Iterative Edit-based Unsupervised Sentence Simplification

by

Dhruv Kumar

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2020

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

We present a new *iterative* approach towards unsupervised edit-based sentence simplification. Our approach is guided by a scoring function to select simplified sentences generated after iteratively performing word and phrase-level edits on the complex sentence. The scoring function measures different aspects of simplification: fluency, simplicity, and preservation of meaning. As a result, unlike past approaches, our method is controllable and interpretable and does not require a parallel training set since it is unsupervised. At the same time, using the Newsela and WikiLarge datasets, we experimentally show that our solution is nearly as effective as state-of-the-art supervised approaches.

Acknowledgements

I thank my supervisor, Professor Lukasz Golab, for his support, patience and guidance throughout my research. I thank Professor Srinivasan Keshav for his help and advice. I also thank Professors Robin Cohen and Olga Vechtomova, who guided me in my research and served as readers for this thesis. I thank Professor Lili Mou for the helpful discussions. Finally, I thank all the people who helped to make this thesis possible.

Dedication

I dedicate this thesis to my family.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Simplification is the task of rewriting text to make it easier to read while preserving its central meaning. Simplification can make complex and specialized text (e.g., law and healthcare documents, research papers, etc.) more accessible to a broader audience. It can even benefit people with autism [8], dyslexia [40], and low-literacy skills [51], reducing inequality. It can also serve as a preprocessing step to improve parsers [5] and summarization systems [15].

Sentence simplification models focus on simplifying complex text at a sentence level. They rewrite a complex sentence by removing peripheral information, substituting complex words with simpler synonyms, and splitting the sentence into several shorter sentences, among others. Text simplification may also incorporate strategies used in text summarization. For example, reducing the content of the given text is the main focus of text summarization. However, a summary though always shorter than the original text, may not be easier to read and understand. Margarido et al. [23] performed a human study to understand if the generated summary (by only reducing content) was easier to understand that the original text. They concluded that the summary was harder to understand for people with lower literacy levels. Finally, a simplified sentence may also be longer than the complex sentence since a part of the sentence may be elaborated.

The following examples taken from the Newsela corpus [54] highlight different strategies used by professional editors to simplify text for children.

Consider the first example:

1 (a) All 16 who died were sherpas, the catch-all term for local mountain guides, porters and camp staff, a trade dominated by the ethnic Sherpa people who populate the Himalayan highlands.

(b) All of the victims were sherpas.

We observe that the peripheral information that is not related to the primary assertion is removed. This is known as content reduction.

2 (a) Han Fuling obeyed Mao Zedong's orders.

(b) Han Fuling followed Mao Zedong's rules.

In this example, the words *obeyed* and *orders* are substituted by *followed* and *rules* respectively. This is an instance of lexical simplification, where complex words are replaced with their simpler synonyms. Replacing words with simpler synonyms can sometimes alter the meaning of the sentence. For example, in this case, *orders* and *rules* do not convey exactly the same meaning.

3 (a) Goddard said his team is designing projects in India that are sensitive to the local culture.

(b) Goddard said his team respects the local culture.

Here, the latter half of the sentence is paraphrased in addition to content reduction. Paraphrasing can also alter the meaning of the sentence. For example, the meaning of the phrase *team is designing projects that are sensitive to the local culture* is not entirely same as that of the phrase *team respects the local culture*. However, paraphrasing helps change the semantic and syntactic formation of the sentence and thus simplifies it.

4 (a) For mining, there's the International Seabed Authority.

(b) The International Seabed Authority is for mining.

We observe that the sentence structure is modified as the phrase *for mining* is moved to the end of the sentence. This operation is referred to as 'syntactic reordering' as it changes the sentence syntax making it easier to read.

5 (a) Weariness frays their voices, but they're still on the freedom highway.

(b) Their voices sound tired.

This is an instance of extreme paraphrasing where the entire complex sentence is rewritten in simpler words. This is one of the more challenging instance of simplification.

Recent efforts in sentence simplification have been influenced by the success of machine translation. Thus, sentence simplification is often treated as a monolingual machine translation task where a complex sentence is translated to a simple one. Such simplification systems are typically trained in a supervised way by either phrase-based machine translation (PBMT) [53, 31, 55] or neural machine translation (NMT) [57, 9, 16]. Recently, sequence-to-sequence (Seq2Seq)-based NMT systems are shown to be more successful and serve as the state of the art.

However, supervised Seq2Seq models have three shortcomings.

1. First, they are hard to interpret and provide little insight into the simplification operations. Interpretable models are important as they help understand the operations that generate better simplifications.

2. Second, they provide little control or adaptability to the different aspects of simplification (e.g., lexical vs. syntactical simplification). Controllability is critical when simplification is required for a diverse set of target users, each having different requirements. For example, it may be more beneficial to focus on content reduction when simplifying text for children. [1]

3. Third, they require a large number of complex-simple aligned sentence pairs, which in turn requires considerable human effort to obtain.

In previous work, researchers have addressed some of the above issues. For example, Alva-Manchego et al. [1] and Dong et al. [7] explicitly model simplification edit operators such as word insertion and deletion (edit-based models explicitly define the editing operators.). Explicitly defining the edit operators makes these models more controllable and interpretable than standard Seq2Seq models. However, they still require large volumes of aligned data to learn these operations.

Surya et al. [47] recently proposed an unsupervised neural text simplification approach based on the paradigm of style transfer [2]. Thus, they do not require aligned complex-simple sentence pairs. However, their model is hard to interpret and control, like other neural

[1] Current simplification models do not focus on controllability since they are evaluated on their performance on the available datasets and not on the diversity of target users they serve.

[2] Converting a sentence with some attribute (for example, positive sentiment) to a sentence with a different attribute (negative sentiment) while preserving the semantic information.

network-based models. Narayan and Gardent [32] attempted to address both issues using a pipeline of lexical substitution, sentence splitting, and word/phrase deletion. However, they have separate modules for the different editing operations and execute them in a fixed order.

Motivated by the shortcomings of existing work, our goal is to develop a model that is interpretable, controllable, does not require a large amount of training data, and produces high-quality simplifications. To do so, we propose an iterative, edit-based unsupervised sentence simplification approach.

We first design a scoring function that measures the quality of a candidate sentence based on the key characteristics of the simplification task, namely, fluency, simplicity, and meaning preservation. Then, we generate simplified candidate sentences by iteratively editing the given complex sentence using four simplification operations (lexical simplification, phrase extraction, deletion and reordering). Our model seeks the best-simplified candidate sentence according to the scoring function. Compared with Narayan and Gardent [32], the order of our simplification operations is not fixed and is decided by the model.

Figure 1.1 shows an example, in which our model first chooses to delete a sentence fragment, followed by reordering the remaining fragments and finally replacing a complex word with a simpler synonym.

We evaluate our approach on the Newsela Corpus[3], a collection of 1,840 news articles written by professional editors for children at five reading levels [54] and Wikilarge, the largest simplification dataset [57]. Experiments using automatic metrics and human readers show that our model outperforms the previous unsupervised methods and performs competitively to state-of-the-art supervised approaches. We also demonstrate the interpretability and controllability of our approach, even without parallel training data and examine the impact of our design choices on the scoring function.

To summarize, we make the following contributions:

1. We design an unsupervised (aligned complex-simple sentence pairs are not required) edit-based model that performs content reduction, structural, and lexical simplification.

2. We evaluate our model on two large datasets and show that it performs better than previous unsupervised models and competitively to the state-of-the-art supervised models.

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

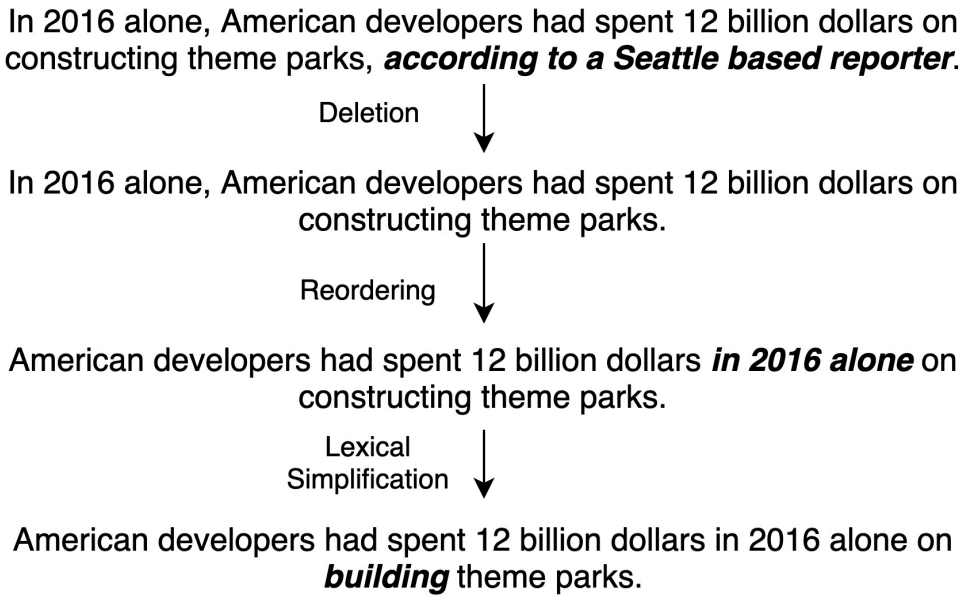

Figure 1.1: An example of three edit operations on a given sentence.

3. We do an extensive evaluation of our model and carry out ablation studies. We also show that our model is interpretable and can control different aspects of simplification.

4. We provide an open-source implementation of our model and release our trained models and system outputs for reproducibility. Our code is available at https://github.com/ddhruvkr/Edit-Unsup-TS.

The remainder of this dissertation is structured as follows. Chapter 2 provides a brief overview of the machine learning techniques relevant to this work. Chapter 3 highlights prior work on sentence simplification. Chapter 4 details our proposed model and the intuitions for its design. Chapter 5 presents the results on the two datasets using both automatic and human evaluation. Finally, we conclude in Chapter 6 providing directions for future work. The work and the text in this thesis is based on our work [17] published at the 58th Annual Meeting of the Association of Computational Linguistics.

Chapter 2

Preliminaries

2.1 Word Embeddings

Word embeddings represent words in a high-dimensional space, mapping them to real-valued vectors. These representations in a continuous vector space are useful for machine learning models since words having similar meaning have a similar representation. There have been several methods proposed in the literature to learn these representations [29]. Methods such as Word2Vec [27] and GloVe [38] are used extensively in Natural Language Processing (NLP).

Mikilov et al. [27] showed that the embeddings capture semantic information such as relationships between words. For example, arithmetic operations such as $\vec{king} - \vec{man} + \vec{woman} \approx \vec{queen}$ are possible. This means when the embedding vector of man is subtracted from that of king, followed by the addition of the embedding vector of woman, it is approximately equal to the embedding vector of queen.

For any two word representations $\vec{a}$ and $\vec{b}$, we can measure their semantic similarity using the cosine-similarity distance metric.

$$cos\theta = \frac{\vec{a} \cdot \vec{b}}{|a| \cdot |b|} \tag{2.1}$$

The $|\cdot|$ represents the L2-norm corresponding to the vector. A high value means that the vector representations of the words are close in the high dimensional space and thus the words are semantically similar.

2.2 Generative Models

There are two main approaches in statistical modelling - discriminative and generative. Given observable data X and corresponding target variable Y, a discriminative model learns the conditional probability of the target Y, given an observation x, i.e. $P(Y|X = x)$. In contrast, a generative model learns the joint probability distribution, i.e., $P(X, Y)$. Thus, a generative model is used to approximate the underlying distribution of the observable data. For a discriminative model, the main task is predicting the label of a given data point, whereas a generative model is used to estimate the density of a given point in the data or to sample from the data. In NLP, generative models can thus be used to assign a probability to a given sequence of text and generate text.

2.2.1 Language Model

A language model is a generative model that computes the probability distribution over a sequence of text. In NLP, a sequence refers to words or characters in a document. It does so in a probabilistic manner by learning to predict the next word in the sequence given the words preceding it. $P_{\text{LM}}(s)$ denotes the language model probability for a sequence s containing words $w_1, w_2, \ldots, w_m$.

$$P_{\text{LM}}(s) = P_{\text{LM}}(w_1, \ldots, w_m) = \prod_{i=1}^{m} P(w_i \mid w_1, \ldots, w_{i-1}) \tag{2.2}$$

The number of preceding words that the model considers can be adjusted depending on the complexity of the model and the amount of training data. Earlier, less powerful n-gram language models considered few previous words (e.g. 3-5). However, recently, neural language models have outperformed n-gram language models since they can capture more context while calculating the probability distribution for the next word. Since a language model learns to predict the next word, it is also used to generate text.

2.2.2 Sequence to Sequence (Seq2Seq) Model

Plain Seq2Seq

Sequence to Sequence (Seq2Seq) models are a family of models that take a sequence as input and generate another sequence as output. They were proposed by Sutskever et al.

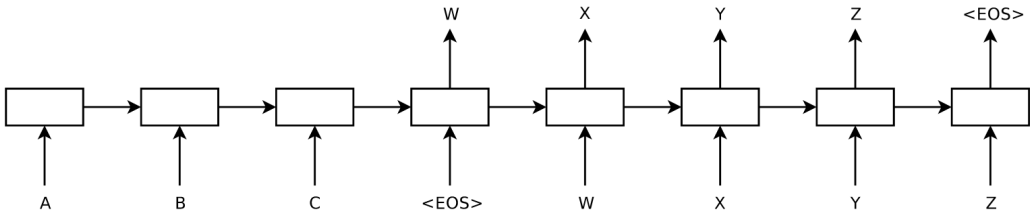

Figure 2.1: Sequence to Sequence model where the encoder takes in a sequence ABC and the decoder generates the sequence WXYZ. Image taken from Sutskever et al. [48].

[48] for machine translation. However, they have been successful in a variety of generation tasks such as question answering, conversational agents, paraphrasing, etc. They consist of two neural networks, an encoder that takes a sequence token by token and generates a latent representation vector for it. The other, called the decoder, uses the latent vector representation of the input sequence and generates an output sequence token by token (thus, these models are also called autoregressive models). An embedding vector represents each of the token in the encoder and decoder. The encoder and decoder have recurrent neural networks (RNNs) as their building blocks. The encoder-decoder architecture is shown in Figure 2.1.

Let $x = (x_1, x_2, \ldots, x_n)$ be the tokens (represented by embeddings) of a source sequence and $y = (y_1, y_2, \ldots, y_m)$ be the tokens of the target sequence. The source and target sequences contain n and m tokens respectively. At a time step i, the encoder takes as input the source sequence token x_i and the hidden representation vector (h_s^{i-1}) from the previous time-step and generates the hidden representation vector h_s^i. The final hidden representation vector generated at the end of the encoding process is h_s^n (since n is the number of tokens in the sequence). For time-step i it is represented below.

$$h_s^i = RNN(h_s^{i-1}, x_i) \tag{2.3}$$

Once the entire encoding process is finished, the decoding process starts. Again, for a time-step i, during generation, the decoder generates a hidden representation of a token (h_t^i) using the hidden representation vector (h_t^{i-1}) and the generated output token (y'_{i-1}) from the previous time-step. Typically, during training, the ground truth (target output) for the time step $i-1$ (represented as y_{i-1}) is given as input instead of the generated output token of the previous time-step, as shown in the equation below (known as teacher-forcing).

$$h_t^i = RNN(h_t^{i-1}, y_{i-1}) \tag{2.4}$$

For the first time-step of the decoder, its previous hidden vector is set equal to the final hidden vector from the encoder, i.e. $h_t^0 = h_s^n$. This is also known as hidden state initialization. The hidden representation vector generated by the decoder is used to predict the actual token at time-step i, as shown below. The weights in W_{out} are learned during training.

$$p(y_i') = Softmax(W_{out} h_t^i)) \tag{2.5}$$

Attention Mechanism

The idea behind attention is that different words in a sentence could have different relative importance. In a Seq2Seq model, attention helps the decoder to use the hidden representations of all the tokens in the input sequence (instead of just the final hidden representation from the encoder) while generating the hidden representations for each time step. Additionally, the decoder can focus more on the hidden representations of the input tokens that are more relevant at a particular time-step. The decoding process with the attention mechanism is shown in Figure 2.2. Luong et al. [21] examined different ways in which the attention mechanism could be applied to Seq2Seq models.

The first step is similar as in the case of plain Seq2Seq model. The hidden vector h_t^i of a token at time-step i is calculated from the decoder. Then, the attention vector (a_t^i) is calculated at every time-step by multiplying the hidden representation vector from the decoder (h_t^i) with all the hidden vectors (for all time-steps and not just the final) from the encoder $\bar{h_s}$.

$$a_t^i = Softmax(h_t^i \cdot \bar{h_s}) \tag{2.6}$$

The attention vector (a_t^i) for a time-step is used to calculate the context vector (c_t^i) for that time-step by taking its dot product with the hidden vectors of the encoder $(\bar{h_s})$. Thus, the attention vector allows the decoder to consider all the hidden vectors of the encoder and give larger weights to more relevant tokens.

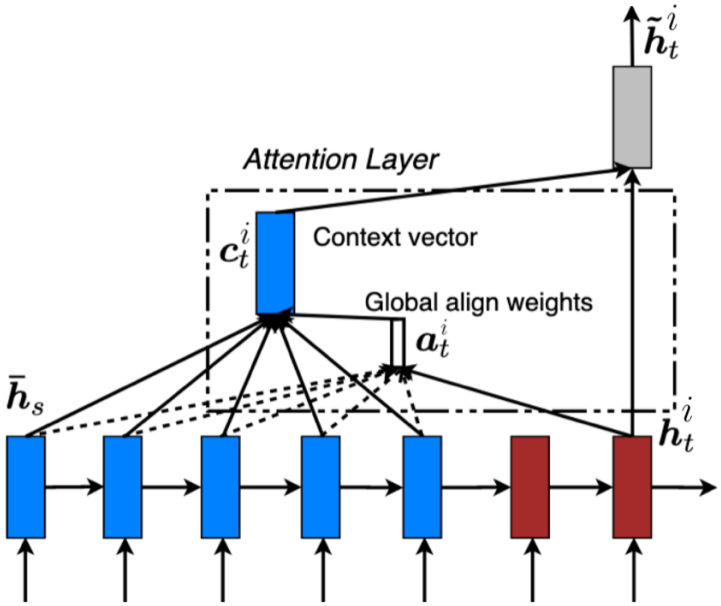

Figure 2.2: Attention mechanism in Seq2Seq model. Image taken from Luong et al. [21]

$$c_t^i = a_t^i \bar{h_s} \tag{2.7}$$

The decoder then calculates a new hidden state vector $\tilde{h_t^i}$ for each time-step by using its original hidden state vector h_t^i and the context vector (c_t^i) for that time-step.

$$\tilde{h_t^i} = tanh(W_c[h_t^i; c_t^i])) \tag{2.8}$$

This new hidden state ($\tilde{h_t^i}$) is then used instead of the original hidden state (h_t^i) to generate tokens as shown below. As mentioned above, W_{out} is a trainable weight matrix.

$$p(y_i') = Softmax(W_{out}\tilde{h_t^i})) \tag{2.9}$$

Chapter 3

Background and Related Work

Text simplification has a long history of research [4, 5]. Initial work on sentence simplification used handcrafted rules to capture syntactic simplification, e.g., to model active/passive transformations [43, 44]. However, rule-based systems work well only when a narrow set of linguistic characteristics are desired [45]. Consequently, recent approaches have taken a more data-driven approach towards modeling text simplification systems. They can be broadly classified into four categories. They tackle simplification at a sentence level.

3.1 Phrase-based Machine Translation

Recent approaches view simplification as a monolingual text-to-text generation task borrowing ideas from machine translation (MT). Zhu et al. [60] extended the syntax-based translation model proposed in Yamada and Knight [56] to perform simplification-specific rewrite operations (split, reorder, delete and substitution). Wubben et al. [53] proposed a two-stage approach: initially, a standard phrase-based machine translation (PBMT) model is trained on complex-simple sentence pairs. Finally, during inference, the k-best outputs of the PBMT model are reranked according to their dissimilarity to the (complex) input sentence. Woodsend and Lapata [52] used quasi-synchronous grammar to formulate a simplification framework and used integer linear programming to score the candidate simplifications. They additionally used the edit history of Simple Wikipedia [60]. Bingel and Søgaard [2] designed a tree-to-string model that operated directly on the dependency trees. The hybrid model by Narayan and Gardent [31] also operated in two phases. Initially, a probabilistic model performed sentence splitting and deletion operations over discourse representation structures (DRS) [12]. Finally, it further simplified the resulting sentences

by using a model similar to Wubben et al. [53]. Xu et al. [55] trained a syntax-based machine translation model using simplification-specific objective functions and features to encourage simpler output.

3.2 Neural Machine Translation

Nisioi et al. [33] were the first to use neural machine translation (NMT) based Seq2Seq models [48] for text simplification. Zhang and Lapata [57] used reinforcement learning methods to optimize a reward based on simplicity, fluency, and relevancy. Vu et al. [50] incorporated memory-augmented neural networks [30] for sentence simplification. Zhao et al. [58] integrated the transformer architecture [49] and paraphrasing rules from Simple PPDB [37] to guide the simplification learning; Sulem et al. [46] combined NMT models with sentence splitting modules for sentence simplification. Guo et al. [9] showed that simplification benefits from multi-task learning with paraphrase and entailment generation. Kriz et al. [16] produced diverse simplifications by generating and re-ranking candidates by fluency, adequacy, and simplicity. Finally, Martin et al. [24] enhanced the transformer architecture with conditioning parameters such as length, lexical and syntactic complexity.

3.3 Edit-based Sentence Simplification

Recently, edit-based techniques have been developed for text simplification. These approaches modify the complex sentences by performing edit operations on each word. Therefore, they are more interpretable. Alva-Manchego et al. [1] first devised a method to automatically identify three simplification operators (copy, replace and delete) in the given parallel simplification corpus. They then used these labels, in addition to the parallel corpus, to train a sequence-labelling model that predicted the simplification operators for each word in a complex sentence. Dong et al. [7] employed a similar approach but in an end-to-end trainable manner referred to as the neural programmer-interpreter model. The edit labels are predicted by the programmer and executed by the interpreter. Figure 3.1 highlights the working of their programmer-interpreter model. However, these approaches are supervised and require large volume of parallel training data; also, their edits are only at the word level. By contrast, our method works at both word and phrase levels in an unsupervised manner.

WikiLarge	
Source	in 2005 , meissner became the second american woman to land the triple axel jump in national competition .
Output	meissner was the second american woman to land the triple axel jump .
Program	DEL DEL DEL KEEP ADD(was) DEL KEEP KEEP KEEP KEEP KEEP KEEP KEEP KEEP KEEP KEEP DEL DEL DEL KEEP
Reference	she is the second american woman and the sixth woman worldwide to do a triple axel jump .
WikiSmall	
Source	theodoros " thodoris " zagorakis -lrb- , born october 27 , 1971 in lyd -lrb- a village near the city of kavala -rrb- , is a retired greek footballer and was the captain of the greece national football team that won the 2004 uefa european football championship .
Output	zagorakis -lrb- born october 27 , 1971 is a former greek football player .
Program	DEL DEL DEL DEL KEEP KEEP DEL KEEP KEEP KEEP KEEP KEEP DEL DEL DEL DEL DEL DEL DEL DEL DEL DEL DEL DEL KEEP KEEP ADD(former) DEL KEEP ADD(football) ADD(player) DEL DEL ... DEL KEEP
Reference	theodoros zagorakis -lrb- born 27 october , 1971 -rrb- is a former football player .
Newsela	
Source	schools and parent groups try to help reduce costs for low-income students who demonstrate a desire to play sports , she said .
Output	schools and parent groups try to help pay for low-income students .
Program	KEEP KEEP KEEP KEEP KEEP KEEP KEEP ADD(pay) DEL DEL KEEP KEEP KEEP DEL DEL DEL DEL DEL DEL DEL DEL DEL DEL KEEP
Reference	clark said that schools do sometimes lower fees for students who do n't have enough money .

Figure 3.1: Edit-based supervised sentence simplification. The trained model first predicts the sequence of edit operations and then executes these operations on the complex sentence to produce the simplified sentence. Image taken from Dong et al.[7].

3.4 Unsupervised Sentence Simplification

There has been little work on unsupervised sentence simplification. Narayan and Gardent [32] built a pipeline-based unsupervised framework. They first learn lexical simplification rules from the simplification corpus in an unsupervised way following the method proposed by Biran et al. [3]. They then perform sentence splitting using a probabilistic model trained on the discourse representation structures (DRS) [12] of Simple Wikipedia sentences and a language model trained on the Simple Wikipedia corpus. Finally, they delete peripheral phrases. They treat phrase deletion as an optimization problem and solve it using integer linear programming. However, each of these operations are handled by different algorithms which makes adding new operations cumbersome. Additionally, these operations are executed in a fixed order. Surya et al. [47] instead proposed an unsupervised neural approach taking inspiration from research on unsupervised machine translation and style transfer. They used an encoder-decoder architecture and added adversarial and denoising auxiliary losses to simultaneously perform content reduction and lexical simplification. However, their neural network-based model is hard to interpret and control.

3.5 Interpretable Unsupervised Methods for Natural Language Generation

Unsupervised edit-based approaches have recently been explored for natural language generation tasks, such as style transfer, paraphrasing, and sentence error correction. Li et al. [19] proposed an edit-based model for style transfer without parallel supervision. They replaced style-specific phrases with those in the target style, which are retrieved from the training corpus. This easy-to-train approach outperformed prior adversarial methods. However, effective sentence simplification requires more than phrase substitution. Miao et al. [26] used Metropolis–Hastings sampling for constrained sentence generation. In this thesis, we model text generation as a search algorithm and design search objective and search actions specifically for text simplification. Additionally, they operate directly on words, whereas we operate on sentence sub-phrases. Concurrent work further shows the success of search-based unsupervised text generation for paraphrasing [20] and summarization [42].

3.6 Evaluation Metrics

We now introduce the relevant and prominently used metrics used to evaluate text simplification models.

BLEU

BLEU (bilingual evaluation understudy) [34] was initially designed to evaluate the quality of the system output (generated sentence) for machine translation but has since been used for other text generation tasks. It estimates the similarity between the system output and the reference sentences using a modified n-gram precision metric. Thus, it measures the n-gram overlap between the system output and the reference sentences. For a given candidate sentence y, system output $\hat{y}$ and its references r, modified n-gram precision (for a specific n-gram) is represented as shown below.

$$p_{ngram} = \frac{\sum_{ngrams \in \hat{y}} count_{clip}(ngram)}{\sum_{ngrams \in \hat{y}} count(ngram)} \tag{3.1}$$

$count_{clip}(ngram)$ represents the minimum number of times an n-gram (from the generated sentence) appears in either the generated or the reference sentence, whereas $count(n-gram)$ represents the number of times an n-gram appears in the generated sentence. The precision scores of all the n-grams in the generated sentence are combined to get a final score.

SARI

Xu et al. [55] designed SARI to compare **s**ystem output **a**gainst **r**eferences and against the **i**nput sentence. More specifically, given an input (complex) sentence, SARI measures the relevancy of words (represented as n-grams) that are added, deleted and kept by the system output by comparing it with the words added, deleted and kept in the reference sentence. Unlike previous metrics, that only compare the system output with the reference sentence, SARI uses the reference as well as the input sentence. This is represented through a Venn diagram shown in Figure 3.2. The final score is computed by taking the F1 score of the add and keep operations and the precision of the delete operation as shown below.

$$SARI = \frac{1}{3}(F_{add} + F_{keep} + P_{del}) \tag{3.2}$$

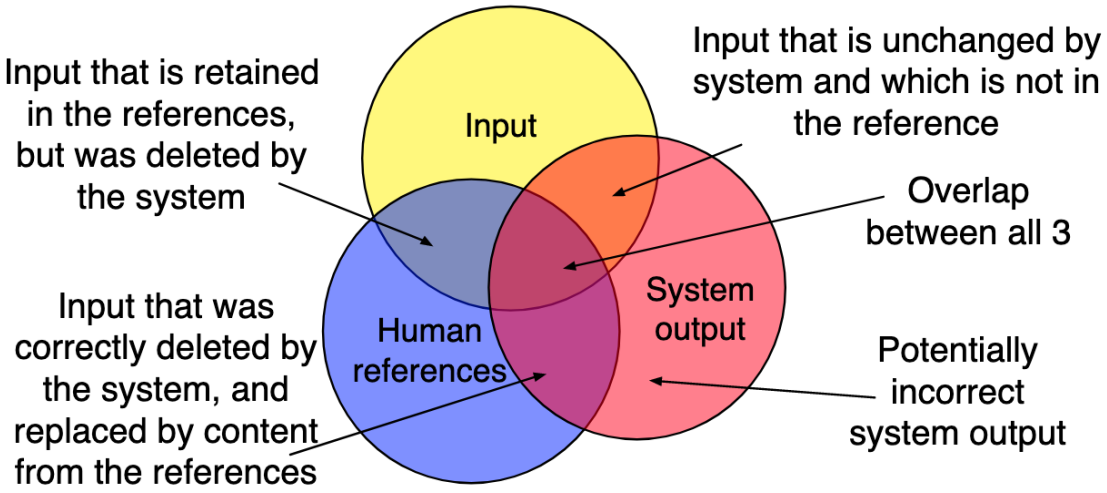

Figure 3.2: An example of three edit operations on a given sentence taken from Xu et al. [55].

FKGL

The Flesch–Kincaid grade level (FKGL) [14] measures the reading difficulty level of a passage. It uses the word and sentence length and the word complexity as its core measures. It has been used extensively in the education sector by teachers, parents, librarians, and others to judge the readability of various books and texts. It also reflects the number of years of education generally required to understand a piece of text. Thus, it is used in the simplification literature to evaluate the performance of models. The grade level is calculated with the following formula:

$$FKGL = 0.39\left(\frac{\text{total words}}{\text{total sentences}}\right) + 11.8\left(\frac{\text{total syllables}}{\text{total words}}\right) - 15.59 \tag{3.3}$$

FRE

Flesch reading ease (FRE) [14] also measures the reading ease of a text using the same core components such as word and sentence length and the word complexity. However, it weighs them differently. The two scores inversely correlate, i.e. a text with a lower score on the FKGL metric will have a higher score on the FRE metric.

$$FRE = 206.835 - 1.015\left(\frac{\text{total words}}{\text{total sentences}}\right) - 84.6\left(\frac{\text{total syllables}}{\text{total words}}\right) \tag{3.4}$$

Chapter 4

Iterative unsupervised text simplification

In this section, we first provide an overview of our approach, followed by a detailed description of each component, namely, the scoring function, the edit operations, and the stopping criteria.

4.1 Overview

We first define a scoring function as our search objective. It allows us to impose both hard and soft constraints, balancing the fluency, simplicity, and adequacy of candidate simplified sentences (Section 4.2).

Our approach iteratively generates multiple candidate sentences by performing a sequence of lexical and syntactic operations. It starts from the input sentence; in each iteration, it performs phrase and word edits to generate simplified candidate sentences (Section 4.3). [1]

Then, a candidate sentence is selected according to a certain criteria. The selected sentence acts as the source sentence in the next iteration and the process is repeated until none of the candidates improve the score of the source sentence by a threshold value. The last candidate is returned as the simplified sentence (Section 4.4).

[1] Figure 1.1 presents an example showing four iterations.

4.2 Scoring Function

Our scoring function is the product of several individual scores that evaluate various aspects of a candidate simplified sentence. This is also known as the product-of-experts model [10].

SLOR score from a syntax-aware language model (f_{eslor})

This measures the *language fluency* and *structural simplicity* of a candidate sentence. A probabilistic language model (LM) is often used as an estimate of sentence fluency [26]. In our work, we make two important modifications to a vanilla language model.

First, we replace an LM's estimated sentence probability with the syntactic log-odds ratio (SLOR) [36], to better measure fluency and human acceptability. According to Lau et al. [18], SLOR shows the best correlation to human acceptability of a sentence, among many sentence probability-based scoring functions. SLOR was also shown to be effective in unsupervised text compression [13].

Given a trained language model (LM) and a sentence s, SLOR is defined as

$$\text{SLOR}(s) = \frac{1}{|s|}(\ln(P_{\text{LM}}(s)) - \ln(P_{\text{U}}(s))) \tag{4.1}$$

where P_{LM} is the sentence probability given by the language model, $P_{\text{U}}(s) = \prod_{w \in s} P(w)$ is the product of the unigram probability of a word w in the sentence, and $|s|$ is the sentence length.

SLOR essentially penalizes a plain LM's probability by unigram likelihood and the length. It ensures that the fluency score of a sentence is not penalized by the presence of rare words. Consider two sentences,

1. I went to England on a vacation.
2. I went to Senegal on a vacation.

Even though both sentences are equally fluent, a standard LM will give a higher score to the former, since the word "England" is more likely to occur than "Senegal."

In simplification, SLOR is preferred for preserving rare words such as named entities.[2]

[2]Note that we do not use SLOR to evaluate lexicon simplicity, which will later be evaluated by the Flesch reading ease (FRE) score. The SLOR score, in fact, preserves rare words, so that we can better design dictionary-based word substitution for lexical simplification (Section 4.3).

Second, we use a syntax-aware LM, i.e., in addition to words, we use part-of-speech and dependency tags as inputs to the LM [59]. For a word w_i, the input to a syntax-aware LM, x_i, can be represented as:

$$x_i = [e(w_i); p(w_i); d(w_i)] \tag{4.2}$$

where $e(w_i)$, $p(w_i)$ and $d(w_i)$ represent the word, part-of-speech (POS) tag and dependency tag embeddings, respectively. The output to be predicted by the LM is still the next word, as in a plain LM. Note that our LM is trained on simple sentences. Thus, the syntax-aware LM prefers a syntactically simple sentence. It also helps to identify sentences that are structurally ungrammatical.

Cosine Similarity (f_{cos})

Cosine similarity is an important measure of meaning preservation. We compute the cosine value between sentence embeddings of the original complex sentence (c) and the generated candidate sentence (s), where our sentence embeddings are calculated as the idf weighted average of individual word embeddings. Our sentence similarity measure acts as a hard filter, i.e., $f_{\text{cos}}(s) = 1$ if $\cos(\boldsymbol{c}, \boldsymbol{s}) > \tau$, or $f_{\text{cos}}(s) = 0$ otherwise, for some threshold τ.

Entity Score (f_{entity})

Named entities help identify the key information of a sentence and therefore are also useful in measuring meaning preservation. Thus, we count the number of entities in the sentence as part of the scoring function, where entities are detected by a third-party tagger.

Length (f_{len})

This score is proportional to the inverse of the sentence length. It forces the model to generate shorter and simpler sentences. In addition to using length as a soft constraint, we also set a hard constraint and reject sentences shorter than a specified length ($\leq$6 tokens) to prevent over-shortening.

FRE (f_{fre})

The Flesch Reading Ease (FRE) score [14] measures the ease of readability in text. It is based on text features such as the average sentence length and the average number of syllables per word. A higher scores indicate that the text is *simpler* to read.

We compute the overall scoring function as the product of individual scores.

$$\begin{aligned} f(s) = f_{\text{eslor}}(s)^{\alpha} \cdot f_{\text{fre}}(s)^{\beta} \cdot f_{\text{len}}(s)^{\gamma} \\ \cdot f_{\text{entity}}(s)^{\delta} \cdot f_{\text{cos}}(s) \end{aligned} \tag{4.3}$$

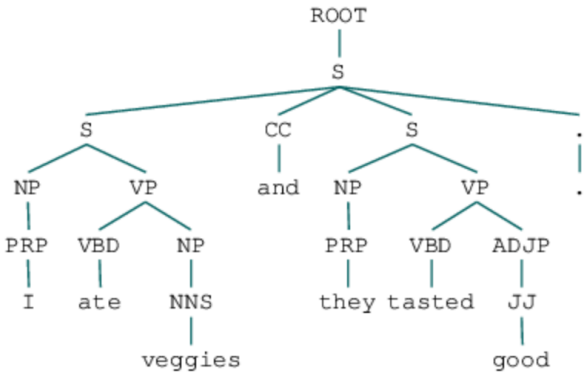

Figure 4.1: Constituency parse tree is used for detecting phrases.

where the weights α, β, γ, and δ balance the relative importance of the different scores. Recall that the cosine similarity measure does not require a weight since it is a hard indicator function.

In Section 5.4.2, we will experimentally show that the weights defined for different scores affect different characteristics of simplification and thus provide more adaptability and controllability.

4.3 Generating Candidate Sentences

We generate candidate sentences by editing words and phrases. We use a third-party parser to obtain the constituency tree of a source sentence. Each clause- and phrase-level constituent (e.g., S, VP, and NP) is considered as a phrase. Since a constituent can occur at any depth in the parse tree, we can deal with both long and short phrases at different granularities. In Figure 4.1, for example, both "*good*" (ADJP) and "*tasted good*" (VP) are constituents and thus considered as phrases, whereas "*tasted*" is considered as a single word. For each phrase, we generate a candidate sentence using the edit operations explained below, with Figure 1.1 being a running example.

Removal

For each phrase detected by the parser, this operation generates a new candidate sentence by removing that phrase from the source sentence. In Figure 1.1, our algorithm can drop the phrase "*according to a Seattle based reporter*," which is not the main clause of the

sentence. The removal operation allows us to remove peripheral information in a sentence for content reduction. Assuming that a sentence has N phrases, for each iteration, the removal operation will generate N candidates (with each phrase being absent).

Extraction.

This operation simply extracts a selected phrase (including a clause) as the candidate sentence. This allows us to select the main clause in a sentence and remove remaining peripheral information. Assuming that a sentence has N phrases, for each iteration, the extraction operation will also generate N candidates.

Reordering

For each phrase in a sentence, we generate candidate sentences by moving the phrase before or after another phrase (identified by clause- and phrase-level constituent tags). In the running example, the phrase "*In 2016 alone*" is moved between the phrases "*12 billion dollars*" and "*on constructing theme parks.*" As seen, the reordering operation is able to perform syntactic simplification.

We reorder each phrase by moving it to each of the N-1 remaining spots in the sentence. Thus for each iteration, we get N(N-1) candidates. We do not manually specify rules for generating reordered sentences but instead rely on our scoring function to detect ungrammatical sentences. We believe this further indicates the capability of our scoring function.

Substitution

In each phrase, we identify the most complex word as the rarest one according to the idf score. For the selected complex word, we generate possible substitutes using a two-step strategy.

First, we obtain candidate synonyms by taking the union of the WordNet synonym set [28] and the closest words from GloVe [38] and Word2Vec [27] embeddings (where embedding closeness is measured by Euclidean distance). Second, a candidate synonym is determined to be an appropriate simple substitute if it satisfies the following conditions: a) it has a lower idf score than the complex word, where the scores are computed from the target simple sentences, b) it is not a morphological inflection of the complex word, c) its

word embedding exceeds a cosine similarity threshold to the complex word, and, d) it is has the same part-of-speech and dependency tags in the sentence as the complex word. We then generate candidate sentences by replacing the complex word with all qualified lexical substitutes. Notably, we do not replace entity words identified by entity taggers.

In our example sentence, consider the phrase "*constructing theme parks*." The word "constructing" is chosen as the word to be simplified, and is replaced with "building." The number of candidate synonyms can be controlled as a hyperparameter. Assuming that on average, there are x substitutes for the most complex word in each phrase, our system generates x.N candidates. As seen, this operation performs lexical simplification.

4.4 Stopping Criteria

Given an input complex sentence, our algorithm iteratively performs edits to search for a higher-scoring candidate.

In each iteration, we consider all the operations (i.e., removal, extraction, reordering, and substitution). Each operation may generate multiple candidates (e.g., multiple words for substitution). We calculate a score for each candidate sentence using the scoring function and filter it out if it does not increase the score by an operation-specific threshold. Thus, we only keep a candidate sentence if it scores better than the source complex sentence (best candidate sentence from the previous iteration) for that iteration by a multiplicative factor, i.e.,

$$f(c)/f(s) > r_{\text{op}} \tag{4.4}$$

where s is the sentence given by the previous iteration, and c is a candidate generated by operator op from s.

Notably, we allow different thresholds for each operation. This provides control over different aspects of simplification, namely, lexicon simplification, syntactic simplification, and content reduction. A lower threshold for substitution, for example, encourages the model to perform more lexical simplification.

From the candidate sentences that are not filtered out, we choose the highest-scoring sentence. It acts as the source sentence for the next iteration. Our algorithm terminates if no candidate sentence passes the threshold.

Chapter 5

Experiments

5.1 Datasets

We use the Newsela [54] and the WikiLarge datasets [57] for evaluating our model.

WikiLarge

The WikiLarge corpus (also called the Turkcorpus dataset) created by Zhang et al. [57] is currently the largest text simplification corpus. It contains 296,402, 2,000, 359 complex-simple sentence pairs for training, validating and testing, respectively. The training set of WikiLarge consists of automatically aligned sentence pairs from the standard and simple Wikipedia versions. The validation and test sets contain multiple human-written references (from Amazon Mechanical Turk workers), which we use to tune and test our model.

Newsela

Newsela is a collection of 1,840 news articles written by professional editors at five reading levels for children (0 is the most complex and 4 is simplest). It was created by Xu et al. [54], who highlighted the problems with the previous simplification datasets. We use the standard split and exclude simple-complex sentence pairs that are one reading level apart, following Zhang et al. [57]. Thus, we use the first 1,070 documents for training, the next 30 documents for development and the last 30 documents for testing. This gives us 95,208 training, 1,129 validation, and 1,077 test sentences. Table 5.1 shows an example of a sentence and its simplified versions for different reading levels, as written by professional editors. However, one drawback of the Newsela dataset is that it is only available for use for academic and non-profit researchers.

Reading Level	Sentence
0	In posting the highest women's score in the short and long program over the past three days, Lipnitskaia established herself as a gold medal contender in the women's singles event, which begins next week.
1	Lipnitskaia posted the highest women's score in the short and long program over the past three days, establishing herself as a gold medal contender in next week's women's singles event.
2	She firmly established herself as a gold medal contender in next week's women's singles event.
3	She made herself a clear possible winner of the gold medal in next week's women's singles event.
4	She is also a favorite to win a gold medal in next week's ladies singles event.

Table 5.1: Example of sentences with different reading levels from Newsela dataset written by professional editors.

	WikiLarge	**Newsela**
Vocabulary size		
Complex	201,841	41,066
Simple	168,962	30,193
Average Sentence Length		
Complex	22.61	23.06
Simple	18.51	12.75

Table 5.2: Statistics for the WikiLarge and Newsela datasets.

We omit datasets such as the WikiSmall corpus (also commonly known as Simple Wikipedia) [60] because Xu et al. [54] found that almost 50% of the sentences in Simple Wikipedia are either not simplified or not aligned correctly.

Table 5.2 provides the statistics about the WikiLarge and the Newsela datasets. Reference simplifications in the Newsela dataset are much shorter than those in the WikiLarge dataset. Since the WikiLarge dataset is bigger, it has a larger vocabulary size.

For each corpus, we only use its training set to learn a language model of simplified sentences. For the WikiLarge dataset, we also train a Word2Vec embedding model from

scratch on its source and target training sentences [1]. These embeddings are used to obtain candidate synonyms in the substitution operation.

5.2 Training Details

5.2.1 Structural Language Model

For the LM, we use a two-layer, 256-dimensional recurrent neural network (RNN) with the gated recurrent unit [6, GRU]. We use the pre-trained 300-dimensional Glove word embeddings [38] and initialize the part-of-speech and dependency tag embeddings randomly with 150-dimensional vectors. We fine-tune all embeddings during training. Out-of-vocabulary words are initialized with uniform random values in the range ± 0.05. We use the averaged stochastic gradient descent (ASGD) algorithm [39] to train the LM, with 0.4 as the dropout and 32 as the batch size.

5.2.2 Simplification model

We evaluate our model with different subsets of operations, i.e., removal (`RM`), extraction (`EX`), reordering (`RO`), and lexical substitution (`LS`). In our experiments, we test the following variants: `RM+EX`, `RM+EX+LS`, `RM+EX+RO`, and `RM+EX+LS+RO`.

For the Newsela dataset, the thresholds r_{op} for the stopping criteria are set to 1.25 for all the edit operations. All the weights in our scoring function $(\alpha, \beta, \gamma, \delta)$ are set to 1. For the WikiLarge dataset, the thresholds are set as 1.25 for the removal and reordering operations, 0.8 for substitution, and 5.0 for extraction. The weights in the scoring function $(\alpha, \beta, \gamma, \delta)$ are set to $0.5, 1.0, 0.25$ and 1.0, respectively. These hyper-parameter values are set after tuning the model on the validation dataset.

We use CoreNLP [22] for constructing the constituency tree and Spacy[2] to generate part-of-speech and dependency tags.

[1] Training a similar Word2Vec embedding model is not beneficial for the Newsela dataset due to the smaller size of its training dataset.

[2] https://spacy.io/

5.3 Baseline Models

We first consider the reference to obtain an upper-bound for a given evaluation metric. We also consider the complex sentence itself as a trivial baseline, denoted by `Complex`.

Next, we develop a simple heuristic that removes rare words occurring ≤ 250 times in the simple sentences of the training corpus, denoted by `Reduce-250`. As discussed in Section 5.4, this simple heuristic demonstrates the importance of balancing different automatic evaluation metrics.

For unsupervised competing methods, we compare with Surya et al., [47], which is inspired by unsupervised neural machine translation. They proposed two variants, `UNMT` and `UNTS`, but their results are only available for WikiLarge.

We also compare our model with supervised methods. First, we consider non-neural phrase-based machine translation (PBMT) methods: `PBMT-R` [53], which re-ranks sentences generated by PBMT for diverse simplifications; `SBMT-SARI` [55], which uses an external paraphrasing database; and `Hybrid` [31], which uses a combination of PBMT and discourse representation structures. Next, we compare our method with neural machine translation (NMT) systems: `EncDecA`, which is a vanilla Seq2Seq model with attention [33]; `Dress` and `Dress-Ls`, which are based on deep reinforcement learning [57]; `DMass` [58], which is a transformer-based model with external simplification rules; `EncDecP`, which is an encoder-decoder model with a pointer-mechanism; `EntPar`, which is based on multi-task learning [9]; `S2S-All-FA`, which a reranking based model focussing on lexical simplification [16]; and `Access`, which is based on the transformer architecture [24]. Finally, we compare with a supervised edit-based neural model, `Edit-NTS` [7].

5.4 Results

In this section, we present the results of automatic and human evaluation for our model, compare it with the state-of-the-art supervised and unsupervised systems, show the controllability of our model and do ablation studies.

5.4.1 Automatic Evaluation

Tables 5.3 and 5.4 present the results of the automatic evaluation on the Newsela and WikiLarge datasets, respectively.

Following previous work, we use the SARI metric [55] to measure the simplicity of the generated sentences. SARI computes the arithmetic mean of the n-gram F1 scores of three rewrite operations: adding, deleting, and keeping. The individual F1-scores of these operations are reported in the columns "Add," "Delete," and "Keep."

Next, we compute the BLEU metric [34] to measure the closeness (n-gram overlap) between a candidate and a reference. Xu et al. [55] and Sulem et al. [46] showed that BLEU correlates with human judgement on fluency and meaning preservation for text simplification[3].

In addition, we include a few intrinsic measures (without reference) to evaluate the quality of a candidate sentence: the Flesch–Kincaid grade level (FKGL) evaluating the ease of reading, as well as the average length of the sentence (denoted 'Len').

A few recent text simplification studies [7, 16] did not use BLEU for evaluation, noticing that the complex sentence itself achieves a high BLEU score (albeit a low SARI score), since the complex sentence is indeed fluent and preserves meaning. This is also shown by our `Complex` baseline.

For the Newsela dataset, however, we notice that the major contribution to the SARI score is from the deletion operation. By analyzing previous work such as `EntPar`, we find that it reduces the sentence length to a large extent, and achieves high SARI due to the extremely high F1 score of "Delete." However, its BLEU score is low, showing the lack of fluency and meaning. This is also seen from the high SARI of our heuristic based method (`Reduce-250`) in Table 5.3. Thus, simply removing infrequent words gives a good SARI score.

Therefore, using SARI alone is not sufficient to evaluate the overall simplification quality in an automated way. Ideally, we want high SARI as well as BLEU scores. Thus, we calculate the geometric mean (GM) of the SARI and BLEU as the main evaluation metric for the Newsela dataset.

On the other hand, this is not the case for WikiLarge, since none of the models achieve high SARI by using only one operation among "Add," "Delete," and "Keep." Moreover, the complex sentence itself yields an almost perfect BLEU score (partially due to the multi-reference nature of WikiLarge). Thus, we do not use GM, and for this dataset, SARI is our main evaluation metric.

Overall results on Newsela.

[3]This does not hold when splitting is involved; however, in the Newsela dataset, splitting happens for only 0.76% and 0.18% of the cases in the training and validation splits.

Table 5.3 shows the results on Newsela. By default (without †), validation is performed using the GM score. Still, our unsupervised text simplification achieves a SARI score around 26–27, outperforming quite a few supervised methods. Further, we experiment with SARI-based validation (denoted by †), following the setting of most previous work [7, 9]. We achieve 30.44 SARI, which is competitive with state-of-the-art supervised methods.

Our model also achieves high BLEU scores. As seen, all our variants, if validated by GM (without †), outperform competing methods in BLEU. One of the reasons is that our model performs text simplification by making edits on the original sentence instead of rewriting it from scratch.

In terms of the geometric mean (GM), our unsupervised approach outperforms all previous work, showing a good balance between simplicity and content preservation. The readability of our generated sentences is further confirmed by the intrinsic FKGL score.

Overall results on WikiLarge.

For the WikiLarge experiments in Table 5.4, we perform validation on SARI, which is the main metric in this experiment. Our model outperforms existing unsupervised methods, and is also competitive with state-of-the-art supervised methods.

We observe that lexical simplification (`LS`) is important in this dataset, as improvement due to it is large compared with the Newsela experiments in Table 5.3. Additionally, reordering (`RO`) does not improve performance, as it is known that WikiLarge does not focus on syntactic simplification [55]. The best performance for this experiment is obtained by the `RM+EX+LS` model.

Table 5.5 shows the simplified sentences generated by the previous simplification models and our proposed model. The first two example sentences are taken from the Newsela dataset and the last one is taken from the WikiLarge dataset. In case of the sentences from the Newsela dataset, we observe the models remove phrases that are not essential in conveying the central idea of the sentences. The example from the WikiLarge dataset focuses on lexical simplification.

5.4.2 Controllability

We now perform a detailed analysis of the scoring function described in Section 4.2 to understand the effect on different aspects of simplification. We use the `RM+EX+LS+RO` variant and the Newsela corpus as the testbed. The threshold values are set to 1.25 and the weights in the scoring function all set to one by default.

The SLOR score with syntax-aware LM.

Method	SARI$^{\uparrow}$	Add$^{\uparrow}$	Delete$^{\uparrow}$	Keep$^{\uparrow}$	BLEU$^{\uparrow}$	GM$^{\uparrow}$	FKGL$^{\downarrow}$	Len
Reference	70.13	-	-	-	100	83.74	3.20	12.75
Baselines								
Complex	2.82	-	-	-	21.30	7.75	8.62	23.06
Reduce-250	28.39	-	-	-	11.79	18.29	-0.23	14.48
Supervised Methods								
PBMT-R	15.77	3.07	38.34	5.90	18.1	16.89	7.59	23.06
Hybrid	28.61*	0.95*	78.86*	6.01*	14.46	20.34	4.03	12.41
EncDecA	24.12	2.73	62.66	6.98	21.68	22.87	5.11	16.96
Dress	27.37	**3.08**	71.61	7.43	23.2	25.2	4.11	14.2
Dress-Ls	26.63	3.21	69.28	7.4	**24.25**	25.41	4.21	14.37
DMass	31.06	1.25	84.12	7.82	11.92	19.24	3.60	15.07
S2S-All-FA	30.73	2.64	81.6	**7.97**	19.55	24.51	2.60	10.81
Edit-NTS	30.27*	2.71*	80.34*	7.76*	19.85	24.51	3.41	10.92
EncDecP	28.31	-	-	-	23.72	**25.91**	-	-
EntPar	**33.22**	2.42	**89.32**	7.92	11.14	19.24	1.34	7.88
Unsupervised Methods (Ours)								
RM+EX	26.07	2.35	68.35	7.5	**27.22**	26.64	2.95	12.9
RM+EX+LS	26.26	2.28	68.94	7.57	27.17	**26.71**	2.93	12.88
RM+EX+RO	26.99	**2.47**	70.88	7.63	26.31	26.64	3.14	12.81
RM+EX+LS+RO	27.11	2.40	71.26	**7.67**	26.21	26.66	3.12	12.81
RM+EX+LS+RO†	**30.44**	2.05	**81.77**	7.49	17.36	22.99	2.24	9.61

Table 5.3: Results on the Newsela dataset. † denotes the model with parameters tuned by SARI; other variants are tuned by the geometric mean (GM). $^{\uparrow}$The higher, the better. $^{\downarrow}$The lower, the better. * indicates a number that is different from that reported in the original paper. This is due to a mistreatment of capitalization in the previous work (confirmed by personal correspondence).

In Table 5.6 we analyze our syntax-aware SLOR score in the search objective. First, we remove the SLOR score and use the standard sentence probability. We observe that SLOR helps preserve rare words, which may be entities. As a result, the readability score (FKGL) becomes better (i.e., lower), but the BLEU score decreases. We then evaluate the importance of using a structural LM instead of a standard LM. We see a decrease in both SARI and BLEU scores. In both cases, the GM score decreases.

Threshold values and relative weights.

Table 5.7 analyzes the effect of the hyperparameters of our model, namely, the threshold in the stopping criteria and the relative weights in the scoring function.

As discussed in Section 4.4, we use a threshold as the stopping criteria for our iterative

Method	SARI$^{\uparrow}$	Add$^{\uparrow}$	Delete$^{\uparrow}$	Keep$^{\uparrow}$	BLEU$^{\uparrow}$	FKGL$^{\downarrow}$	Len
Baselines							
Complex	27.87	-	-	-	99.39	-	22.61
Supervised Methods							
PBMT-R	38.56	5.73	36.93	73.02	81.09	8.33	22.35
Hybrid	31.40	1.84	45.48	46.87	48.67	**4.56**	13.38
EncDecA	35.66	2.99	28.96	**75.02**	**89.03**	8.42	21.26
Dress	37.08	2.94	43.15	65.15	77.41	6.59	16.14
Dress-Ls	37.27	2.81	42.22	66.77	80.44	6.62	16.39
Edit-NTS	38.23	3.36	39.15	72.13	86.69	7.30	18.87
EntPar	37.45	-	-	-	81.49	7.41	-
Access	**41.87**	**7.28**	**45.79**	72.53	75.46	7.22	22.27
Models using external knowledge base							
SBMT-SARI	39.96	5.96	41.42	72.52	73.03	7.29	23.44
DMass	40.45	5.72	42.23	73.41	-	7.79	-
Unsupervised Methods							
UNMT	35.89	1.94	37.68	68.04	70.61	8.23	21.85
UNTS	37.20	1.50	41.27	68.81	74.02	7.84	19.05
RM+EX	36.46	1.68	35.17	**72.54**	**88.90**	6.47	18.62
RM+EX+LS	**37.85**	**2.31**	43.65	67.59	73.62	**6.30**	18.45
RM+EX+RO	36.54	1.73	36.10	71.79	85.07	6.89	19.24
RM+EX+LS+RO	37.58	2.30	**43.97**	66.46	70.15	6.69	19.54

Table 5.4: Results on the WikiLarge dataset. $^{\uparrow}$The higher, the better. $^{\downarrow}$The lower, the better.

search algorithm. For each operation, we require that a new candidate should be better than the previous iteration by a multiplicative threshold r_{op} in Equation (4.4). In this analysis, we set the same threshold for all operations for simplicity. As seen in Table 5.7, increasing the threshold leads to better meaning preservation since the model is more conservative (making fewer edits). This is shown by the higher BLEU and lower SARI scores.

Regarding the weights for each individual scoring function, we find that increasing the weight β for the FRE readability score makes sentences shorter, more readable, and thus simpler. This is also indicated by higher SARI values. When sentences are rewarded for being short (with large γ), SARI increases but BLEU decreases, showing less meaning preservation. The readability scores initially increase with the reduction in length, but then decrease. Finally, if we increase the weight δ for the entity score, the sentences become

Model	Sentence
	Newsela
Complex	**Breyfogle, a former college basketball player**, was reluctant at first, but he signed up his son , Easton , to play on Edina's team.
Reference	Still, he was not sure if he should sign up his son, Easton, to play traveling basketball.
Hybrid	A player was a player to play.
Dress-Ls	Breyfogle, a former college basketball player, was afraid at first.
EditNTS	Breyfogle play on Edina's team.
S2SALLFA	Breyfogle , a former college basketball player.
Ours	He signed up his son , Easton , to play on Edina's team.
Complex	**Afterward,** a group of teens **from the community** came together to bring change **to their local schools**.
Reference	A group of teens came together to change things.
Hybrid	A group came to bring change.
Dress-Ls	A group of teens from the community came together to bring change to their local schools.
EditNTS	A group of teens from the community came together to teach schools.
S2SALLFA	Afterward, a group of teens from the community came together.
Ours	Afterward , a group of teens came together.
	WikiLarge
Complex	The collapsed dome of the main church has been restored entirely.
Reference	The fallen dome of the main church was entirely rebuilt .
Dress-Ls	The collapsed dome of the main church has been restored entirely.
EditNTS	The collapsed dome of the main church has been **kept**.
Access	The collapsed dome of the main church has been **taken over by the church.**
UNTS	The collapsed dome of the main church has been restored entirely.
Ours	The collapsed dome of the main church has been **rebuilt**.

Table 5.5: Examples of simplified sentences by supervised and unsupervised models for the Newsela and WikiLarge dataset.

Method	SARI$^{\uparrow}$	Add$^{\uparrow}$	Delete$^{\uparrow}$	Keep$^{\uparrow}$	BLEU$^{\uparrow}$	GM$^{\uparrow}$	FKGL$^{\downarrow}$	Len
RM+EX+LS+RO	27.11	2.40	71.26	7.67	26.21	26.66	3.12	12.81
− SLOR	27.63	2.22	73.20	7.49	24.14	25.83	2.61	12.37
− syntax-awareness	26.91	2.16	71.19	7.39	24.98	25.93	3.65	12.76

Table 5.6: Ablation test of the SLOR score based on syntax-aware language modeling.

longer and more complex since the model is penalized more for deleting entities.

In summary, the above analysis shows the controllability of our approach in terms of different simplification aspects, such as simplicity, meaning preservation, and readability.

Value	SARI	Add	Delete	Keep	BLEU	GM	FKGL	FRE	Len
Effect of threshold r_{op}									
1.0	29.20	2.22	77.60	7.77	21.69	25.17	2.64	95.76	11.75
1.1	28.38	2.28	75.14	7.73	23.59	25.87	3.07	94.40	12.17
1.2	27.45	2.37	72.31	7.67	25.54	26.48	3.10	94.20	12.62
1.3	26.60	2.40	69.78	7.63	26.47	26.53	3.53	92.93	13.07
Effect of weight α for f_{eslor}									
0.75	27.04	2.33	71.27	7.53	25.75	26.39	2.94	95.38	12.46
1.25	26.91	2.38	70.73	7.61	25.96	26.43	3.17	93.72	12.96
1.50	26.74	2.16	70.57	7.48	25.20	25.96	3.61	92.34	13.06
2.0	26.83	2.23	70.89	7.37	24.29	25.53	3.71	91.61	13.15
Effect of weight β for f_{fre}									
0.5	26.42	2.31	69.51	7.45	25.53	25.97	3.89	90.33	13.20
1.5	27.38	2.45	71.98	7.71	26.04	26.70	2.83	96.15	12.58
2.0	27.83	2.42	73.31	7.75	25.27	26.52	2.54	98.19	12.26
3.0	28.29	2.36	74.86	7.64	23.69	26.52	1.76	102.05	11.91
Effect of weight γ for f_{len}									
0.5	24.54	2.24	64.14	7.24	25.06	24.80	3.91	91.97	14.55
2.0	29.00	2.12	77.28	7.61	21.65	25.06	2.33	96.17	10.93
3.0	29.93	2.05	80.18	7.57	19.05	23.88	2.45	95.27	10.09
4.0	30.44	2.05	81.77	7.49	17.36	22.99	2.24	95.03	9.61
Effect of weight δ for f_{entity}									
0.5	27.81	2.39	73.42	7.61	24.68	26.20	2.52	96.58	12.01
2.0	25.44	2.31	66.66	7.34	24.63	25.03	4.12	90.45	14.28

Table 5.7: Analysis of the threshold value of the stopping criteria and relative weights in the scoring function.

5.4.3 Human Evaluation

We conducted a human evaluation on the Newsela dataset since automated metrics may be insufficient for evaluating text generation. We chose 30 sentences from the test set for annotation and considered a subset of baselines. For our model variants, we chose `RM+EX+LS+RO`, considering both validation settings (GM and SARI).

We followed the evaluation setup in Dong et al., [7], and measure the adequacy (*How much meaning from the original sentence is preserved?*), simplicity (*Is the output simper*

Method	A	S	F	Avg	FI
Hybrid	2.63	2.74	2.39	2.59	**0.03**
Dress-Ls	**3.29**	3.05	**4.11**	**3.48**	0.2
EntPar	1.92	2.97	3.16	2.68	0.47
S2S-All-FA	2.25	**3.24**	3.90	3.13	0.3
Edit-NTS	2.37	3.17	3.73	3.09	0.23
Base+LS+RO	2.97	3.09	3.78	3.28	**0.03**
Base+LS+RO*	2.58	3.21	3.33	3.04	0.07
Reference	2.91	3.49	4.46	3.62	0.77

Table 5.8: Result of human evaluation on Newsela, measuring adequacy (A), simplicity (S), fluency (F), their average score on a five-point Likert scale (Avg), and average instances of false information per sentence (FI).

than the original sentence?), and fluency (*Is the output grammatical?*) on a five-point Likert scale. We recruited three volunteers, one native English speaker and two non-native fluent English speakers. Each of the volunteer was given 30 sentences from different models (and references) in a randomized order. Additionally, we asked the volunteers to measure the number of instances where models produce incorrect details or generate text that is not implied by the original sentence. We did this because neural models are known to hallucinate information [41]. We report the average count of false information per sentence, denoted as FI.

We observe that our model `RM+EX+LS+RO` (when validated by GM) performs better than `Hybrid`, a combination of PBMT and discourse representation structures, in all aspects. It also performs competitively with remaining supervised NMT models.

For adequacy and fluency, `Dress-Ls` performs the best since it produces relatively longer sentences. For simplicity, `S2S-All-FA` performs the best since it produces shorter sentences. Thus, a balance is needed between these three measures. As seen, `RM+EX+LS+RO` ranks second in terms of the average score in the list (reference excluded). The human evaluation confirms the effectiveness of our unsupervised text simplification, even when compared with supervised methods.

We also compare our model variants `RM+EX+LS+RO` (validated by GM) and `RM+EX+LS+RO`† (validated by SARI). As expected, the latter generates shorter sentences, performing better in simplicity but worse in adequacy and fluency.

Regarding false information (FI), we observe that previous neural models tend to generate more false information.By contrast, our approach only uses neural networks in the

scoring function, but performs discrete edits of words and phrases. Thus, we achieve high fidelity (low FI) similar to the non-neural `Hybrid` model, which also performs editing on discourse parsing structures with PBMT. Interestingly, the reference of Newsela has a poor (high) FI score, because the editors wrote simplifications at the document level, rather than the sentence level.

In summary, our model takes advantage of both neural networks (achieving high adequacy, simplicity, and fluency) and traditional phrase-based approaches (achieving high fidelity).

Chapter 6

Conclusions and Future Work

In this work, we propose a simple and interpretable unsupervised approach for text simplification. We observe that there are two fundamental ways of designing the unsupervised models for text simplification. First, there are neural approaches based on the Seq2Seq architecture [47] and second, edit-based approaches (our approach as well as the model proposed by Narayan and Gardent [32]).

Neural approaches require large amounts of data and are hard to interpret, but have more coverage and can perform complex operations such as paraphrasing. Edit-based approaches provide more interpretability and controllability but are limited in coverage by the pre-defined edit operations. For example, our model does not perform paraphrasing. It is important to emphasize that edit-based approaches could also have a neural component. For instance, our approach uses a neural network for the syntax-aware language model. However, the distinction is that the unsupervised neural models are generally based on the Seq2Seq architecture and do not explicitly define simplification operations.

In this thesis, we present the following contributions and insights:

1. We design an unsupervised edit-based model that performs content reduction, and structural and lexical simplification.

2. We define explicit edit operations (such as content deletion, lexical simplification and reordering) that provide interpretable results. We observe that the deletion and reordering operations are more effective in the case of the Newsela dataset. In contrast, deletion and lexical simplification operations are more effective in the case of the WikiLarge dataset. Such observations are hard to make without manually analyzing the simplified sentences generated by the less interpretable neural models.

3. Our model provides controllability over the different aspects of simplification. The results presented in Table 5.7 demonstrate the controllability of our proposed model. A controllable text simplification model is useful for generating simplified sentences for a diverse set of target users having different requirements.

4. We show that solely optimizing simplicity is not the best strategy for text simplifications. Instead, we argue for models to optimize fluency, adequacy and simplicity in parallel. We observe that jointly optimizing the models on these three axes is hard. For example, the models achieve high scores on the SARI metric at the cost of the BLEU metric (shown in Table 5.3 and Table 5.4).

5. We thus propose to use a new automatic metric to evaluate text simplification models. We calculate the geometric mean of scores from the SARI and BLEU metrics. We show that it correlates well with human evaluation on overall simplification (shown in Table 5.8).

6. We evaluate our model on two large datasets (Newsela and WikiLarge) and show that it performs better than previous unsupervised models and competitively to the state-of-the-art supervised models.

7. Comparing to Narayan and Gardent [32], the work closest to ours, we have a single module that is responsible for performing all the edit operations. Thus, we provide a more general framework for text simplification and additionally, adding new edit operators is more natural. Lastly, the order of the edit operations is not fixed and is decided by the model.

8. In comparison to the previous supervised edit-based models that perform edits only on words, our model instead operates on both words and phrases. We observe that operating on phrases is beneficial for removing peripheral clauses and provides better computational efficiency for longer sentences with multiple clauses.

9. We provide an open-source implementation of our model and release our trained models and system outputs for reproducibility. Our code is available at https://github.com/ddhruvkr/Edit-Unsup-TS.

Moving forward, we see several possible directions to build on our work. Simplification often depends on the audience for whom it is done. For example, if the target audience is children, we might focus more on content reduction. However, if the target audience is adult non-native speakers, we could focus more on lexical simplification while preserving content.

Since our approach is both interpretable and controllable, it could be applied towards developing personalized simplification systems. In our model, the threshold weights of edit operators (described in Section 4.4) and the relative weights of the different components of the scoring function (described in Section 4.2) can be tuned as per the target audience. We show the effect of varying these weights in Table 5.7 as part of our controllability experiments.

As mentioned above, one of the weaknesses of our model is that it does not perform paraphrasing (i.e., it can only lexically simplify words and not phrases). So another exciting direction is extending our model for paraphrasing. We outline two such approaches.

Explicit paraphrasing

We could enable our model to do paraphrasing by adding new edit operations. One approach is to add a paraphrasing edit operation to our method. Thus, for every phrase in a complex sentence, we also generate new candidate sentences by paraphrasing the given phrase. To do so, we train a paraphrasing model using the Seq2Seq architecture [48] on the Parabank dataset [11]. Parabank dataset is one of the largest and diverse paraphrasing datasets publically available. It was created by training a Czech-English neural machine translation(NMT) model and using the generated translations as paraphrases for the English reference sentences. Training the paraphrasing model does not make our approach supervised since we still do not require the aligned complex-simple sentence pairs.

We use a smaller version of the dataset containing five million paraphrasing pairs. Additionally, we apply data cleaning and remove some paraphrase pairs that contain special characters and those exceeding sentence length of 15. The paraphrasing model has three layers with a hidden dimension of 256.

Table 6.1 shows the results of incorporating the paraphrasing operation in our edit-based model `RM+EX+LS+PR`. We do not observe any improvement in the SARI metric even as the Add and Delete operation scores increase. One of the reasons for this could be that the Parabank dataset contains entire sentences in its training corpus. In contrast, we require the paraphrases for sub-phrases (and not entire sentences).

Another way to explicitly add paraphrasing is to break the paraphrasing operation into separate edit operations such as "add" and "replace". A similar approach is used in Miao et al. [26]. Additionally, this approach does not require any external dataset for training a separate paraphrasing model. We will explore this approach in the future.

Method	SARI$^{\uparrow}$	Add$^{\uparrow}$	Delete$^{\uparrow}$	Keep$^{\uparrow}$	BLEU$^{\uparrow}$	Len
UNTS	37.20	1.50	41.27	68.81	74.02	19.05
RM+EX+LS	37.85	2.31	43.65	67.59	73.62	18.45
RM+EX+LS+PR	37.76	**2.91**	**44.17**	66.21	67.58	20.09
Ensemble	**38.18**	2.24	43.33	**68.98**	**75.25**	18.99

Table 6.1: Performance of our model after adding implicit and explicit paraphrasing using the WikiLarge corpus. $^{\uparrow}$The higher, the better. $^{\downarrow}$The lower, the better.

Implicit paraphrasing

Alternatively, we could implicitly add paraphrasing through an ensemble approach. The hypothesis is that since neural models are better at paraphrasing than the edit-base approaches, they might produce better simplifications for specific instances. Alternatively, there might also be instances where edit-based approaches produce better simplifications. This could be attributed to the properties of the complex sentences that makes it easier for a model to generate simplifications. For example, a neural model may produce better simplifications for shorter sentences and an edit-based model could give a better performance for longer sentences. Even though the implicit paraphrasing method has its merits, we lose interpretability when the chosen simplified sentence is from the neural Seq2Seq model.

We also present initial results for the implicit paraphrasing (ensemble) approach. Given a complex sentence, we simplify it using the unsupervised neural seq2seq-based approach [47] as well as our method. We score the simplified sentences generated by both methods using our scoring function and select the higher scoring sentence.

We evaluate the `Ensemble` method on the test set of the WikiLarge corpus and show the results in Table 6.1.

The ensemble model achieves higher SARI and BLEU scores than the individual models. We observe that in 62% of cases, the simplified sentences generated by our model are preferred over that of the `UNTS` model. However, for 6% of the cases, both models generate similar simplifications (in these cases, no change is made to the complex sentences).

Interestingly, for longer complex sentences, the scoring function favours the simplified sentences generated by our model. The average length of the complex sentences when simplifications generated by our model are preferred is 24.26, whereas the average length drops to 21.56 when the simplified sentences from `UNTS` model are chosen. For cases, when both the models produce similar simplifications, the average length of the complex sentences is 11.96. Thus, for extremely short sentences, both the models leave the original

complex sentences unchanged. Current text simplification models do not explicitly decide whether a complex sentence should be simplified or left unchanged. Thus, they could oversimplify a sentence. McNamara et al. [25] claim that oversimplified sentences could be difficult to comprehend. Thus, another future direction is to study use-cases where not simplifying a sentence (to prevent oversimplification) might be a better strategy.

Another direction for future work is to explore our scoring function as a metric for evaluation of simplification. Most metrics currently used for evaluating text simplification systems require the use of parallel corpus. Thus, they are limited by quality reference data which, as argued before, require considerable human effort to obtain. Since our scoring function is designed to identify simplicity in the sentence without requiring the reference sentence, we see potential in extending it as an evaluation metric.

References

[1] Fernando Alva-Manchego, Joachim Bingel, Gustavo Paetzold, Carolina Scarton, and Lucia Specia. Learning how to simplify from explicit labeling of complex-simplified text pairs. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 295–305, 2017.

[2] Joachim Bingel and Anders Søgaard. Text simplification as tree labeling. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 337–343, 2016.

[3] Or Biran, Samuel Brody, and Noémie Elhadad. Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 496–501. Association for Computational Linguistics, 2011.

[4] John Carroll, Guido Minnen, Darren Pearce, Yvonne Canning, Siobhan Devlin, and John Tait. Simplifying text for language-impaired readers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, 1999.

[5] Raman Chandrasekar, Christine Doran, and Bangalore Srinivas. Motivations and methods for text simplification. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 1041–1044. Association for Computational Linguistics, 1996.

[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[7] 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.

[8] Richard Evans, Constantin Orăsan, and Iustin Dornescu. An evaluation of syntactic simplification rules for people with autism. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*, pages 131–140, Gothenburg, Sweden, April 2014. Association for Computational Linguistics.

[9] Han Guo, Ramakanth Pasunuru, and Mohit Bansal. Dynamic multi-level multi-task learning for sentence simplification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 462–476, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[10] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[11] J Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. Parabank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6521–6528, 2019.

[12] Hans Kamp. A theory of truth and semantic representation. *Formal semantics-the essential readings*, pages 189–222, 1981.

[13] Katharina Kann, Sascha Rothe, and Katja Filippova. Sentence-level fluency evaluation: References help, but can be spared! In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 313–323, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[14] 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. 1975.

[15] Beata Beigman Klebanov, Kevin Knight, and Daniel Marcu. Text simplification for information-seeking applications. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 735–747. Springer, 2004.

[16] Reno Kriz, João Sedoc, Marianna Apidianaki, Carolina Zheng, Gaurav Kumar, Eleni Miltsakaki, and Chris Callison-Burch. Complexity-weighted loss and diverse reranking for sentence simplification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language*

Technologies, Volume 1 (Long and Short Papers), pages 3137–3147, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[17] 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.

[18] Jey Han Lau, Alexander Clark, and Shalom Lappin. Grammaticality, acceptability, and probability: A probabilistic view of linguistic knowledge. *Cognitive Science*, 41(5):1202–1241, 2017.

[19] Juncen Li, Robin Jia, He He, and Percy Liang. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[20] Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. Unsupervised paraphrasing by simulated annealing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 302–312, Online, July 2020. Association for Computational Linguistics.

[21] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[22] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[23] Paulo RA Margarido, Thiago AS Pardo, Gabriel M Antonio, Vinícius B Fuentes, Rachel Aires, Sandra M Aluísio, and Renata PM Fortes. Automatic summarization for text simplification: Evaluating text understanding by poor readers. In *Companion Proceedings of the XIV Brazilian Symposium on Multimedia and the Web*, pages 310–315, 2008.

[24] Louis Martin, Benoît Sagot, Éric de la Clergerie, and Antoine Bordes. Controllable sentence simplification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[25] Danielle S McNamara, Max M Louwerse, and Arthur C Graesser. Coh-metrix: Automated cohesion and coherence scores to predict text readability and facilitate comprehension. Technical report, Technical report, Institute for Intelligent Systems, University of Memphis ..., 2002.

[26] Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842, 2019.

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[28] George A. Miller. Wordnet: a lexical database for english., 1995.

[29] Bhaskar Mitra and Nick Craswell. Neural text embeddings for information retrieval. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 813–814, 2017.

[30] Tsendsuren Munkhdalai and Hong Yu. Neural semantic encoders. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 397–407, Valencia, Spain, April 2017. Association for Computational Linguistics.

[31] Shashi Narayan and Claire Gardent. Hybrid simplification using deep semantics and machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 435–445, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[32] Shashi Narayan and Claire Gardent. Unsupervised sentence simplification using deep semantics. In *INLG 2016 - Proceedings of the Ninth International Natural Language Generation Conference, September 5-8, 2016, Edinburgh, UK*, pages 111–120, 2016.

[33] 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, 2017.

[34] 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 on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[36] Adam Pauls and Dan Klein. Large-scale syntactic language modeling with treelets. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 959–968. Association for Computational Linguistics, 2012.

[37] 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, Berlin, Germany, August 2016. Association for Computational Linguistics.

[38] Jeffrey Pennington, Richard Socher, and Christopher 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.

[39] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

[40] Luz Rello, Clara Bayarri, Azuki Gòrriz, Ricardo Baeza-Yates, Saurabh Gupta, Gaurang Kanvinde, Horacio Saggion, Stefan Bott, Roberto Carlini, and Vasile Topac. Dyswebxia 2.0!: more accessible text for people with dyslexia. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*, page 25. Citeseer, 2013.

[41] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4035–4045, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[42] Raphael Schumann, Lili Mou, Yao Lu, Olga Vechtomova, and Katja Markert. Discrete optimization for unsupervised sentence summarization with word-level extraction. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 5032–5042, Online, July 2020. Association for Computational Linguistics.

[43] Advaith Siddharthan. An architecture for a text simplification system. In *Language Engineering Conference, 2002. Proceedings*, pages 64–71. IEEE, 2002.

[44] Advaith Siddharthan. Text simplification using typed dependencies: a comparison of the robustness of different generation strategies. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 2–11. Association for Computational Linguistics, 2011.

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

[46] Elior Sulem, Omri Abend, and Ari Rappoport. BLEU is not suitable for the evaluation of text simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 738–744, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[47] 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.

[48] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[49] 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.

[50] Tu Vu, Baotian Hu, Tsendsuren Munkhdalai, and Hong Yu. Sentence simplification with memory-augmented neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 79–85, New Orleans, Louisiana, jun 2018. Association for Computational Linguistics.

[51] Willian Massami Watanabe, Arnaldo Candido Junior, Vinícius Rodriguez Uzêda, Renata Pontin de Mattos Fortes, Thiago Alexandre Salgueiro Pardo, and Sandra Maria Aluísio. Facilita: reading assistance for low-literacy readers. In *Proceedings of the*

27th ACM international conference on Design of communication, pages 29–36. ACM, 2009.

[52] Kristian Woodsend and Mirella Lapata. Text rewriting improves semantic role labeling. *Journal of Artificial Intelligence Research*, 51:133–164, 2014.

[53] Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1015–1024. Association for Computational Linguistics, 2012.

[54] 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.

[55] 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, 2016.

[56] Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, 2001.

[57] 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 595–605. Association for Computational Linguistics, 2017.

[58] Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3164–3173, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[59] Yang Zhao, Zhiyuan Luo, and Akiko Aizawa. A language model based evaluator for sentence compression. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 170–175, 2018.

[60] 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*, pages 1353–1361. Association for Computational Linguistics, 2010.