling the size of low-frequency words in the answer space. It shows that the efficient method (with a much smaller LM) outperforms GPT-3 zero-shot method. The results also show that an abstractive text summarization pre-finetuning can strengthen the zero-shot performance in the text simplification task.

5.7 Example Results

TABLE 5.10: Result examples obtained with the different templates.

Templates	**Examples**
B_0	The island chain forms part of the Hebrides, separated from the Scottish mainland and from the Inner Hebrides by the stormy waters of the Minch, the Little Minch and the Sea of the Hebrides.
Results	
Back-Translation(Thai)	The island chain is part of the Hebrides, separated from the Scottish mainland and from the Inner Hebrides by the stormy currents of Minch, Little Minch and the Sea of the Hebrides.

B_1	The $\langle mask \rangle$ of the Hebrides $\langle mask \rangle$ from the Scottish $\langle mask \rangle$ and from the Inner Hebrides by the $\langle mask \rangle$ of the Minch $\langle mask \rangle$ the Little Minch and the Sea of the Hebrides $\langle mask \rangle$
BART-Large-CNN	
+ B_1	The island forms part of the Hebrides. It is separated from the Scottish by waters of the Minch.
+ B_2	The island chain forms part of the Hebrides, separated from the Scottish mainland by the Minch, the Little Minch and the Sea of the hebrides.
+ B_3	The Hebrides are a chain of islands off the coast of Scotland. The islands are separated from the Scottish mainland by the Sea of Minch.
+ B_4 +V2000	The Minch, the Little Minch and the Sea of the Hebrides form part of the Scottish coast. The islands are separate from the Inner Hebride and the Scottish Highlands.
GPT-3	
+ G_1	The Hebrides are a chain of islands located off the coast of Scotland. The chain is separated from the mainland and from the Inner Hebrides by the stormy waters of the Minch, the Little Minch and the Sea of the Hebrides.
+ G_2	The Hebrides are a chain of islands that form part of the Hebrides, which are in turn separated from the Scottish mainland by the stormy waters of the Minch, the Little Minch and the Sea of the Hebrides.

B_0	Fearing that Drek will destroy the galaxy, Clank asks Ratchet to help him find the famous superhero Captain Qwark, in an effort to stop Drek.
Results	
Back-Translation(Thai)	Fearing Drek will destroy the galaxy, Clank asks Ratchet to help him find the famed superhero captain Qwark to stop Drek.
B_1	$\langle mask \rangle$ that Drek will $\langle mask \rangle$ the $\langle mask \rangle$ Clank $\langle mask \rangle$ Ratchet to $\langle mask \rangle$ him find the $\langle mask \rangle$ Qwark $\langle mask \rangle$ in $\langle mask \rangle$ to $\langle mask \rangle$ Drek $\langle mask \rangle$
BART-Large-CNN	
+ B_1	Drek will work with Clank Ratchet to help him find the Qwark in effort to stop Drek from winning the award.
+ B_2	Clank asks Ratchet to help him find the famous superhero Captain Qwark in an effort to stop Drek.
+ B_3	Clank asks Ratchet to help him find the famed superhero captain Qwark to stop Drek.
+ B_4 +V2000	Fearing Drek, Clank wants Ratchet to help him find Captain Qwark to stop him.
GPT-3	
+ G_1	Captain Qwark is a famous superhero who is rumored to be able to stop Drek from destroying the galaxy. Clank asks Ratchet to help him find Captain Qwark, in order to stop Drek.
+ G_2	Clank asks Ratchet to help him find Captain Qwark in order to stop Drek.

B_0	The other incorporated cities on the Palos Verdes Peninsula include Rancho Palos Verdes, Rolling Hills Estates and Rolling Hills.
Results	
Back-Translation(Thai)	Other cities incorporating on the Palos Verdes peninsula include Rancho Palos Verdes, Rolling Hills Estates, and Rolling Hills.
B_1	the other $\langle mask \rangle$ on the $\langle mask \rangle$ Rancho Palos Verdes $\langle mask \rangle$ Rolling Hills Estates and Rolling Hills $\langle mask \rangle$
BART-Large-CNN	
+ B_1	The other cities on the list include Rancho Palos Verdes Verdes Rolling Hills Estates and Rolling Hills.
+ B_2	The other incorporated cities on the Palos Verdes Peninsula include Rancho Palo Verdes, Rolling Hills Estates and Rolling Hills.
+ B_3	Rancho Palos Verdes, Rolling Hills Estates and Rolling Hills are all on the same peninsula.
+ B_4 +V2000	The other cities on the Palos Verdes Peninsula include Rancho Palo Verdes, Rolling Hills Estates and Rolling Hills.
GPT-3	
+ G_1	The Palos Verdes Peninsula is a peninsula located in Los Angeles County, California. It is south of the Los Angeles neighborhood of West Adams and north of the Palos Verdes Estates neighborhood. The other incorporated cities on the Palos Verdes Peninsula include Rancho Palos Verdes, Rolling Hills Est
+ G_2	The Palos Verdes Peninsula includes the cities of Rancho Palos Verdes, Rolling Hills Estates and Rolling Hills.

B_0	Some towns on the Eyre Highway in the south-east corner of Western Australia, between the South Australian border almost as far as Caiguna, do not follow official Western Australian time.
Results	
Back-Translation(Thai)	Some cities on the Eyre Highway, in the southeastern corner of Western Australia, between the nearly South Australian border to Caiguna, do not meet Western Australia's official time.
B_1	Some $\langle mask \rangle$ on the Eyre Highway in the $\langle mask \rangle$ of Western Australia $\langle mask \rangle$ the South Australian $\langle mask \rangle$ as $\langle mask \rangle$ as Caiguna $\langle mask \rangle$ do not $\langle mask \rangle$ Western Australian time $\langle mask \rangle$
BART-Large-CNN	
+ B_1	Some towns on the Eyre Highway in Western Australia are almost as far as Caiguna.
+ B_2	Some towns on the Eyre Highway in the south-east corner of Western Australia, do not follow official Western Australian time. The Eyre highway runs between the South Australian border almost as far as Caiguna.
+ B_3	Some cities on the Eyre Highway, in the southeastern corner of Western Australia, do not meet Western Australia's official time.
+ B_4 +V2000	Cities on the Eyre Highway in the south-east of Western Australia do not follow official Western Australian time. The highway runs between the South Australian border almost as far as Caiguna.
GPT-3	

+ G_1	Some towns on the Eyre Highway in the south-east corner of Western Australia, between the South Australian border almost as far as Caiguna, use a different time zone than the rest of Western Australia.
+ G_2	Some towns in the south-east corner of Western Australia, including towns on the Eyre Highway between the South Australian border almost as far as Caiguna, do not follow the official Western Australian time.

B_0	At the Voyager 2 images Ophelia appears as an elongated object, the major axis pointing towards Uranus.
Results	
Back-Translation(Thai)	At Voyager 2, Ophelia appears as a long-axis object pointing to Uranus.
B_1	At the Voyager 2 $\langle mask \rangle$ Ophelia $\langle mask \rangle$ as $\langle mask \rangle$ the $\langle mask \rangle$ Uranus $\langle mask \rangle$
BART-Large-CNN	
+ B_1	at the Voyager 2 images Ophelia as the major Uranus.
+ B_2	At the Voyager 2 images Ophelia appears as an elongated object, the major axis pointing towards Uranus.
+ B_3	**At Voyager 2, Ophelia appears as a long-axis object pointing to Uranus.**
+ B_4 +V2000	Ophelia appears as a long-axis Object on Voyager 2. At the Voyager 2 image, Ophelia is seen as an elongation of Uranus.
GPT-3	
+ G_1	Ophelia is an elongated object that is pointing towards Uranus.
+ G_2	Ophelia appears as an elongated object in the Voyager 2 images, with the major axis pointing towards Uranus.

Chapter 6

Conclusions and Future Work

The thesis has developed various knowledge-transferring deep neural network methods that are able to generate simplified text from input text. The proposed methods can effectively simplify the original text and assist people's reading. This chapter summarises the results and contributions of the thesis and highlights future work that attempts to deal with the current limitations.

6.1 Conclusions

Training data for most NLP tasks is not always sufficient and of good quality, which is the main limitation of the text simplification task. All of the methods proposed in this thesis focus on avoiding the reliance on a large amount of high-quality parallel data, which is scarce in the TS task. We use a range of techniques from the application of adversarial networks, to fine-tuning pre-trained models, to prompt-based zero-shot learning methods. All of the methods proposed in the thesis can be trained or predicted on an academic budget, producing good results.

In Chapter 3, we first map a sentence in latent space to find a linear relationship between complex and simple representations. Then we propose an

adversarial unsupervised asymmetric denoising text simplification method with sentence content preservation. We train the discriminator on a set-level paired data, and the embeddings are initialized with the *GloVe* embeddings. It uses an asymmetric denoising technique for sentences with different complexity in the unsupervised adversarial autoencoder architecture. The designed asymmetric denoising technique, which is independently tailored to simple and complex sentences, allows the model to be trained to simulate simplification operations (phrase deletion, reordering, and lexical simplification) and makes the simplification process more interpretable. The proposed sentence similarity loss can help preserve original content while training. Our method achieves the best SARI scores on TurkCorpus in the unsupervised category of comparison methods and presents a good ability to achieve content preservation.

In Chapter 4, we first analyze the most widely used training dataset WikiLarge for TS to demonstrate that WikiLarge has many errors. We propose using *SBERT* sentence similarity to refine the dataset to reduce the errors. The experimental results show that fine-tuning the same model on the refined WikiLarge dataset will generate a better result than on the original WikiLarge. We also propose a continue-fine-tuning strategy that fine-tunes the model that pre-fine-tuned in other related tasks (Summarization in our experiments). It uses the knowledge of the summarization task, which will make the model convergence faster. Moreover, a simple-word-only decoding strategy is also introduced in this chapter, which improves the results significantly. Our method achieves the best SARI score of 41.75 on the ASSET test set, FKGL score at 6.84, and LC score at 7.44 in comparison to other methods trained with the same dataset.

Motivated by recent zero-shot learning and prompting-based methods, we propose a prompting-based method in Chapter 5. It circumvents the data-scarce issue by using only prompting and pre-trained models without model training. It uses prompts to modify the input, transferring the tasks from

TS to sentence denoising. It also obtains state-of-the-art results in various evaluation metrics (43.24 SARI score on ASSET and 5.36 FKGL score). For prompt generation, we corrupt the source text by replacing a complex part with a special token ($\langle mask \rangle$ in our implementation) while keeping the simple words and the named entities. A paraphrase is also concatenated as part of the input that incorporates context, which helps the reconstruction process preserve the original meaning. A pre-trained denoising language model is used to reconstruct the corrupted parts. In this way, the text simplification objective is transferred as a denoising objective. We also propose a simple-word answer searching method to constrain the prediction search space with simple words only during sentence construction. By using the prompting approach, the method proposed in this chapter is easy to implement and achieves good results.

6.2 Current Limitations and Future Work

In this section, we discuss the limitations of text simplification and potential improvements to work on in the future. Various approaches can be further explored to obtain better results and interpretability and applied to the medical domain.

6.2.1 Building New Datasets by Using Sentence Representation

A major limitation of the TS task is the lack of sufficient parallel data. The target simplification transformation for each source is not consistent (See examples in Table 4.2). Recent methods collect new datasets from other text data repositories for the TS model training (Martin et al., 2020e; Omelianchuk et al., 2021). However, the data collection methods concentrate on character-level or word-level operations. We argue that using the pre-trained model sentence representation (such as $SBERT$) can perform better. In future work,

we will use sentence representation to build a new parallel dataset for this task.

6.2.2 Better Automatic Evaluation Metrics

SARI, as a reference-based metric, has been the golden automatic evaluation metric for a long time. It relies on n-grams overlaps among the test sentence input, the output and human-generated references for each test sentence. Even though there are 8 and 10 human-generated references in TurkCorpus and ASSET test sets, the simplification transformations they present are still insufficient (Scialom et al., 2021). In Chapter 5, we propose using the FKGL score in combination with *SBERT* sentence similarity to evaluate the result. However, FKGL is also a problematic metric as it only calculates word length and sentence length. A model-based text simplicity measurement can be studied in the future to replace the FKGL score.

6.2.3 Better Prompts

In Chapter 5, we propose a pre-defined discrete prompt to lead the Seq2Seq model to simplify the source sentence. In Section 5.5.3, we have discussed the limitations. The prompt's modality does not have to be human interpretable for a pre-trained model. Many recent types of research have explored learned continuous prompts, which achieved better results. The continuous prompt works for controlling text simplicity have not been deeply explored, and it is worth studying to push the text simplification field forward.

6.2.4 Application: Medical Text Simplification

Future work could focus on adapting our method for medical text simplification, simplifying complex medical text into a simpler form for non-domain experts. This domain-specific application could use our prompting methodology. Furthermore, it will collect medical text and thesaurus data to create

a medical text simplification system that may help patients rationally assess their condition and treatment. For example, patients may directly see treatment information through online health records. Suppose they can understand their diagnosis, prognosis, and clinical findings. In that case, they will cooperate with the treatment in a more stable manner, which is conducive to treating the patient's disease. There are many PTMs trained on medical text data, which could facilitate the development of our approach in the context of medical applications.

Appendix A

Additional Example Results

The additional result examples are reported in Table A.1. Model Paraphrase stands for the back-translation result with Chinese as the middle language, Full means model training with full WikiLarge dataset, Refine means training with refined Wikilarge, Voc-2000 stands for Refine model with 2000 simple words decoding strategy, Voc-15000 stands for Refine model with 15000 simple words decoding strategy.

TABLE A.1: Example results of models that fine-tuned with different datasets and decoding with different strategies in Chapter 4.

Model	Example Results
Original-1	MacGruber starts asking for simple objects to make something to defuse the bomb, but he is later distracted by something (usually involving his personal life) that makes him run out of time.
Paraphrase	MacGruber started asking for simple objects to make something to defuse the bomb, but then he got distracted by something (usually involving his personal life) that left him out of time.
Full	MacGruber starts to ask for a simple object to make something to defuse the bomb, but he is later disturbed by something. (Usually related to his personal life) which makes him run out of time.
Refine	MacGruber starts asking for simple objects to make something to counter the bomb , but he is later distracted by something (usually involving his personal life) that makes him run out he is time.
Voc-2000	MacGruber is trying to make a bomb, but he is late because of something that makes him run out of time.

Voc-15000	MacGruber starts asking for simple objects to make something to defuse the bomb, but he is later made to run out of time by something (usually involving his personal life).
Original-2	This was absorbed into battalions being formed for XI International Brigade.
Paraphrase	This was absorbed into the battalion being formed for the 11th International Brigade.
Full	This was absorbed into battalions being formed for XI International Brigade.
Refine	This was added to battalions being formed for XI International Brigade.
Voc-2000	This was used by the XI International Brigade to fight in the First World War.
Voc-15000	This was absorbed into battalions being formed for XI International Brigade.
Original-3	It has a capacity of 98,772, making it the largest stadium in Europe, and the eleventh largest in the world.
Paraphrase	With a capacity of 98,772 people, it is the largest stadium in Europe and the 11th largest in the world.
Full	It has a capacity of 98,772 , making it the largest stadium in Europe , and the eleventh largest in the world .
Refine	It can seat 98,772, making it the largest stadium in Europe, and the eleventh largest in the world.
Voc-2000	It is the largest stadium in Europe, and the eleventh largest in the world.
Voc-15000	It has a capacity of 98,772.
Original-4	Jurisdiction draws its substance from public international law, conflict of laws, constitutional law and the powers of the executive and legislative branches of government to allocate resources to best serve the needs of its native society.
Paraphrase	Jurisdiction draws substance from public international law, conflicts of laws, the constitution, and the powers of the executive and legislative branches of government to allocate resources to best serve the needs of their own society.
Full	Jurisdiction draws its substance from public international law, conflict of law, law law and the powers of the executive and legislative branches of government to determine resources to best serve the needs of its native society.

Refine	It draws its substance from public international law, conflict of laws, constitutional law and the powers of the executive and legislative branches of government to allocate resources to best serve the needs of its native society.
Voc-2000	Jurisdiction comes from public international law, conflict of laws, and other laws to give resources to best serve the needs of its native society.
Voc-15000	Jurisdiction draws its substance from public international law, conflict of laws, constitutional law and the powers of the executive and legislative branches of government to use resources to best serve the needs of its native society.
Original-5	Roof tiles dating back to the Tang Dynasty with this symbol have been found west of the ancient city of Chang'an (modern-day Xian).
Paraphrase	West of Chang'an Ancient City (Modern Xi'an) Tang Dynasty roof tiles with this symbol were found.
Full	Roof tiles back to the Tang Dynasty with this symbol have been found west of the ancient city of Chang " (present-day Xian).
Refine	Roof tiles dating back to the Tang Dynasty with this symbol have been found west of the ancient city of Chang 'an.
Voc-2000	It is believed to be from the Tang Dynasty, when the city of Chang 'an was built.
Voc-15000	Roof tiles dating back to the Tang Dynasty with this symbol have been found west of the ancient city of Chang 'an (modern day Xian).
Original-6	She was among the first doctors to object to cigarette smoking around children, and drug use in pregnant women.
Paraphrase	She was one of the first doctors to speak out against smoking in children and drug use in pregnant women.
Full	was among among the first doctors to object to smoking smoking around children , and drug use in pregnant women.
Refine	She was one of the first doctors that said cigarette smoking near children and drug use in pregnant women was not safe.
Voc-2000	She was one of the first doctors to say there was not safe places where children should be used, and she did not want people to use drugs. She also did not allow people to have sex around children.

Voc-15000	She was one of the first doctors to object to cigarette smoking around children, and drug use in pregnant women.
Original-7	Defiantly, she vowed to never renounce the Commune, and dared the judges to sentence her to death.
Paraphrase	She defiantly vowed never to give up the commune and dared a judge to sentence her to death.
Full	Defiantly, she vowed to accept the Commune, and chose the judges to sentence her to death.
Refine	She refused to give up the Commune and dared the judges to sentence her to death.
Voc-2000	Defiantly, she said that she would never give up the Commune, and would not let her be put to death.
Voc-15000	Defiantly, she vowed to never to give up the Commune, and dare the judges to sentence her to death.
Original-8	The term gribble was originally assigned to the wood-boring species, especially the first species described from Norway by Rathke in 1799, Limnoria lignorum.
Paraphrase	The term gribble was originally assigned to wood borer species, especially Limnoria lignorum, the first species described by Rathke from Norway in 1799.
Full	The term gribble was originally assigned to the wood-boring species , especially the first species described from Norway by Rathke in 1799 , Limnoria lignorum.
Refine	The term gribble was first used for the wood-boring species, especially the first species described from Norway by Rathke in 1799, Limnoria lignorum.
Voc-2000	The first species described from Norway by Rathke in 1799, Limnoria Lignorum, is called ”Gribble”.
Voc-15000	The term gribble was originally used for the wood-boring species, especially the first species described from Norway by Rathke in 1799, Limnoria lignorum.

Bibliography

Suha S Al-Thanyyan and Aqil M Azmi. Automated text simplification: A survey. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021.

Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia. EASSE: Easier automatic sentence simplification evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 49–54, Hong Kong, China, November 2019. Association for Computational Linguistics. . URL https://aclanthology.org/D19-3009.

Fernando Alva-Manchego, Louis Martin, Antoine Bordes, Carolina Scarton, Benoît Sagot, and Lucia Specia. ASSET: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4668–4679, Online, July 2020a. Association for Computational Linguistics. . URL https://aclanthology.org/2020.acl-main.424.

Fernando Alva-Manchego, Louis Martin, Antoine Bordes, Carolina Scarton, Benoît Sagot, and Lucia Specia. Asset: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. *arXiv preprint arXiv:2005.00481*, 2020b.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. Data-driven sentence simplification: Survey and benchmark. *Computational Linguistics*, 46(1):135–187, 2020c.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. The (un) suitability of automatic evaluation metrics for text simplification. *Computational Linguistics*, 47(4):861–889, 2021.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*, 2017.

Beata Beigman Klebanov, Kevin Knight, and Daniel Marcu. Text simplification for information-seeking applications. In *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2004, Agia Napa, Cyprus, October 25-29, 2004. Proceedings, Part I*, pages 735–747. Springer, 2004.

Arendse Bernth. Easyenglish: a tool for improving document quality. In *Fifth Conference on Applied Natural Language Processing*, pages 159–165, 1997.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020a. URL https://arxiv.org/abs/2005.14165.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020b.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, 46(3):904–911, 2014.

John Carroll, Guido Minnen, Yvonne Canning, Siobhan Devlin, and John Tait. Practical simplification of english newspaper text to assist aphasic readers. In *Proceedings of the AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*, pages 7–10, 1998.

Raman Chandrasekar and Bangalore Srinivas. Automatic induction of rules for text simplification. *Knowledge-Based Systems*, 10(3):183–190, 1997.

Raman Chandrasekar, Christine Doran, and Srinivas Bangalore. Motivations and methods for text simplification. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*, 1996.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning

phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Kenneth Ward Church, Zeyu Chen, and Yanjun Ma. Emerging trends: A gentle introduction to fine-tuning. *Natural Language Engineering*, 27(6):763–778, 2021.

Alexis Conneau and Guillaume Lample. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7059–7069, 2019.

William Coster and David Kauchak. Learning to simplify sentences using wikipedia. In *Proceedings of the workshop on monolingual text-to-text generation*, pages 1–9, 2011.

Antonia Creswell and Anil Anthony Bharath. Denoising adversarial autoencoders. *IEEE transactions on neural networks and learning systems*, 30(4):968–984, 2018.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online, August 2021a. Association for Computational Linguistics. . URL https://aclanthology.org/2021.findings-acl.161.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. Template-based named entity recognition using bart. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, 2021b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Siobhan Devlin. The use of a psycholinguistic database in the simplification of text for aphasic readers. *Linguistic databases*, 1998.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402, Florence, Italy, July 2019. Association for Computational Linguistics. . URL https://aclanthology.org/P19-1331.

Allyson Ettinger. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48, 2020. . URL https://aclanthology.org/2020.tacl-1.3.

Angela Fan, David Grangier, and Michael Auli. Controllable abstractive summarization. *arXiv preprint arXiv:1711.05217*, 2017.

Thibault Fevry and Jason Phang. Unsupervised sentence compression using denoising auto-encoders. *arXiv preprint arXiv:1809.02669*, 2018.

Rudolph Flesch. A new readability yardstick. *Journal of applied psychology*, 32 (3):221, 1948.

Markus Freitag and Scott Roy. Unsupervised natural language generation with denoising autoencoders. *arXiv preprint arXiv:1804.07899*, 2018.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. Style transfer in text: Exploration and evaluation. *arXiv preprint arXiv:1711.06861*, 2017.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online, August 2021. Association for Computational Linguistics. . URL https://aclanthology.org/2021.acl-long.295.

Goran Glavaš and Sanja Štajner. Simplifying lexical simplification: Do we need simplified corpora? 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 63–68, 2015.

Shuhao Gu, Jinchao Zhang, Fandong Meng, Yang Feng, Wanying Xie, Jie Zhou, and Dong Yu. Token-level adaptive training for neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1035–1046, Online, November 2020. Association for Computational Linguistics. . URL https://aclanthology.org/2020.emnlp-main.76.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. Dynamic multi-level multi-task learning for sentence simplification. *arXiv preprint arXiv:1806.07304*, 2018.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. A deep generative framework for paraphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Thanh-Le Ha, Jan Niehues, and Alexander Waibel. Toward multilingual neural machine translation with universal encoder and decoder. *arXiv preprint arXiv:1611.04798*, 2016.

Wei He, Katayoun Farrahi, and Adam Prugel-Bennett. Text simplification using pre-trained language models without fine-tuning. In *Submitted to The 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*, 2023.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.

Ke Hu, Ying Deng, and Xiaobin Liu. Wordsift: Reading easier by understanding key words. *RELC Journal*, page 00336882221087464, 2022.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. *arXiv preprint arXiv:1703.00955*, 2017.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

Shaojie Jiang, Pengjie Ren, Christof Monz, and Maarten de Rijke. Improving neural response diversity with frequency-aware cross-entropy loss. In *The World Wide Web Conference*, pages 2879–2885, 2019.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020. . URL https://aclanthology.org/2020.tacl-1.28.

David Kauchak. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 1537–1546, 2013.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Controlling output length in neural encoder-decoders. *arXiv preprint arXiv:1609.09552*, 2016.

Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch, 1975.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*, 2017.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

Reno Kriz, Joao Sedoc, Marianna Apidianaki, Carolina Zheng, Gaurav Kumar, Eleni Miltsakaki, and Chris Callison-Burch. Complexity-weighted loss and diverse reranking for sentence simplification. *arXiv preprint arXiv:1904.02767*, 2019.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. Evaluating the factual consistency of abstractive text summarization. *arXiv preprint arXiv:1910.12840*, 2019.

Henry Kucera, Henry Kučera, and Winthrop Nelson Francis. *Computational analysis of present-day American English*. University Press of New England, 1967.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*, 2018.

Dhruv Kumar, Lili Mou, Lukasz Golab, and Olga Vechtomova. Iterative edit-based unsupervised sentence simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7918–7928, Online, July 2020. Association for Computational Linguistics. . URL https://aclanthology.org/2020.acl-main.707.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*, 2017.

Corinne Le Quéré, Robbie M Andrew, Pierre Friedlingstein, Stephen Sitch, Judith Hauck, Julia Pongratz, Penelope A Pickers, Jan Ivar Korsbakken, Glen P Peters, Josep G Canadell, et al. Global carbon budget 2018. *Earth System Science Data (Online)*, 10(4), 2018.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference*

on Empirical Methods in Natural Language Processing, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. . URL https://aclanthology.org/2021.emnlp-main.243.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. . URL https://aclanthology.org/2020.acl-main.703.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online, August 2021. Association for Computational Linguistics. . URL https://aclanthology.org/2021.acl-long.353.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021a.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021b.

Yang Liu, Matt Gardner, and Mirella Lapata. Structured alignment networks for matching sentences. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1554–1564, 2018.

Xinyu Lu, Jipeng Qiang, Yun Li, Yunhao Yuan, and Yi Zhu. An unsupervised method for building sentence simplification corpora in multiple languages. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 227–237, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. . URL https://aclanthology.org/2021.findings-emnlp.22.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Jonathan Mallinson and Mirella Lapata. Controllable sentence simplification: Employing syntactic and lexical constraints. *arXiv preprint arXiv:1910.04387*, 2019.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. Felix: Flexible text editing through tagging and insertion. *arXiv preprint arXiv:2003.10687*, 2020.

Louis Martin, Benoît Sagot, Eric de la Clergerie, and Antoine Bordes. Controllable sentence simplification. *arXiv preprint arXiv:1910.02677*, 2019.

Louis Martin, Éric de la Clergerie, Benoît Sagot, and Antoine Bordes. Controllable sentence simplification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4689–4698, Marseille, France, May 2020a. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.577.

Louis Martin, Éric Villemonte de la Clergerie, Benoît Sagot, and Antoine Bordes. Controllable sentence simplification. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4689–4698, 2020b.

Louis Martin, Angela Fan, Éric de la Clergerie, Antoine Bordes, and Benoît Sagot. Multilingual unsupervised sentence simplification. *CoRR*, abs/2005.00352, 2020c. URL https://arxiv.org/abs/2005.00352.

Louis Martin, Angela Fan, Éric de la Clergerie, Antoine Bordes, and Benoît Sagot. Multilingual unsupervised sentence simplification. *arXiv preprint arXiv:2005.00352*, 2020d.

Louis Martin, Angela Fan, Éric de la Clergerie, Antoine Bordes, and Benoît Sagot. Muss: Multilingual unsupervised sentence simplification by mining paraphrases. *arXiv preprint arXiv:2005.00352*, 2020e.

Clara Meister, Ryan Cotterell, and Tim Vieira. If beam search is the answer, what was the question? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2173–2185, Online, November 2020. Association for Computational Linguistics. . URL https://aclanthology.org/2020.emnlp-main.170.

George A Miller. *WordNet: An electronic lexical database*. MIT press, 1998.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany, August 2016. Association for Computational Linguistics. . URL https://aclanthology.org/K16-1028.

Shashi Narayan and Claire Gardent. Hybrid simplification using deep semantics and machine translation. 2014.

Shashi Narayan and Claire Gardent. Unsupervised sentence simplification using deep semantics. *arXiv preprint arXiv:1507.08452*, 2015.

Gonzalo Navarro. A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, 33(1):31–88, 2001.

Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157, 2016.

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P. Dinu. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 85–91, Vancouver, Canada, July 2017. Association for Computational Linguistics. . URL https://aclanthology.org/P17-2014.

Kostiantyn Omelianchuk, Vipul Raheja, and Oleksandr Skurzhanskyi. Text simplification by tagging. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 11–25, 2021.

Gustavo H Paetzold and Lucia Specia. Unsupervised lexical simplification for non-native speakers. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Eleni Partalidou, Eleftherios Spyromitros-Xioufis, Stavros Doropoulos, Stavros Vologiannidis, and Konstantinos I Diamantaras. Design and implementation of an open source greek pos tagger and entity recognizer using

spacy. In *2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 337–341. IEEE, 2019.

Ellie Pavlick and Chris Callison-Burch. 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, 2016.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China, November 2019a. Association for Computational Linguistics. . URL https://aclanthology.org/D19-1250.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, 2019b.

Ben Poole, Jascha Sohl-Dickstein, and Surya Ganguli. Analyzing noise in autoencoders and deep networks. *arXiv preprint arXiv:1406.1831*, 2014.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia, July 2018. Association for Computational Linguistics. . URL https://aclanthology.org/P18-1080.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26, 2020.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits

of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019a. Association for Computational Linguistics. . URL https://aclanthology.org/D19-1410.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019b.

Timo Schick and Hinrich Schütze. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online, April 2021a. Association for Computational Linguistics. . URL https://aclanthology.org/2021.eacl-main.20.

Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online, June 2021b. Association for Computational Linguistics. . URL https://aclanthology.org/2021.naacl-main.185.

Thomas Scialom, Louis Martin, Jacopo Staiano, Éric Villemonte de la Clergerie, and Benoît Sagot. Rethinking automatic evaluation in sentence simplification. *arXiv preprint arXiv:2104.07560*, 2021. URL https://arxiv.org/abs/2104.07560.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. *arXiv preprint arXiv:1705.09655*, 2017.

Tianxiao Shen, Jonas Mueller, Regina Barzilay, and Tommi Jaakkola. Educating text autoencoders: Latent representation guidance via denoising. *arXiv preprint arXiv:1905.12777*, 2019.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online, November 2020. Association for Computational Linguistics. . URL https://aclanthology.org/2020.emnlp-main.346.

Advaith Siddharthan. Text simplification using typed dependencies: A comparision of the robustness of different generation strategies. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 2–11, 2011.

Advaith Siddharthan. A survey of research on text simplification. *ITL-International Journal of Applied Linguistics*, 165(2):259–298, 2014.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

Sanja Štajner and Sergiu Nisioi. A detailed evaluation of neural sequence-to-sequence models for in-domain and cross-domain text simplification. In *Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018)*, 2018.

Sanja Štajner and Maja Popović. Automated text simplification as a preprocessing step for machine translation into an under-resourced language. 2019.

Elior Sulem, Omri Abend, and Ari Rappoport. Bleu is not suitable for the evaluation of text simplification. *arXiv preprint arXiv:1810.05995*, 2018.

Sai Surya, Abhijit Mishra, Anirban Laha, Parag Jain, and Karthik Sankaranarayanan. Unsupervised neural text simplification. *arXiv preprint arXiv:1810.07931*, 2018.

Sai Surya, Abhijit Mishra, Anirban Laha, Parag Jain, and Karthik Sankaranarayanan. Unsupervised neural text simplification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2058–2068, Florence, Italy, July 2019. Association for Computational Linguistics. . URL https://aclanthology.org/P19-1198.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. 2014.

Julien Tissier, Christophe Gravier, and Amaury Habrard. Dict2vec: Learning word embeddings using lexical dictionaries. 2017.

Shruti Tyagi, Deepti Chopra, Iti Mathur, and Nisheeth Joshi. Classifier based text simplification for improved machine translation. In *2015 International Conference on Advances in Computer Engineering and Applications*, pages 46–50. IEEE, 2015.

Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6):1606–1618, 2007.

Laura Vásquez-Rodríguez, Matthew Shardlow, Piotr Przybyła, and Sophia Ananiadou. Investigating text simplification evaluation. *arXiv preprint arXiv:2107.13662*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

Tu Vu, Baotian Hu, Tsendsuren Munkhdalai, and Hong Yu. Sentence simplification with memory-augmented neural networks. *arXiv preprint arXiv:1804.07445*, 2018.

Haifeng Wang, Jiwei Li, Hua Wu, Eduard Hovy, and Yu Sun. Pre-trained language models and their applications. *Engineering*, 2022a.

Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. What language model architecture and pretraining objective work best for zero-shot generalization? *arXiv preprint arXiv:2204.05832*, 2022b.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. Should you mask 15% in masked language modeling? *arXiv preprint arXiv:2202.08005*, 2022.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Kristian Woodsend and Mirella Lapata. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 409–420, 2011.

Sander Wubben, EJ Krahmer, and APJ van den Bosch. Sentence simplification by monolingual machine translation. 2012.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297, 2015.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415, 2016a. . URL https://aclanthology.org/Q16-1029.

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

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. Unsupervised text style transfer using language models as discriminators. *arXiv preprint arXiv:1805.11749*, 2018.

Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China, November 2019. Association for Computational Linguistics. . URL https://aclanthology.org/D19-1404.

Farooq Zaman, Matthew Shardlow, Saeed-Ul Hassan, Naif Radi Aljohani, and Raheel Nawaz. Htss: A novel hybrid text summarisation and simplification architecture. *Information Processing & Management*, 57(6):102351, 2020.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

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

Yuhao Zhang, Daisy Yi Ding, Tianpei Qian, Christopher D Manning, and Curtis P Langlotz. Learning to summarize radiology findings. *arXiv preprint arXiv:1809.04698*, 2018.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. Adversarially regularized autoencoders. In *International Conference on Machine Learning*, pages 5902–5911, 2018a.

Sanqiang Zhao, Rui Meng, Daqing He, Saptono Andi, and Parmanto Bambang. Integrating transformer and paraphrase rules for sentence simplification. *arXiv preprint arXiv:1810.11193*, 2018b.

Yanbin Zhao, Lu Chen, Zhi Chen, and Kai Yu. Semi-supervised text simplification with back-translation and asymmetric denoising autoencoders. In *AAAI*, pages 9668–9675, 2020.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1353–1361, 2010.