Controllable Text Simplification

Z Li

MPhil 2023

Controllable Text Simplification

Zihao Li

A thesis submitted in fulfilment of the requirements of Manchester Metropolitan University for the degree of Master of Philosophy

Department of Computing and Mathematics
Manchester Metropolitan University

2023

Contents

Word count: 21115

List of figures

List of tables

List of publications

Li, Zihao and Matthew Shardlow (2024). “How do control tokens affect natural language generation tasks like text simplification”. In: *Natural Language Engineering*, pp. 1–28. DOI: 10.1017/S1351324923000566.

Li, Zihao, Matthew Shardlow, and Fernando Alva-Manchego (Sept. 2023). “Comparing Generic and Expert Models for Genre-Specific Text Simplification”. In: *Proceedings of the Second Workshop on Text Simplification, Accessibility and Readability*. Ed. by Sanja Štajner et al. Varna, Bulgaria: INCOMA Ltd., Shoumen, Bulgaria, pp. 51–67. URL: https://aclanthology.org/2023.tsar-1.6.

Li, Zihao, Matthew Shardlow, and Saeed Hassan (Dec. 2022). “An Investigation into the Effect of Control Tokens on Text Simplification”. In: *Proceedings of the Workshop on Text Simplification, Accessibility, and Readability (TSAR-2022)*. Ed. by Sanja Štajner et al. Abu Dhabi, United Arab Emirates (Virtual): Association for Computational Linguistics, pp. 154–165. DOI: 10.18653/v1/2022.tsar-1.14. URL: https://aclanthology.org/2022.tsar-1.14.

Terms and abbreviations

API	application programming interface
ASSET	Abstractive Sentence Simplification Evaluation and Tuning dataset
BART	Bidirectional Auto-Regressive Transformers
BLEU	Bilingual evaluation understudy
BPE	byte-pair encoding
CNN	convolution neural networks
DRESS	Deep REinforcement Sentence Simplification
DTD	Dependency Tree Depth Ratio
EW	English Wikipedia
FKGL	Flesch-Kincaid Grade Level
LDA	Lightweight Dependency Analyzer
LLM	large language model
LR	Length Ratio
LSTM	Long-short term memory
LV	Replace Only Levenshtein Ratio
NapSS	Narrative Prompting and Sentence-matching Summarization
NLG	Natural language generation
NTS	Neural Text Simplification
OECD	Organisation for Economic Co-operation and Development

PISA	Programme for International Student Assessment
PWKP	Parallel Wikipedia Simplification
RNN	recurrent neural networks
SARI	System output Against Reference and Input
SEW	Simple English Wikipedia
SOTA	State of the Art
T5	Text-to-Text Transfer Transformer
TF-IDF	term frequency–inverse document frequency
TICO-19	Translation Initiative for COVID-19
WR	Word Rank Ratio

Abstract

Text simplification is a tool that enhances the accessibility of text at both lexical and syntactical levels. It aids individuals in comprehending complex texts more easily, particularly children, students, and people with reading difficulties. This thesis aims to investigate the effects, limitations, and potential enhancements of the current state-of-the-art method in text simplification.

The thesis consists of three main experiments and addresses the challenges in text simplification. In the first experiment, we focused on the various needs in text simplification, explored the impact of the control mechanism in text simplification, re-implemented the state-of-the-art system with less computation power, and redesigned the tokenization and quantization for the control mechanism to improve the performance by up to 0.5 points in the metrics. In the second experiment, we addressed the impact of text style in text simplification tasks in different domains, constructed a genre-specific test scenario focused on coronavirus, verified the effect of the genre in text simplification tasks, and compared these models with large language models (e.g. ChatGPT) as the generic model. In the final experiment, we addressed the lack of adaptation ability in the system, fine-tuned models to predict the value of four control tokens, integrated these predictors with the current system, and thereby enhanced the practicality and popularity of controllable text simplification systems.

As a result, we explored the mechanism of control tokens, verified the effectiveness of controllable text simplification in the genre-specific corpus, and improved the overall performance and adaptability of the controllable text simplification system.

Declaration of originality

I hereby confirm that no portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright statement

Chapter 1

Introduction

1.1 Motivation and Background

The research of readability received relatively little attention until the early 20th century (Zakaluk and Samuels 1988), primarily focusing on the identification and measurement of readability at the vocabulary and sentence levels (Dolch 1928; Vogel and Washburne 1928). Despite its long history of research, the problem remains unsolved and continues to be a significant concern. A study by the Programme for International Student Assessment (PISA) revealed alarming statistics regarding reading literacy among 15-year-old students from 79 countries and economies, indicating that a staggering 22.6% of students struggle with basic reading literacy issues, while only 10.9% demonstrate mastery of complex reading tasks (OECD 2019). This situation has worsened over the years, as evidenced by the declining median scores observed between the 2000 and 2018 PISA assessment results. Unfortunately, it is probable for this situation to further deteriorate during the COVID-19 pandemic. Various studies conducted in Europe have shown that school closures and disruptions caused by the pandemic significantly undermined learning outcomes and exacerbated educational inequalities (Tomasik, Helbling, and Moser 2021; Engzell, Frey, and Verhagen 2020; Maldonado and De Witte 2022).

It is crucial to note that these challenges are not limited to the younger population alone, as adults also face similar difficulties. A report published by the OECD in 2013 examines the literacy levels of adults aged 16 to 65 in 24 countries, as defined by the OECD. The report indicates that 16.7% of adults possess a literacy level lower than 2, struggling with tasks such as low-level inference, information integration, and identifying information from various parts of a document (OECD 2013). These findings emphasize the urgent need for a comprehensive approach to address readability issues. Recognizing the severity of the problem and its impact on both students and adults, it becomes increasingly imperative to develop effective strategies, tools, and interventions to enhance literacy levels and promote accessible and inclusive reading comprehension for all individuals.

As a subfield within natural language processing (NLP), automatic text simplification plays a crucial role in reducing linguistic complexity at both syntactic and lexical levels. The primary objective of this field is to modify the content and structure of the text while preserving its core ideas and meaning (Alva-Manchego, Scarton, and Specia 2020). This process enables the creation of simplified versions of texts, which have valuable applications in assisting various target audiences. It is commonly employed to aid children (De Belder and Moens 2010), non-native speakers (Petersen and Ostendorf 2007; Paetzold 2016), and individuals with dyslexia (Rello et al. 2013) in reading and comprehending complex texts.

One of the primary beneficiary groups of automatic text simplification is children. By using simplified texts, young readers can overcome the challenges posed by complex language structures and vocabulary, enabling them to comprehend and engage with a wider range of materials, thus fostering their literacy development (De Belder and Moens 2010). In addition, non-native speakers often encounter obstacles when dealing with intricate language constructs. Text simplification techniques offer support to these individuals, facilitating their language acquisition and comprehension (Petersen and Ostendorf 2007; Paetzold 2016). Another group that benefits from automatic text simplification is individuals with dyslexia. Dyslexia presents difficulties in reading and understanding complex texts, making the simplification process helpful for enhancing accessibility and promoting inclusivity for those with this learning difficulty (Rello et al. 2013).

By catering to these diverse target groups, automatic text simplification contributes to fostering inclusive education, improving literacy rates, and promoting equal access to information. Its applications extend beyond these specific contexts, offering potential benefits to individuals with cognitive impairments, older adults, and those with limited reading skills, among others. The field continues to evolve, exploring innovative approaches and technologies to further enhance the effectiveness and applicability of automatic text simplification in facilitating information access and comprehension for a broad range of users.

1.2 Research Questions

In the exploration of satisfying the different needs of various user groups, researchers (Scarton and Specia 2018; Martin, Sagot, et al. 2019; Martin, Fan, et al. 2020; Sheang and Saggion 2021) introduced the control mechanism, so that the features of a generated simplification (e.g. the length) can be determined during inference.

In this thesis, we focus on three major questions:

- Research Question 1: How do the control mechanism (e.g. the control tokens) affect the text simplification outcomes and how to effectively leverage these control mechanisms?
- Research Question 2: Can current controllable text simplification models compete with large language models in scenarios related to the professional domain?
- Research Question 3: How can we improve the practicability of current controllable text simplification systems?

1.3 Aim and Objectives

In this thesis, our main focus is to examine and respond to the following objectives:

In Chapter 3, covering the content in the first publication (Li, Shardlow, and Hassan 2022), our goal is to address the first research question by exploring the control mechanism and highlighting the importance of controlled text simplification. We aim to understand the principles behind the control tokens and find a way to better leverage the control mechanism.

In Chapter 4, covering the content in the second publication (Li, Shardlow, and Alva-Manchego 2023), our goal is to address the second research question by comparing the controllable text simplification systems with large language models on either genre-specific corpus or general corpus. We aim to verify the effectiveness of the controllable text simplification system in certain domains. At the same time, we also investigate the constraints of popular metrics. By objectively assessing existing metrics, we aim to uncover their limitations and explore potential areas for improvement.

In Chapter 5, covering the content in the third publication (Li and Shardlow 2024), our goal is to address the final research question by designing a new pattern for leveraging the control tokens. We aim to improve the adaptability and accuracy of controlled text simplification by proposing and investigating the control token predictors. Our objective is to prompt the application in real-life situations and advance the development of reliable, dependable solutions for text simplification.

1.4 Contributions

In this thesis, we give a general introduction and reviews on related literature in Chapter 1 and 2. In Chapter 3 to 5, we deliver three different experiments and made such contributions:

In Chapter 3, we reimplemented the current state-of-the-art (SOTA) in controllable text simplification to evaluate existing methods and conducted experiments to investigate how the control tokens affect the text simplification outcomes in Section 3.3.2 and 3.4. We also redesigned the tokenization strategy and application of control tokens in Section 3.3.1 and improved the System output Against Reference and Input (SARI) score by 0.5 on the ASSET test set (Alva-Manchego, Martin, Bordes, et al. 2020).

In Chapter 4, we leveraged the Simple TICO 19 (Shardlow and Alva-Manchego

2022) dataset and created subsets for genre-specific text simplification tasks. We applied transfer learning on the subsets to create genre-specific models and verified the performance in Section 4.3.2. As a reference, we evaluated the performance differences between the controllable text simplification system and large language models on both general and genre-specific tasks at the same time in Section 4.3.1.

In Chapter 5, we proposed a new method to improve the practicality by altering the way of applying control tokens in the controllable text simplification system. We trained regression and classification models to predict the value of control tokens, which improved the SARI score by 0.03 points on the ASSET test set (Alva-Manchego, Martin, Bordes, et al. 2020). While in other datasets like PWKP (Zhu, Bernhard, and Gurevych 2010a) and Turk Corpus (Xu, Napoles, et al. 2016), the increment can change to 3.04 or -0.47 in the SARI score.

Chapter 2

Literature Review

In this chapter, we start on a review through the timeline of NLP development, delving into the history of studies related to text simplification and controlled text simplification. By tracing the evolution of NLP, our aim is to provide a comprehensive overview of the advancements in these areas. Additionally, we explore the resources available for text simplification, encompassing not only corpora but also essential metrics and models used in the field. These resources serve as valuable assets in advancing text simplification research and development. Through an examination of the collective knowledge and tools accumulated over time, our objective is to offer a comprehensive review of the various dimensions of text simplification and its controlled variants.

2.1 Early Exploration in Readability Formulas and Text Simplification

As part of early research in readability field, with its primary objective of aligning reading materials with appropriate readers, the readability formulas has a long history and some of them played a vital role in NLP tasks like simplification and summarisation. It traces back to the pioneering work of Sherman (1893), who employed a statistical approach to analyze readability. His groundbreaking findings revealed a trend of decreasing sentence lengths over time and proposed the notion that concise and concrete expressions enhance readability. The early focus on readability formulas gained momentum in the early 1920s (Zakaluk and Samuels 1988;

DuBay 2004), with Klare et al. (1963) defining readability as ”the ease of understanding or comprehension due to the style of writing.” In the pursuit of predicting readability, researchers developed plenty of readability formulas. Notably, Lively and Pressey (1923) introduced the first-ever readability formula. Over the years, more than 200 formulas emerged, with prominent examples including the Flesch-Kincaid Grade Level (FKGL) (Flesch 1948), Gunning Fog Index (Robert 1952), Coleman Liau Index (Coleman and Liau 1975), and SMOG Index (Mc Laughlin 1969). During this period, these formulas gained some acceptance as potential tools for assessing the readability of public documents and even as criteria for publication (Bruce, Rubin, and Starr 1981).

Despite the early enthusiasm for readability formulas, several shortcomings hindered their ability to fulfil their intended purpose of accurately predicting readability. Bruce, Rubin, and Starr (1981) highlights three primary reasons for their limited success. Firstly, many of the initial formulas lacked consideration for crucial factors that influence readability, such as the complexity of concepts, the presence of necessary background knowledge, the number of required inferences, dialect variations, and the degree of discourse coherence. By solely focusing on word difficulty and sentence length, these formulas failed to capture the multidimensional nature of readability. Secondly, the early formulas suffered from a lack of robust statistical grounding. They were often validated on datasets that were not specifically designed for readability assessment, resulting in poor generalization across different user groups. This lack of tailored validation hindered their ability to accurately predict readability across diverse texts and audiences. Lastly, there was a widespread tendency to inappropriately apply readability formulas. Formulas derived from specific books or datasets were indiscriminately employed on unrelated texts or documents, disregarding whether the target audience and readership aligned or not. This disregard for context and audience-specific considerations further undermined the effectiveness and reliability of the formulas. These collective limitations underscore the need for more comprehensive and context-aware approaches to measuring readability, prompting subsequent research to address these shortcomings and develop more robust readability assessment methods.

In addition to the development of readability formulas, there have been various efforts in creating manual simplification guidebooks and manual text simplification systems. Notable contributions in this area include the recommendations put forth by Basic English (Ogden 1930) and the Plain English initiative (Crystal 1987). These initiatives emphasized the use of a limited vocabulary and restricted grammar rules to enhance comprehension. The effectiveness of text revision in improving readability has been confirmed by studies such as L'ALLIER (1981), where readers with lower literacy levels showed higher performance when presented with simplified texts. Similar positive outcomes have been observed in several other investigations (Noordman and Vonk 1992; McNamara et al. 1996; Linderholm et al. 2000).

2.2 Automatic Text simplification

Similar to developement in many other researches in the NLP field, automatic text simplification systems became mainstream and went through the four different paradigms (P. Liu et al. 2023). Text simplification benefits from the development of other NLP tasks and deep learning theories in many different aspects, including tokenization, model design and training paradigm. In this section, we dive into the chronological timeline of these NLP paradigms, tracing their evolution alongside the development of text simplification systems. By examining the historical progression, we gain a comprehensive understanding of the interplay between NLP advancements and the growth of text simplification techniques.

The first paradigm in the evolution of text simplification systems is feature engineering, which involves the development of hand-crafted rule-based approaches. Chandrasekar, Doran, and Bangalore (1996) first constructed a rule-based system for syntactic text simplification, employing an initial analysis of structural representations within sentences and applying rules to identify and simplify specific units. Addressing the issue of ambiguity, Chandrasekar and Srinivas (1997) introduced a parser to enhance the system's accuracy. The introduction of a Lightweight Dependency Analyzer (LDA) enabled the heuristic determination of constituent structures and dependencies between constituents in both the original and simplified sentences. Building upon these advancements, Carroll et al. (1998) proposed the first lexical simplification system, consisting of an analyzer and a simplifier. The analyzer provided syntactic analysis, while the simplifier adjusted the output to improve readability. Expanding on this work, Siddharthan (2006) decomposed the task into three stages and incorporated their rules during the generation stage.

In the second paradigm, known as architectural engineering, the advancement in computational power and the introduction of recurrent neural networks (RNN) have facilitated the application of various neural network structures in various NLP tasks. Simultaneously, the evolution of tokenization strategies has also played a significant role in driving related research forward. The early tokenization strategy, Bag-of-Words (BOW), was first presented by Harris (1954). However, this approach faces challenges with large dictionaries due to the issue of sparsity. To overcome this limitation, researchers have proposed novel word embedding techniques, such as Word2vec (Mikolov et al. 2013) and GloVe (Pennington, Socher, and Manning 2014). Leveraging these techniques, Nisioi et al. (2017) introduced the first sequence-to-sequence Neural Text Simplification (NTS) method, utilizing a two-layered long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997) network provided by the OpenNMT framework (Klein et al. 2017). Another notable approach is the Deep REinforcement Sentence Simplification (DRESS) proposed by X. Zhang and Lapata (2017a), which combines reinforcement learning architecture with standard encoder-decoder LSTM models. They designed a reward function that takes into account simplicity, relevance, and fluency and employed a reinforcement learning algorithm (Williams 1992) to optimize the model's performance. Expanding on these ideas, Vu et al. (2018) replaced LSTM with Neural Semantic Encoders (NSE) (Munkhdalai and Yu 2017) to capture more contextual information and developed two models based on the main metric. These advancements in architectural engineering have brought about new possibilities for improving text simplification models and enhancing their performance.

The third paradigm, known as pre-train and fine-tuning, emerged with the introduction of the transformer architecture (Vaswani et al. 2017), which revolutionized the field of NLP. This new architecture showed improvements in both performance and computational efficiency, swiftly establishing itself as the dominant approach in NLP research (Yang et al. 2019; Floridi and Chiriatti 2020; Lewis et al. 2020). Consequently, numerous tasks and leaderboards witnessed the emergence of SOTA results (Schwartz et al. 2014; Rajpurkar et al. 2016; Wang et al. 2018). As the field of sequence-to-sequence machine translation flourished, text simplification benefited from the advancements as well (Guo, Pasunuru, and Bansal 2018; Surya et al. 2019; Omelianchuk, Raheja, and Skurzhanskyi 2021). Early on, Zhao et al. (2018) integrated the transformer architecture (Vaswani et al. 2017) with a Paraphrase Database for Simplification (Simple PPDB) (Pavlick and Callison-Burch 2016), achieving the SOTA performance at the time of publication. Building upon this progress, X. Lu et al. (2021b) expanded the training materials by incorporating content from neural machine translation and conducted extensive experiments using various models, including the Bidirectional Auto-Regressive Transformers (BART) (Lewis et al. 2020). Their system, Trans-SS, even surpassed Multilingual unsupervised sentence simplification (MUSS) (Martin, Fan, et al. 2020) in the French language, showcasing the remarkable potential of this paradigm.

The fourth paradigm is the pre-train, prompt, and predict approach. Over the years, there has been a significant increase in the number of parameters used in models for text simplification, starting from LSTM-based models (Hochreiter and Schmidhuber 1997) to transformer-based pre-trained models (Raffel et al. 2019; Lewis et al. 2020). For instance, the BART model has 140 million parameters (Lewis et al. 2020), Text-to-Text Transfer Transformer (T5) has 220 million parameters (Raffel et al. 2020), GPT-3 has a staggering 175 billion parameters (Brown et al. 2020), and the Switch Transformer model takes it even further with an astonishing 1.6 trillion parameters (Fedus, Zoph, and Shazeer 2021). The increasing size of these large language models (LLM) has made it less feasible to follow the traditional pre-train and fine-tune paradigm. As a result, researchers have started exploring the potential of these LLMs with prompt-based approaches and have discovered their zero-shot/few-shot learning capabilities (Brown et al. 2020; Lester, Al-Rfou, and Constant 2021; Thoppilan et al. 2022; X. Liu et al. 2022). For example, J. Lu et al.

(2023) developed the Narrative Prompting and Sentence-matching Summarization (NapSS) system, which consists of a two-stage summarization and simplification process at the paragraph level. In the first stage, they fine-tuned BERT (Devlin et al. 2019) using abstracts and plain English summaries. In the second stage, they combined the generative summary with prompts generated by Stanza (Qi et al. 2020) and further simplified the summary using BART (Lewis et al. 2020). Additionally, Feng et al. (2023) explored the zero-shot or few-shot learning abilities of GPT-3.5 (Brown et al. 2020) and ChatGPT models by using manually crafted prompts, and they achieved remarkable results. These advancements in the pre-train, prompt, and predict paradigm has opened up new possibilities for leveraging large language models in text simplification tasks.

In addition to advancements in the training paradigm, there have been researches focusing on the controllability of system output in text simplification (Scarton and Specia 2018; Martin, Sagot, et al. 2019; Martin, Fan, et al. 2020; Sheang and Saggion 2021). Recognizing the diverse needs of lay users in text simplification, it is challenging for generic outputs to fully satisfy the requirements of the main user group (Xu, Callison-Burch, and Napoles 2015; Stajner 2021). Controlled text simplification has emerged as a solution to address the varied demands of different user groups and different scenarios, where explicit or implicit constraints are imposed on the output. In the targeTS system, Scarton and Specia (2018) introduced targeted grade levels and operations to provide greater control over the output. The AudienCe-CEntric Sentence Simplification (ACCESS) project by Martin et al. (2019) introduced four control tokens to enhance control capabilities. Sheang and Saggion (2021) extended the control tokens to five and replaced the BART model (Lewis et al. 2020) with the T5 model (Raffel et al. 2020), achieving state-of-the-art performance. The combination of performance and flexibility in controlled text simplification opens up possibilities for competing with large pre-trained language models. These advancements in controllability contribute to addressing specific user needs and providing tailored text simplification solutions.

2.3 Corpora for Text Simplification

Corpora, being the fundamental of language modelling, assume a vital role in linguistics and natural language processing. Their significance extends beyond these fields, encompassing language analysis, information retrieval, and various other applications. In this section, we present a comprehensive overview of the prevalent resources employed in English text simplification, diving into their details. By exploring these commonly utilized resources, we aim to review previous corpora associated with text simplification.

Simple English Wikipedia: From a strict standpoint, simple English Wikipedia (SEW) is not a typical parallel corpus for text simplification. Nonetheless, this platform is still a valuable resource for text simplification as it encompasses a vast array of articles crafted to satisfy the specific needs of students, children, adults facing learning challenges, and individuals seeking to enhance their proficiency in the English language. By mirroring the content found in the conventional English Wikipedia while employing a more accessible lexicon and employing simpler grammatical structures, the SEW emerges as a fitting resource for constructing a corpus tailored to the broader domain of general text simplification. Furthermore, the existence of the SEW has played a significant role in fostering the development and emergence of several other corpora (Zhu, Bernhard, and Gurevych 2010a; Coster and Kauchak 2011; Kauchak 2013; Hwang et al. 2015; Kajiwara and Komachi 2016). These corpora have expanded the available resources and paved the way for advancements in the field of text simplification, enabling researchers to explore innovative approaches and refine techniques for making texts more accessible and comprehensible.

PWKP/WikiSmall: Zhu, Bernhard, and Gurevych (2010a) compiled the Parallel Wikipedia Simplification (PWKP) corpus by amalgamating the content from the SEW and the English Wikipedia (EW). Leveraging the SEW as a valuable resource, PWKP employed the sentence-level term frequency–inverse document frequency (TF-IDF) measure as the underlying mechanism for automatic alignment. With an adjusted threshold, they managed to reach the peak performance in the F1 score. Within this dataset, the researchers embraced a flexible approach, accommodating both 1-to-0 and 1-to-N sentence alignments. The resulting corpus encapsulated a collection of 108,016 parallel sentence pairs, elucidating the transformation from complex to simplified language, extracted from 65,133 articles spanning the SEW and EW. Subsequently, X. Zhang and Lapata (2017a) unveiled an enhanced and standardized rendition of the Wikismall corpus. In their process, they eliminated any redundant sentence pairs, thus ensuring a more refined and coherent dataset. As a result, the training set comprised a total of 89,042 sentence pairs. While the training set underwent modifications, the integrity of the test set was upheld, retaining the original composition of 100 sentence pairs for the purposes of rigorous evaluation and comparison.

Coster and Kauchack Corpus: In the study conducted by Coster and Kauchak (2011), their approach involved aligning paragraphs through the utilization of the TF-IDF cosine similarity metric. Subsequently, the dynamic programming algorithm, as introduced by Barzilay and Elhadad (2003), was employed to determine the most optimal sentence alignment across the paragraphs. This method extended beyond the scope of Wikismall by incorporating contextual information, thereby augmenting the comprehensiveness of the dataset. As a result, an extensive collection of 137,000 sentence pairs was generated, accommodating both 1-to-1 and 1-to-N alignments to facilitate a comprehensive representation of sentence transformations.

EW-SEW: The research conducted by Hwang et al. (2015) introduced a novel approach for aligning sentences, employing word-level semantic similarity based on Wikidictionary as the primary alignment method. The researchers initiated the process by creating a graph utilizing valuable synonym information and the co-occurrence patterns of words and their definitions extracted from Wiktionary. Subsequently, the similarity between words was assessed by considering the number of shared neighbors they possessed. This word-level similarity measure was then combined with a similarity score that considered the dependency structures of the words. In order to determine the most appropriate alignments between the original and simplified sentences, a greedy algorithm was employed, utilizing the calculated overall similarity rate. The algorithm was designed to enforce strict 1-to-1 alignment, ensuring a strict alignment between the original and simplified sentence pairs. The rigorous alignment process facilitated the creation of a dataset boasting an impressive collection of over 390,000 sentence pairs, all automatically aligned through this innovative methodology.

sscorpus: In their work, Kajiwara and Komachi (2016) introduced an unsupervised approach to autonomously construct a monolingual parallel corpus for text simplification. Their methodology revolved around leveraging sentence similarity based on word embeddings. The researchers devised a process where they meticulously computed the maximum similarity score for each word across different sentences and subsequently derived the average similarity value for all words within the target sentence. This innovative technique enabled them to capture the nuanced relationships between sentences. Through their diligent efforts, Kajiwara and Komachi (2016) successfully compiled a collection of 492,993 sentence pairs, obtained from a pool of 126,725 article pairs. It is noteworthy that their approach exclusively employed 1-to-1 alignments.

Turk corpus: In their study, Xu, Napoles, et al. (2016) selected sentences from a subset of the PWKP/WikiSmall (Zhu, Bernhard, and Gurevych 2010a). To ensure the highest level of accuracy and consistency in the simplification process, the researchers engaged the expertise of eight annotators sourced from Amazon Mechanical Turk. These annotators were tasked with manually simplifying the selected sentences, resulting in a collection of 2,350 sentences, each accompanied by eight reference simplifications. To further refine the dataset, the researchers conducted a thorough examination, eliminating sentences with poor simplifications through a manual review process. As a result, they built a final dataset of 2,350 sentences, partitioned into two subsets. The larger subset, consisting of 2,000 sentences, was designated for tuning purposes, enabling fine-tuning and optimization of the simplification models. The remaining 350 sentences were dedicated to the evaluation, facilitating a comprehensive assessment of the models' performance. It is important to note that the Turk corpus presents a notable deviation from previous corpora in terms of its focus on paraphrasing. Unlike its predecessors, this corpus prioritizes the generation of paraphrase-only simplifications, contributing to a different understanding of simplification techniques in the context of paraphrasing.

Newsela: Diverging from the previously discussed corpora, the Newsela corpus stands out as a meticulously crafted text simplification resource comprising 1,911 news articles (Xu, Callison-Burch, and Napoles 2015). The authors identified limitations in terms of alignment accuracy and the level of simplicity in existing resources such as SEW-based automatically aligned corpora. To address these concerns, they undertook the task of creating a manually annotated corpus, rewriting the articles with the expertise of professional editors. Each article within the Newsela corpus is accompanied by four reference simplifications. However, it is important to note that Newsela is primarily accessible to researchers and lacks adequate reference lines tailored to existing models. Despite this limitation, the Newsela corpus serves as a valuable reference for researchers seeking to dive deeper into the complexities of text simplification, especially within the context of news articles.

WikiLarge: In their study, X. Zhang and Lapata (2017a) compiled aligned sentence pairs from multiple resources, including the works by Zhu, Bernhard, and Gurevych (2010a), Kauchak (2013), and Woodsend and Lapata (2011). By aggregating these diverse sources, they created a robust corpus known as Wikilarge, which emerged as a prominent training resource in the field of text simplification. The Wikilarge corpus comprises a collection of 296,402 sentence pairs for the training set. These pairs exhibit various alignment patterns, encompassing 1-to-1, 1-to-N, and N-to-1 alignments. This diverse alignment structure enhances the corpus's utility for training models with a comprehensive understanding of alignment variations. To expand the corpus further, X. Zhang and Lapata (2017a) also integrated complex sentences extracted from the WikiSmall corpus, along with their corresponding simplifications generated by Amazon Mechanical Turk workers, as development and test sets. This integration not only enriched the diversity of the corpus but also provided a unified evaluation benchmark for assessing the performance of simplification models. While the Wikilarge corpus may not be the largest in terms of volume, it has emerged as one of the widely used corpora for training neural sequence-to-sequence models in the field of text simplification.

ASSET: Alva-Manchego, Martin, Bordes, et al. (2020) built the Abstractive Sentence Simplification Evaluation and Tuning dataset (ASSET). To construct this dataset, they adopted the same set of original sentences utilized in the Turk corpus (Xu, Napoles, et al. 2016), employing a similar framework that allowed for manual simplification operations along with reference sentences. The ASSET dataset comprises a development and test set, consisting of 2000 and 350 sentences respectively. However, each sentence has ten manual reference sentences, providing a rich pool of alternative simplifications. These references encompass both 1-to-1 and 1-to-N alignments. The ASSET dataset has been widely adopted as a standard benchmark for evaluating the performance of text simplification models.

Simple TICO-19: Shardlow and Alva-Manchego (2022) constructed simple the TICO-19 corpus designed for text simplification purposes. This corpus was derived from the Translation Initiative for COVID-19 (TICO-19) dataset (Anastasopoulos et al. 2020). To create the simple TICO-19 corpus, the researchers asked annotators to manually simplify the content from TICO-19 and labelled simplification operations and resources for each sentence. These annotators also labelled the simplification operations undertaken for each sentence, providing insights into the simplification process. The resulting Simple TICO-19 corpus boasts a collection of 3,173 parallel sentences, encompassing both English and Spanish translations. The alignments within this corpus showcase a mix of 1-to-1 and 1-to-N alignment patterns. Similar to the Newsela corpus (Xu, Callison-Burch, and Napoles 2015), Simple TICO-19 focuses primarily on the medical domain rather than general domains, with a specific emphasis on COVID-19-related information. It serves as a valuable resource for researchers operating within the medical field, facilitating the exploration of text simplification techniques in this specialized domain.

As highlighted in previous studies (Xu, Callison-Burch, and Napoles 2015; Amancio and Specia 2014), there are limitations associated with automatically aligned corpora sourced from SEW and EW. These drawbacks include inadequate alignment quality and a lack of diversity in the simplification operations employed. Xu, Callison-Burch, and Napoles (2015) specifically observed that these corpora tend to favour deletion, paraphrasing, and hybrid operations over sentence splitting, resulting in simplifications that lean towards compression rather than expansion. Consequently, such corpora are not suitable for learning conceptual simplification, an essential yet frequently overlooked aspect in this field. Furthermore, none of the aforementioned corpora incorporate 1-to-0 alignment. This absence of alignment poses challenges when training text simplification models at the paragraph level. Therefore, relying solely on this kind of corpora for training purposes may not provide an ideal solution in these cases.

2.4 Evaluation Methods for Text Simplification

In this section, we will focus on the popular metrics employed in the field of text simplification, with a particular focus on evaluating the effectiveness of these metrics. The evaluation of text simplification outputs is of paramount importance as it aids in determining the success and impact of various simplification techniques.

Human Evaluation: Human evaluation stands out as the most reliable and indispensable approach when it comes to evaluating text simplification. By involving expert annotators or target users, we can gather subjective judgments and feedback on crucial aspects such as grammaticality, meaning preservation, and simplicity of the simplified texts (Alva-Manchego, Scarton, and Specia 2020).

While automated metrics serve as valuable tools for quantitative analysis, they often fail to capture the intricacies of language and the subjective experience of readers. Human evaluation, on the other hand, offers a comprehensive and nuanced assessment of text simplification outputs. By combining the strengths of both automated metrics and human evaluation, we can achieve a more accurate evaluation of text simplification, leading to improved accessibility and understanding for a wide range of readers.

FKGL: Flesch-Kincaid Grade Level (FKGL) stands as one of the legacy metrics that was originally developed to assess the readability of text and align it with the appropriate grade level of readers (Flesch 1948). The FKGL formula incorporates several factors, including the average number of syllables per word, the average number of words per sentence, and the overall number of sentences in a text, and is shown in Equiation 2.1.

$$FKGL = 0.39\frac{N_{words}}{N_{sentences}} + 11.8\frac{N_{syllables}}{N_{words}} - 15.59 \tag{2.1}$$

Although initially designed to evaluate the readability of human-generated text, FKGL has also been introduced as a metric for automatic text simplification by Zhu, Bernhard, and Gurevych (2010b), gaining traction in subsequent studies. However, the simplicity of the FKGL formula makes it susceptible to manipulation. Tanprasert and Kauchak (2021) demonstrated that FKGL can be easily manipulated by randomly replacing words with a period or employing other similar techniques, while other metrics remain largely unaffected. This inherent vulnerability raises concerns about the reliability and robustness of FKGL as a metric for evaluating text simplification. Moreover, studies have revealed a low correlation between FKGL scores and the quality of simplifications produced (Martin, Humeau, et al. 2019; Alva-Manchego, Scarton, and Specia 2020). This discrepancy indicates that FKGL may not adequately capture the nuances of text simplification, limiting its usefulness in accurately assessing the effectiveness of simplification techniques. Consequently, researchers have increasingly turned to alternative metrics that provide more comprehensive and accurate evaluations. Given these shortcomings and limitations, the reliability and effectiveness of FKGL is questionable.

BLEU: Bilingual evaluation understudy (BLEU) emerges as one of the widely used metrics for evaluating text-to-text generation tasks, including text simplification. Initially designed to assess the quality of output text against multiple references, BLEU offers a robust framework for comparison (Papineni et al. 2002). The metric is defined by incorporating various components: the brevity penalty (BP), modified precision (p_n), and corresponding weights (w_n). The BP is calculated as in Equation 2.2, where c is the length of the candidate translation, r is the reference corpus length, and r/c is used in a decaying exponential (in this case, c is the total length of the candidate translation corpus).

$$BP = \begin{cases} 1 & \text{if c > r} \\ e^{(1-r/c)} & \text{if c} \leq \text{r} \end{cases} \tag{2.2}$$

The final BLEU scores is calculated in Equation 2.3.

$$BLEU = BP \cdot \exp \sum_{n=1}^{N} w_n \log(p_n) \tag{2.3}$$

In the machine translation tasks, BLEU has demonstrated a positive correlation with human judges, indicating its effectiveness as an evaluation metric (Papineni et al. 2002). However, Callison-Burch, Osborne, and Koehn (2006) highlighted that BLEU may show a poor correlation when dealing with a vast number of translations and low-quality references. In the context of text simplification, Xu, Napoles, et al. (2016) discovered that BLEU exhibits a stronger correlation with meaning preservation rather than simplicity. Moreover, Sulem, Abend, and Rappoport (2018) provided further evidence that BLEU struggles to accurately reflect the effectiveness of sentence splitting operations and sometimes even negatively correlates with simplicity.

SARI: System output Against Reference and Input (SARI) is an operation-specific metric for evaluating the performance on the simplicity gained through text simplification system (Xu, Napoles, et al. 2016). It offers a comprehensive evaluation by comparing the system output to both the reference and input sentences and considering the *add*, *keep*, and *delete* operations. The final SARI score is calculated in Equation 2.4

$$SARI = d_1 F_{\text{add}} + d_2 F_{\text{keep}} + d_3 F_{\text{del}} \tag{2.4}$$

where

$$d_1 = d_2 = d_3 = 1/3 \tag{2.5}$$

and

$$\begin{aligned} P_{\text{operation}} &= \frac{1}{k}\sum_{n=1}^{k} \text{p}_{\text{operation}}(n) \quad R_{\text{operation}} = \frac{1}{k}\sum_{n=1}^{k} \text{r}_{\text{operation}}(n) \\ F_{\text{operation}} &= \frac{2 \times P_{\text{operation}} \times R_{\text{operation}}}{P_{\text{operation}} + R_{\text{operation}}} \quad \text{operation} \in \{\text{del}, \text{keep}, \text{add}\} \end{aligned} \tag{2.6}$$

In Equation 2.4, the different operations have different algorithm, which take into account of output (O), the input sentence (I), references (R), and the binary indicator for the occurrence of n-grams g in a given set ($\#_g(\cdot)$). The n-gram precision $p(n)$ and recall $r(n)$ are calculated in the following equations:

$$\begin{aligned} p_{\text{add}}(n) &= \frac{\sum_{g\in O} \min(\#g(O\cap \overline{I}), \#g(R))}{\sum_{g\in O} \#g(O\cap \overline{I})}, \quad \#g(O\cap \overline{I}) = \max(\#g(O) - \#g(I), 0); \\ r_{\text{add}}(n) &= \frac{\sum_{g\in O} \min(\#g(O\cap \overline{I}), \#g(R))}{\sum_{g\in O} \#g(R\cap \overline{I})}, \quad \#g(R\cap \overline{I}) = \max(\#g(R) - \#g(I), 0), \\ p_{\text{keep}}(n) &= \frac{\sum_{g\in I} \min(\#g(I\cap O), \#g(I\cap R_0))}{\sum_{g\in I} \#g(I\cap O)}, \quad \#g(I\cap O) = \min(\#g(I), \#g(O)), \\ r_{\text{keep}}(n) &= \frac{\sum_{g\in I} \min(\#g(I\cap O), \#g(I\cap R_0))}{\sum_{g\in I} \#g(I\cap R_0)}, \quad \#g(I\cap R_0) = \min(\#g(I), \#g(R)/r), \\ p_{\text{del}}(n) &= \frac{\sum_{g\in I} \min(\#g(I\cap \overline{O}), \#g(I\cap R_0))}{\sum_{g\in I} \#g(I\cap \overline{O})}, \quad \#g(I\cap \overline{O}) = \max(\#g(I) - \#g(O), 0), \\ & \qquad \#g(I\cap R_0) = \max(\#g(I) - \frac{\#g(R)}{r}, 0). \end{aligned} \tag{2.7}$$

Compared to BLEU (Papineni et al. 2002), it compares the output against the reference and input sentences at the same time and takes the same operation in the reference sentences into account rather than all n-gram matches. However, there are significant limitations to the SARI score. It can only partially reflect level the of syntactical simplification in the output, nor the fluency and grammar in the sentences. In addition, it highly relies on the diversity and quality of reference sentences. A recent study also showed the possible poor correlation for evaluation systems with SARI score only and gave a suggested pattern to automatic evaluations (Alva-Manchego, Scarton, and Specia 2021).

BERTScore: The BERTScore is a metric used for evaluating the similarity between the system output and reference sentences (T. Zhang et al. 2019). This metric employs cosine similarity to measure the distance in contextual embeddings created by BERT (Devlin et al. 2019) in a likelihood matrix, aiming to maximize the similarity between the output and reference sentences. BERTScore comprises two components: $BERTScore_{precision}$, which matches the system output with the reference, and $BERTScore_{recall}$, which compares the reference against the system output. The two parts are then combined in $BERTScore_{F1}$. The equation is shown in 2.8, where x and $\hat{x}$ are the references and candidates.

$$\begin{aligned} BERTScore_{recall} &= \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} x_i^\top \hat{x}_j, \\ BERTScore_{precision} &= \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} x_i^\top \hat{x}_j, \\ BERTScore_{F1} &= \frac{2 \cdot BERTScore_{precision} \cdot BERTScore_{recall}}{BERTScore_{precision} + BERTScore_{recall}} \end{aligned} \tag{2.8}$$

Notably, BERTScore has been found to have a stronger correlation with human evaluation compared to SARI (Scialom et al. 2021). It focuses on capturing the meaning rather than just lexical paraphrasing, emphasizing the similarity in terms of semantic content. To enhance the assessment of text quality, researchers like Alva-Manchego, Scarton, and Specia (2021) have recommended combining BERTScore with other automatic evaluation metrics.

2.5 Level of literacy

In this section, we aim to provide an overview of the different levels of literacy and highlight the specific requirements and challenges associated with each level, particularly in relation to text simplification. According to the classification by OECD (2013), literacy is categorized into five distinct levels. These levels reflect varying degrees of reading and comprehension abilities, and individuals at different levels may encounter different types of difficulties when engaging with written material. Understanding the unique needs and limitations of individuals at each literacy level is crucial for developing effective controllable text simplification strategies. By tailoring simplification techniques to address the specific challenges faced by individuals with lower literacy levels, we can significantly enhance their reading experience and promote better comprehension.

Furthermore, recognizing the diversity within literacy levels allows us to adapt our approach to suit different contexts and purposes. For instance, individuals at higher literacy levels may benefit from more nuanced simplification techniques that preserve the complexity and depth of the original text, while those at lower literacy levels may require more fundamental simplification methods that prioritize clarity and accessibility. By acknowledging the range of literacy levels and the corresponding challenges, we can better design and implement text simplification approaches that cater to the needs of diverse readerships. This understanding enables us to foster greater inclusivity and improve the accessibility of information for individuals across the literacy spectrum.

Below level 1: Individuals at this level can only comprehend short texts about familiar subjects and find one particular piece of information that matches the information given in a question or instruction. They are not expected to grasp the organization of sentences or paragraphs, and only a basic understanding of vocabulary is necessary. Activities below Level 1 do not involve utilizing any digital text-related features.

Level 1: Individuals at this level are able to read concise digital or printed texts, whether they are continuous, non-continuous, or a combination of both, in order to find a specific piece of information that matches or has the same meaning as the information provided in a question or instruction. These texts do not contain much conflicting information. Individuals performing at this level can fill out uncomplicated forms, comprehend fundamental vocabulary, decipher the meaning of sentences, and read continuous texts with a certain level of ease and fluency.

Level 2: Individuals at this level are able to combine multiple pieces of information using specific criteria, analyze and highlight similarities and differences, and engage in reasoning to make basic inferences. They can proficiently navigate through digital texts, locating and extracting information from different sections of a document.

Level 3: Individuals at this level can comprehend and react suitably to complex or extensive texts, encompassing continuous, non-continuous, mixed, or multiple-page formats. They possess an understanding of text structures and rhetorical techniques, enabling them to recognize, interpret, or assess multiple pieces of information and draw relevant inferences. Moreover, they can execute multi-step operations and discern pertinent data from conflicting information in order to identify and formulate well-considered responses.

Level 4: Individuals at this level have the capability to execute multi-step tasks that involve integrating, interpreting, or synthesizing information from intricate or lengthy texts of various types. These texts may be continuous, non-continuous, mixed, or encompass different formats. The information within these texts can contain conditional or conflicting details. Individuals can make sophisticated inferences and effectively utilize their background knowledge to appropriately apply it in these tasks. They are also adept at interpreting and evaluating nuanced truth claims or arguments.

Level 5: Individuals at this level have the ability to engage in various tasks that require them to search for and combine information from multiple intricate texts. They can create comprehensive summaries of similar and different ideas or perspectives, as well as assess evidence and arguments. They possess the skills to employ and assess logical and conceptual frameworks, determine the credibility of sources, and identify crucial information. Additionally, they possess an awareness of subtle rhetorical indicators and can draw advanced conclusions or utilize specialized knowledge.

Given the diverse literacy levels among users, it becomes imperative to adopt a tailored approach to text simplification that caters to their specific needs (Stajner 2021). Depending on an individual's literacy level, different aspects of simplification come into focus. For individuals below level 4, emphasis should be placed on conceptual simplification, which involves presenting information in a manner that is easily grasped and understood. This level of simplification ensures that complex ideas are broken down into simpler, more digestible concepts.

For users below level 3, syntactical simplification assumes greater importance. This form of simplification involves modifying the structure and arrangement of sentences to enhance clarity and comprehension. By reducing complexity in sentence construction, individuals at this literacy level can more effectively navigate and understand written material.

In the case of individuals below level 2, the primary demand lies in lexical simplification. This involves replacing or rephrasing complex words and expressions with simpler alternatives that are more familiar and accessible. By employing this approach, individuals with limited vocabulary and language skills can better engage with and comprehend the text.

Understanding the varying needs and challenges at different literacy levels allows us to develop targeted strategies for text simplification. By addressing these specific requirements, we can enhance the accessibility and inclusivity of written content, enabling individuals across all literacy levels to engage meaningfully with information and knowledge.

2.6 Review Summary

In this chapter, we explored the development history of text simplification, sorted out the popular corpus and evaluation methods used in text simplification task and analysed the different needs for people with different literacy levels. With all these reviews and analysis, the limitation of current text simplification has been revealed that most text simplification systems regard the task as an universal problem and few of them can satisfy the demands of different user groups.

However, with the assistance of the controllable text simplification technique adopted in the following chapters, the researchers can adjust the level and focus of lexical and syntactical simplifications freely to help satisfy the various demands of different user groups. Yet, for the conceptual simplifications, there remain few studies in the related domains.

Chapter 3

Investigation of Control Token

In this chapter, we focus on the first research question of how the control mechanism works by exploring the principles of control tokens and evaluating how they affect simplification. We made a thorough investigation of each control token separately about how they affect the SARI score, BERTScore, and the output sentence and learned the mechanism and limitations of each control token. At last, we re-designed the tokenization strategy to apply the control tokens and improved the SARI score.

3.1 Methodology

In this section, we demonstrate the methodology in figure 3.1 and illustrate the 4 main steps shown in the figure: preprocessing, fine-tuning, optimisation and evaluation.

3.1.1 Preprocessing

The preprocessing step followed the MUSS project (Martin, Fan, et al. 2020). The authors defined four types of prompts used as control tokens to manipulate the features of the outputs. Each control token is designed to represent one character of the sentence. The <DEPENDENCYTREEDEPTHRATIO_x> represents the syntactic complexity; The <WORDRANKRATIO_x> represents the lexical complexity; The <REPLACEONLYLEVENSHTEINRATIO_x> represents the inverse sim-

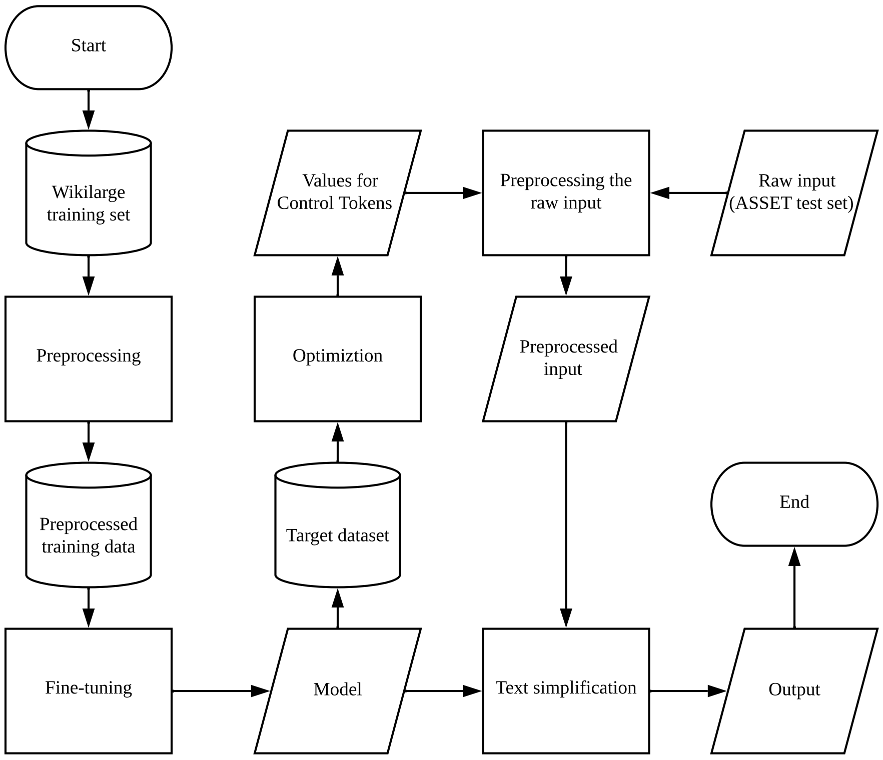


Figure 3.1. The reimplementation methodology is represented as a flow chart. We fine-tune BART base on the preprocessed WikiLarge training set so that the model learns to simplify under the control token or control tokens. After optimisation on the target dataset, the model can apply the optimal value of control tokens to the input and generate desired simplifications.

ilarity of input and output at the letter level; The <LENGTHRATIO_x> represents the length ratio of input and output. The value of each control token is calculated based on the reference complex-simple pairs in the training set, which is Wikilarge in this project (X. Zhang and Lapata 2017b). After the calculation, these control tokens will be added to the beginning of complex sentences, and the model will be trained on this preprocessed dataset. In addition to the combined control tokens, this project also explored the effects of a single control token; only the corresponding control tokens are kept in that dataset.

3.1.2 Fine-truning

Moving on to the training phase, we adopted a methodology that aligns with the conventional fine-tuning procedures applied to pretrained language models. The process involved providing the model with preprocessed sentence pairs, encompassing complex and simplified versions of the text. Through this training process, the model was expected to acquire the ability to simplify text based on the guidance provided by control tokens. In order to investigate the impact of different tokenization strategies and the individual influence of specific control tokens, a total of 16 models were fine-tuned during the experiment. This comprised a baseline model, three models incorporating all control tokens, and twelve models featuring only one control token. The inclusion of models with single control tokens enabled us to evaluate the significance of utilizing combined control tokens in achieving desired simplification outcomes. This comprehensive approach aimed to shed light on the interplay between control tokens and their cumulative effects on text simplification.

3.1.3 Optimisation

The following step is optimisation. As mentioned in previous sections, the value of control tokens is limited to a small range. All options fall between 0.2 to 1.5 except the Levenshtein, whose upper boundary is limited to 1 due to the calculation method that divides the minimum replacement steps to change from the original sentence to the target sentence by the maximum possible steps of replacement. Only these options are provided during optimisation, and the optimisation problem is reduced to finding the best value combination of control tokens within the range. Even though only finite combinations can be applied to the model, the optimisation algorithm is still supported by the Nevergrad (Rapin and Teytaud 2018) API to compare with the current SOTA. Within the budget of 64 times, which serves as a limitation to repeat the optimisation process, the algorithm can find a relatively optimised result in all 334,611 combinations of control token values. Although we have half the optimisation budget compared to MUSS, the system still manages to achieve better performance than MUSS. A sample result of SARI in the optimisation procedure is shown in 3.4. In order to ensure the reliability of the score under the optimised combination, a bootstrapping on the ASSET (Alva-Manchego, Martin, Bordes, et al. 2020) test dataset will be executed by resampling the dataset 200 times and hence generate a 95% confidence interval.

3.1.4 Evaluation

The last step is evaluation. We leveraged the Easier Automatic Sentence Simplification Evaluation(EASSE), which is a open-source code repository as an integration of popular metrics and test sets, to apply multiple evaluation metrics at the same time easily (Alva-Manchego, Martin, Scarton, et al. 2019). The SARI score is adopted as the primary metric to compare with the current SOTA, while the BERT score is added as a second reference. Different from the common applications in other projects, the BERT score in this project is the correlation between the output and references. One coefficient array can be used to combine different evaluation metrics and give a weighted score. However, in this project, we also follow the operations in MUSS and maximise the SARI score, so only the SARI score is taken into account, and the corresponding coefficient is set to 1. The models will be evaluated on the ASSET (Alva-Manchego, Martin, Bordes, et al. 2020) test dataset, which contains 359 complex-simple pairs, and each complex sentence has ten reference simplifications.

3.1.5 Control Token Definition

Following the settings from MUSS, we set 4 control tokens in the following form: <DEPENDENCYTREEDEPTHRATIO_*x*> (DTD), <WORDRANKRATIO_*x*> (WR), <REPLACEONLYLEVENSHTEINRATIO_*x*> (LV) and <LENGTHRATIO_*x*> (LR) (Martin, Fan, et al. 2020) and they are calculated based on following equations:

$$DTD = \frac{SimpleDependencyTreeDepth}{ComplexDependencyTreeDepth} \tag{3.1}$$

The depth of the dependency tree of sentences is generated by spaCy(Honnibal et al. 2020) and used to represent the syntactical complexity. The ratio is calculated by dividing the highest depth of the dependency tree of the simple sentence by the depth of the corresponding complex sentence.

$$WR = \frac{SimpleWordRank}{ComplexWordRank} \tag{3.2}$$

The word rank of sentences is determined by word rank of the 4^{th} quantile word of all words in the sentence in the ascending order. The word rank is then determined by the word frequency in the fasttext (Grave et al. 2018) and used to represent the lexical complexity. The ratio is calculated by dividing the word rank of the simple sentence by the word rank of the corresponding complex sentence.

$$LV = 1 - \frac{ReplaceonlyLevenshitineDistance}{max(SimpleLength, ComplexLength)} \tag{3.3}$$

The Levenshitine distance is calculated by the steps to change from the simple sentence to the complex sentence by replacing the characters. The ratio is calculated by one minus the quotient of Levenshitine distance and the max length of simple and complex sentences and used to represent the similarity of the simple sentence to the complex sentence.

$$LR = \frac{SimpleLength}{ComplexLength} \tag{3.4}$$

The length is the word count of specified sentences. The ratio is calculated by dividing the word count of the simple sentence by the word count of the complex sentence and used to represent the difference in length. The value of the above-mentioned control tokens is calculated in the preprocessing step and rounded to the nearest 0.05.

3.2 Experiment design

In this section, we will focus on the detailed settings and changes in this project compared to the original ACCESS(Martin, Sagot, et al. 2019).

3.2.1 Tokenization Strategies

Strategy	Raw Input	’<DEPENDENCYTREEDEPTHRATIO_0.6>’
Default	IDs	[0, 41552, 41372, 9309, 23451, ..., 2571, 6454, 1215, 288, 4, ...]
	tokenization	[’<s>’, ’<’, ’DEP’, ’END’, ’ENCY’, ..., ’_’, ’0’, ’.’, ’6’, ’>’, ...]
Joint	IDs	[0, 50265, ...]
	tokenization	[’<s>’, ’<DEPENDENCYTREEDEPTHRATIO_0.6>’, ...]
Separate	IDs	[0, 50265, 50266, 15698, ...]
	tokenization	[’<s>’, ’<DEPENDENCYTREEDEPTHRATIO_’, ’0.6’, ’>’, ...]

Table 3.1. Tokenization under differing strategies for the input starting with: ’<DEPENDENCYTREEDEPTHRATIO_0.6>’

The first main difference is that we introduced two new tokenization methods to explore the effects of tokenization strategies. As shown in Table 3.1, the default tokenization method in the MUSS project is regarding the control tokens as plain text. In comparison, we added 2 more tokenization strategies: The Joint tokenization strategy is to regard the whole control token as one token in the tokenizer; the Separate tokenization strategy is to break the control token into a combination of type and value and add them separately to the tokenizer. These 2 strategies are achieved by manually adding all possible control tokens to the dictionary of the tokenizer. This will affect not only the evaluation and optimisation process but also the training process, thus each tokenization strategy requires an independent fine-tuned model.

3.2.2 Quantisation Strategy

The second main difference is that we changed the quantisation strategy for the control tokens in the optimisation step and the corresponding algorithm. In the pre-processing step, there are 4 control tokens calculated and added to the beginning of complex sentences in the complex dataset. As mentioned in the literature review, the 4 types of control tokens are <DEPENDENCYTREEDEPTHRATIO_*x*> (DTD), <WORDRANKRATIO_*x*> (WR), <REPLACEONLYLEVENSHTEINRATIO_*x*> (LV) and <LENGTHRATIO_*x*> (LR). As an augmentation to the control tokens, the calculated values are rounded to the nearest 0.05. However, in the original optimisation process, the calculated values by the algorithm provided by the Nevergrad (Rapin and Teytaud 2018) API are continuous and have long verbose digits, 0.249452…for example. During the reimplementation, we found that only the first one or two digits are recognised as input values and the remaining digits didn't provide any meaningful instruction. On the contrary, it might bring unnecessary information to the system and even lowered the performance of the model. As a comparison, we replaced the continuous values with discrete ones like 0.2, 0.25, 0.3, ..., 1.0 and changed to the corresponding discrete algorithm in Nevergrad (Rapin and Teytaud 2018). The results are shown in Table 3.3.

3.2.3 Reimplementation of ACCESS

One of the goals of this project is to reimplement and verify the effect of control tokens in the current SOTA. However, since the main focus of this project is on the control tokens, instead of training on both supervised and unsupervised datasets, it would be more practical to claim the reimplementation of ACCESS rather than MUSS. In order to build a unified baseline, this project also applied the BART model (Lewis et al. 2020), which is adopted in the MUSS project. The original project can be divided into the following sections: data mining, preprocessing, training, evaluation and optimisation.

Since the goal is verification, there is no need to rewrite the code for all sections. Thus only the codes related to training and some other peripheral functions have been altered to achieve similar results. The other functions, such as preprocessing and optimisation, still kept most of the original code. The original core API used for training is fairseq. This project replaced it with another open-source API — Huggingface. Huggingface provides a collection of the most popular pre-trained models and datasets, including the BART (Lewis et al. 2020) and a unified, advanced and user-friendly API to achieve the most common applications, which made it easier for future upgrading and modification. The hyper-parameters of models in the reimplementation, including the learning rate and weight decay, are set to be identical to the original project so that the influence of irrelevant factors can be lowered. The last difference between the reimplementation and the original project is the tokeniser. The tokeniser in the reimplementation is the BART-base byte-pair encoding(BPE) tokeniser instead of the GPT2 BPE tokeniser (Radford et al. 2019). Both tokenisers serve the same purpose and perform very similarly to each other. The new one consumes fewer computer resources, which presumably causes only a little effect on the results. Due to the variation of control tokens, the optimisation algorithm has also changed. The original algorithm is the OneplusOne provided by Nevergrad (Rapin and Teytaud 2018), and the current one is the PortfolioDiscreteOnePlusOne, which fits the discrete values better. As for the metrics, the SARI score is kept as the primary evaluation method (Xu, Napoles, et al. 2016), and the BERT score is introduced as a co-reference.

However, due to the limitation of computation resources and mass fine-tuning demands of models with different tokenization strategies, this project also downgraded the training scale and limited the epochs in both baseline and reimplementation. Here are the changes applied to both the reimplementation and the baseline as follows:

- All results are from models trained in BART-base instead of BART-large.
- All training processes are set to 10 epochs only.
- All models are trained on Wikilarge (X. Zhang and Lapata 2017b) only.

As explained earlier, each tokenization strategies is corresponding to one model and there is a total of 16 models that need to be fine-tuned. This is why only BART-base is applied and the training epochs are limited. As for the reason for choosing 10 as the targeting epoch number, it is because the training loss for models with combined control tokens has reached 0.85 and decreased very slowly between epochs, while the validation loss started increasing. If continuing training, the overfitting problem may occur. The results of the baseline shown in the next section can also partially prove the training process is long enough.

3.3 Results

We first report the SARI score of influential supervised text simplification models with our reimplementations in Table 3.2. Although MUSS (with mined data) (Martin, Fan, et al. 2020) is slightly lower than our reimplementation, our reimplementation stays within the 95% confidence interval of MUSS (with mined data). To verify the significance of the difference in the SARI score, we conducted significance studies against the official output of MUSS (without mined data) with a student's t-test of the SARI score of the two groups and reported the p-value for the bottom four models. As shown in the table, our reimplementation required fewer resources and training data, while maintaining a significant difference. In addition, we propose the prediction method rather than optimisation on control tokens and get a higher BERTScore in section 5. The bottom two methods in Table 3.2 are described in Section 5.

Model	SARI score ↑
EditNTS (Dong et al. 2019)	34.95
Dress-LS (X. Zhang and Lapata 2017b)	36.59
DMASS-DCSS (Zhao et al. 2018)	38.67
ACCESS (Martin, Sagot, et al. 2019)	40.13
TST (Omelianchuk, Raheja, and Skurzhanskyi 2021)	43.21
MUSS (without mined data) (Martin, Fan, et al. 2020)	43.63
MUSS (with mined data) (Martin, Fan, et al. 2020)	44.15 (p=0.059)
Our Reimplementation	44.65 (p=0.033)
Our Reimplementation (with only predictors)	42.30 (p=0.133)
Hybrid Method	44.61 (p=0.056)

Table 3.2. The SARI score of supervised text simplification systems (p in the brackets refers to the p-value of the SARI score against the MUSS (without mined data)).

3.3.1 Combined performance

With similar BART-base and 10 epochs limitations, the baseline is derived from the original code of the MUSS and achieved 43.83 in SARI score on the ASSET test set(Alva-Manchego, Martin, Bordes, et al. 2020), which is consistent with the reported score in the MUSS without minded paraphrasing data, which is 43.63±0.71. The reason we compare this model is that this is the test scenario closest to our reimplementation. There is no confidence interval and BERTScore in the base-

Prompts	SARI↑	BERT↑	DTD	WR	LV	LR
Baseline	43.83	—	0.249452…	0.814345…	0.758764…	0.858546…
Default	44.00±0.05 (p=0.620)	0.754	0.25	0.8	0.75	0.85
Joint to-kens	44.02±0.05 (p=0.700)	0.769	0.25	0.8	0.75	0.85
Separate tokens	44.04±0.05 (p=0.779)	0.754	0.25	0.8	0.75	0.85

Table 3.3. Results on SARI and BERTScore under differing tokenization strategies, with comparison to the baseline (p in the brackets refers to the p-value of the SARI score against the baseline).

Optimisa-tion target	Prompts	SARI↑	BERT↑	DTD	WR	LV	LR
Validation set	Default	44.31±0.05 (p=0.628)	0.759	0.4	0.6	0.8	1.1
	Joint tokens	44.58±0.05 (p=0.885)	0.814	0.35	0.7	0.85	0.9
	Separate tokens	44.65±0.05 (p=0.701)	0.783	0.35	0.75	0.75	0.95
Test set	Default	44.36±0.05 (p=0.931)	0.733	0.6	0.7	0.65	0.85
	Joint tokens	44.67±0.05 (p=0.313)	0.786	0.35	0.75	0.75	0.9
	Separate tokens	44.75±0.05 (p=0.321)	0.784	0.35	0.75	0.75	0.9
N/A	Default	43.34±0.06	0.827	0.6	0.85	0.85	0.85
	Joint tokens	43.83±0.06	0.829	0.6	0.85	0.85	0.85
	Separate tokens	43.99±0.06	0.828	0.6	0.85	0.85	0.85

Table 3.4. SARI and BERTScore on the ASSET test set under different optimisation targets (p in the brackets refers to the p-value of the SARI score against the baseline). The first three rows optimise on the validation set, the middle three rows optimise on the test set and the bottom three rows show the performance under average value of control tokens.

line because the baseline is generated by rerunning the code in MUSS by altering specific settings only. The actual output lacks these 2 features. As shown in the 4 rows in Table 3.3, the SARI score with 95% confidence in the reimplementation is slightly higher than the baseline. However, significance testing shows that the improvement in score according to the different tokenization strategies is not significantly different from the baseline result for any of the cases we examined. This implies that the tokenization strategy does not affect control token performance.

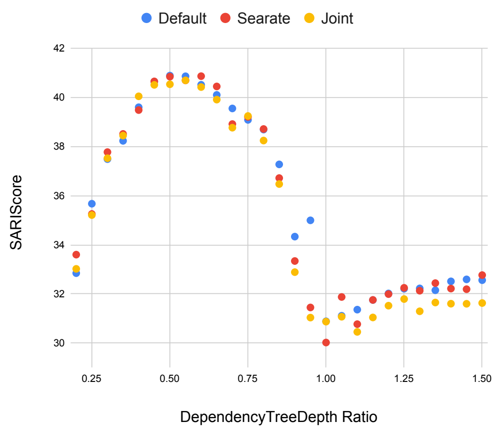
Different from the optimisation target in MUSS, we tried optimising control tokens on both the validation and test set of ASSET to find out the peak performance of the system and the results are shown in Table 3.4. The top 3 rows show the best SARI score with optimised options of control tokens optimised on the validation set. With optimised control tokens on the validation set, the separate tokenization strategy achieved the highest score within the optimisation budgets, while the joint tokenization method has the highest BERTScore. The middle 3 rows are the results directly optimised on the test set, which shows the upper limit of the model. Among the 3 methods, the separate tokenization strategy had the highest SARI score. Interestingly, the BERTScore is not always proportional to the SARI score, but the BERTScore of optimal value is still quite high. The optimised values of control tokens are pretty close in all situations except the DTD. Similar to the results in Table 3.3, the p-values show no significant difference between the baseline and the reimplementations. We continue to record the results for each of our three tokenization strategies, but significance testing again shows that changes in the tokenization strategy have not led to gains which are significantly improved from the baseline in these cases. The bottom 3 rows show the performance difference under a unified value of control tokens. The unified value is the average value of all possible values for each control token. Under the unified condition, the separated one outperformed the other two, and the default tokenization method still performs worse. As for the BERTScore, the joint tokenization method still outperforms the other two.

3.3.2 Effects of single control tokens

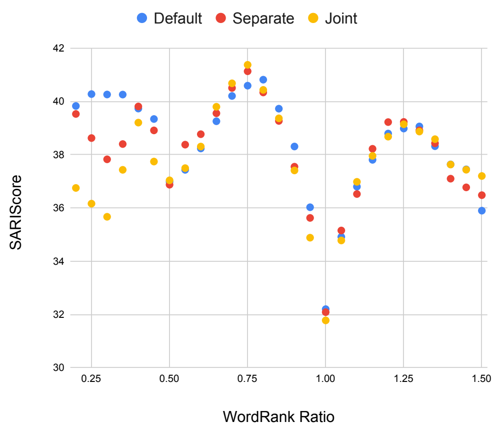
Control Token	Value	SARI_add	SARI_keep	SARI_del	SARI↑
DTD_joint	0.2	2.71	27.03	69.32	33.02
	0.6	5.24	58.50	57.51	40.41
	1.0	3.30	62.64	26.68	30.87
	1.5	4.41	62.66	27.82	31.63
WR_joint	0.5	5.10	37.47	68.54	37.04
	0.75	6.65	54.91	62.57	41.37
	1.0	3.38	62.04	29.90	31.77
	1.25	4.19	54.88	58.35	39.14
LV_joint	0.2	7.15	50.83	63.83	40.60
	0.7	9.14	60.15	57.60	42.30
	1.0	2.25	61.62	32.17	32.01
LR_joint	0.2	1.80	19.27	69.46	30.18
	0.65	5.54	56.84	59.36	40.56
	1.0	2.43	62.42	15.26	26.70
	1.2	5.80	61.46	26.03	31.10

Table 3.5. SARI score by operation at turning points in Figure 3.2.

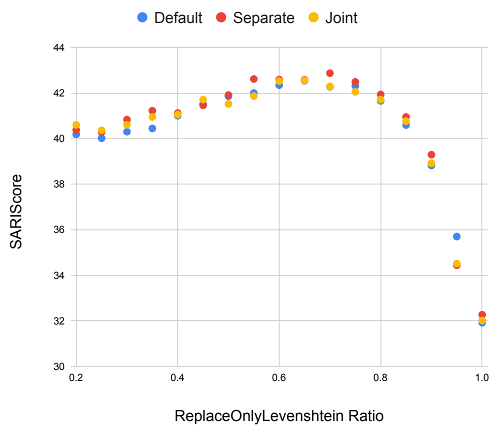
In order to verify the effects of each single control token, a more detailed investigation of the SARI score was done on control tokens respectively and the results are shown in Figure 3.2. Except for Figure 3.2b, all 3 tokenization methods show a



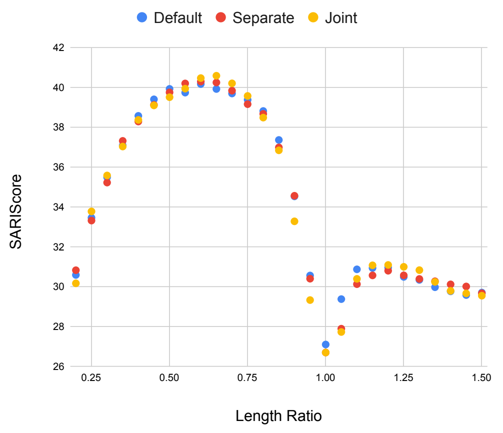
(a) SARI score of models with the DependencyTreeDepth ratio only



(b) SARI score of models with the WordRank ratio only



(c) SARI score of models with the ReplaceOnlyLevenshtein ratio only



(d) SARI score of models with the Length ratio only

Figure 3.2. The effect of varying control tokens under different tokenization strategies on SARI Score.

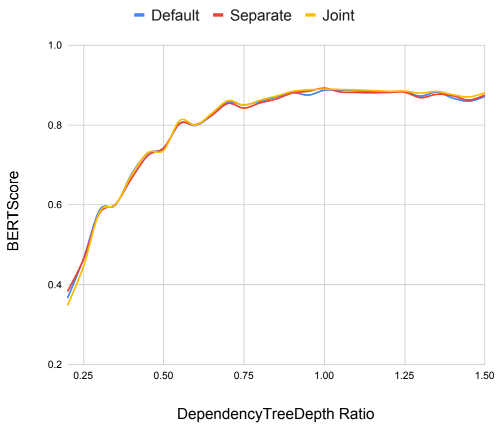
DTD	Strategy	SARI	BERTScore
0.55	Default	40.82 ±0.05	0.805
	Separate	40.68±0.06	0.804
	Joint	40.71±0.06	0.812
0.6	Default	40.54 ±0.05	0.799
	Separate	**40.87±0.05**	0.801
	Joint	40.43±0.06	0.800

WR	Strategy	SARI	BERTScore
0.75	Default	40.61±0.06	0.720
	Separate	41.08±0.06	0.738
	Joint	**41.42±0.06**	0.733
0.8	Default	40.80±0.06	0.776
	Separate	40.32±0.05	0.797
	Joint	40.43±0.06	0.782

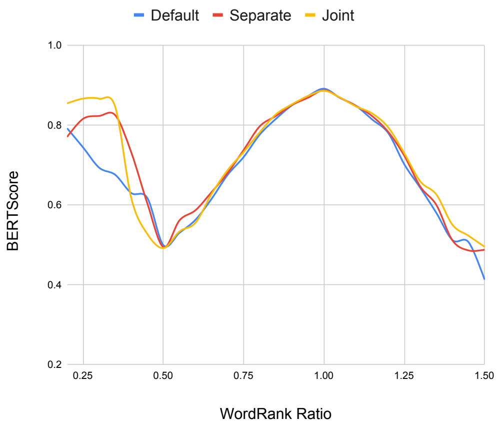
LV	Strategy	SARI	BERTScore
0.65	Default	42.52±0.06	0.750
	Separate	42.55±0.06	0.747
	Joint	42.63±0.06	0.761
0.7	Default	42.26±0.08	0.785
	Separate	**42.86±0.06**	0.782
	Joint	42.31±0.07	0.787

LR	Strategy	SARI	BERTScore
0.6	Default	40.15±0.06	0.758
	Separate	40.25±0.06	0.760
	Joint	40.46±0.05	0.758
0.65	Default	39.91±0.05	0.782
	Separate	40.27±0.05	0.781
	Joint	**40.64±0.05**	0.785

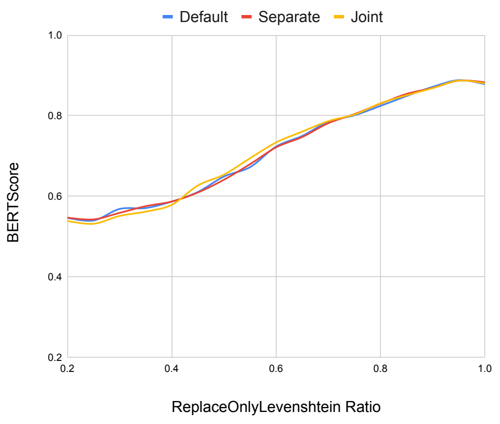
Table 3.6. Results on SARI and BERTScores of peak points in different control tokens (We choose the points with the highest SARI score in the three strategies and corresponding points in the other two strategies).



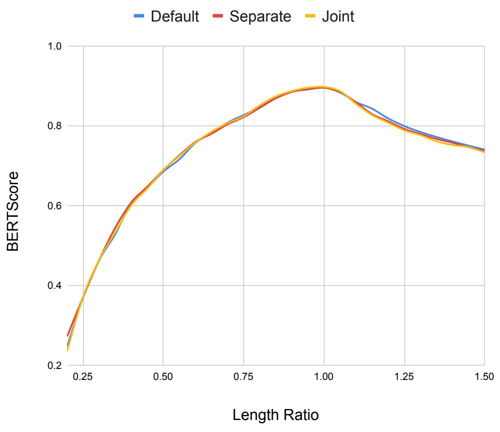
(a) BERTScore of models with DependencyTreeDepth ratio only



(b) BERTScore of models with Word Rank ratio only



(c) BERTScore of models with ReplaceOnlyLevenshtein ratio only



(d) BERTScore of models with Length ratio only

Figure 3.3. The effect of varying control tokens with different tokenization strategies on BERT score.

high consistency in the curves and have a common minimum point at 1. As shown in Table 3.5, it is mainly caused by the low score in both deletion and adding operations.

In addition to the curves, the differences in tokenization methods have marginal effects on the scores while the value of control tokens can change the performance significantly. In Figure 3.2a and 3.2c, the separate tokenization method shows the highest peak point, while in Figure 3.2b and Figure 3.2d, the joint tokenization method has the best performance. The corresponding Table 3.6 also shows the scores in pairs under a unified value. Although the advantage is not as clear as the combined control tokens, the optimised SARI score of either separate or joint tokenization methods is still slightly higher than the default tokenization method.

The Table 3.5 is designed to help readers better understand the reason for variations in Figure 3.2. It shows some local minimum or maximum points within the domain and the corresponding SARI score by operations. The addition score is much lower than the keeping and deletion. It is because there is only limited adding operation in the references and much more expression options to carry a similar meaning, which leads to a low hit rate of the addition operation. At the same time, the keep and deletion are chosen from the existing input and thus have a much bigger hit rate and score.

As for the BERT score, as shown in Figure 3.3, nearly all 3 tokenization strategies show high similarity to each other except Figure 3.3b. The figures show that nearly all models have the highest BERT score around 1. Since the BERT score calculates the correlation between the output and references, when the control token is set to 1, the model processes nothing, and the output is very similar to the input. Under this situation, as shown in Table 3.5, the SARI_keep reaches the top. However, the peak of the BERT score in 3.3c slightly deviates to the left, which shows that the references and input are not identical.

3.3.3 Optimisation sample

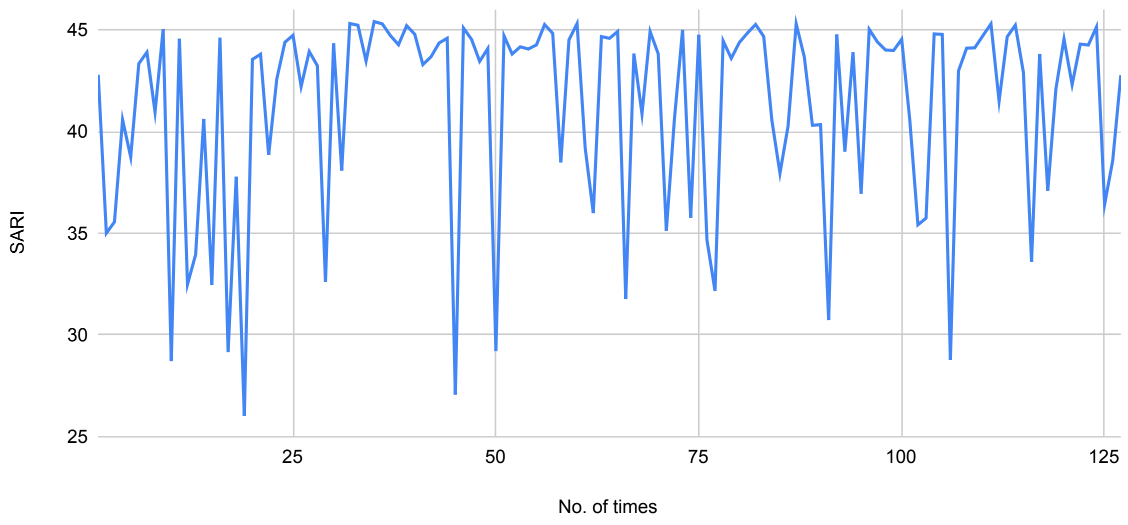


Figure 3.4. SARI score during 128 times of optimisation.

Figure 3.4 shows the SARI score during the optimisation procedure in 128 times attempts. The highest score appears at the 35th attempt and four of the top-five scores appear within 64 times. Even though a higher SARI score can be found between 65 and 128, there is no performance gap between the highest and the second-highest scores.

3.4 Case study

3.4.1 Effect and Limitation of Control Tokens

Source	Reflection nebulae are usually blue because the scattering is more efficient for blue light than red (this is the same scattering process that gives us blue skies and red sunsets).	SARI	BERTScore
LR_1.2	Reflection nebulae are usually blue because the scattering is more efficient for blue light than red (this is the same scattering process that gives us blue skies and red sunsets) and because the light reflects off of them.	21.70	0.825
LR_1.0	Reflection nebulae are usually blue because the scattering is more efficient for blue light than red (this is the same scattering process that gives us blue skies and red sunsets).	19.03	0.937
LR_0.8	Reflection nebulae are usually blue because the scattering is more efficient for blue light than red (this is the same scattering process that gives us blue skies).	25.73	0.879
LR_0.6	Reflection nebulae are usually blue because the scattering is more efficient for blue light than red.	41.71	0.880
LR_0.4	Reflection nebulae are usually blue because the scattering is more efficient.	39.85	0.846
LR_0.2	Reflection nebulae are usually blue in colour.	36.11	0.594

Table 3.7. Effect of varying Length ratio with the others remaining at 1.0.

In this section, we aim to demonstrate the impact of each control token through illustrative examples. Table 3.7 shows the outputs of a sample sentence as the Length Ratio control token varies from 1.2 to 0.2, while keeping the other three control tokens fixed at 1. The differences between adjacent sentences are highlighted with underlines. The results clearly indicate that the output with a Length Ratio of 1.2 has the highest character length, whereas the output with a Length Ratio of 0.2 shows the smallest length. Notably, when the Length Ratio is set to 1, there is no noticeable variation observed. The influence of the Length Ratio control token on the sentence length is evident, and the model effectively learns this relationship. However, it is important to note that the desired meaning of the control tokens in this context is to adjust the sentence length while maintaining the character, lexical complexity, and syntactical complexity identical to that of the source sentence, which is inherently challenging. Despite the model being influenced by the control tokens, their impact is not absolute, and certain limitations persist as the control token is not the sole constraint imposed on the model.

Source	Moderate to severe damage extended up the Atlantic coastline and as far inland as West Virginia.	SARI	BERTScore
LV_1.0	Moderate to severe damage extended up the Atlantic coastline and as far inland as West Virginia.	20.29	0.959
LV_0.8	Moderate to severe damage happened along the Atlantic coast and as far inland as West Virginia.	44.30	0.897
LV_0.6	Moderate to severe damage happened along the Atlantic coast and as far inland as West Virginia.	44.30	0.897
LV_0.4	In West Virginia, the storm caused moderate to severe damage along the Atlantic coast and inland.	44.33	0.527
LV_0.2	The National Hurricane Center (NHC) said that the storm was a ”major hurricane” and not a tropical storm.	26.40	0.058

Table 3.8. Effect of varying ReplaceOnlyLevenshtein ratio with the others remaining at 1.0.

In Table 3.8, we present a set of outputs derived from the same source sentence, each produced with different ReplaceOnlyLevenshtein ratios while maintaining the other three control tokens at 1. Given that the ReplaceOnlyLevenshtein ratio cannot exceed 1, we show sentences generated with ReplaceOnlyLevenshtein ratios ranging from 1 to 0.2. The output with LV_1.0 corresponds to an identical sentence to the source sentence. In the outputs with LV_0.8 and LV_0.6, the two sentences are identical since the model is only able to generate meaningful sentences with grammatical correctness that align with the control token requirements. When the variance in values is too low (e.g., 0.05 to 0.1), it becomes very likely for the model to produce identical outputs. On the other hand, the output for LV_0.2 exhibits the greatest disparity, but its meaning significantly deviates from the source sentence. This observation underscores the limitation of controllable text simplification with control tokens, as there is no guarantee of factual accuracy or preservation of the original meaning.

In Table 3.9, we show 2 sets of outputs with different WordRank ratios along with some other ratios and the control tokens unmentioned remain at 1. In the second and third rows, the model replaces the ’inadvertently’ with ’deliberately’ and ’does not follow’. However, when WR is set to 0.4, the model generates identical outputs. In the fifth and sixth rows, we set the LV to 0.8, which allows more variation in the output, and WR to 0.4 and 0.2. The model further replaces the ’initiates’ with ’starts’ and ’migration’ with ’move away’. Similarly, when WR set 0.2, the model recovers the ’move away’ to ’migration’. The examples in this table show the limitation of the interrelationship among all control tokens.

Source	Shade sets the main plot of the novel in motion when he impetuously defies that law, and inadvertently initiates a chain of events that leads to the destruction of his colony's home, forcing their premature migration, and his separation from them.	SARI	BERTScore
WR_0.8	Shade sets the main plot of the novel in motion when he deliberately defies that law, and inadvertently initiates a chain of events that lead to the destruction of his colony's home, forcing their premature migration, and his separation from them.	24.58	0.862
WR_0.6	Shade sets the main plot of the novel in motion when he does not follow that law, and inadvertently initiates a chain of events that lead to the destruction of his colony's home, forcing their premature migration, and his separation from them.	25.95	0.828
WR_0.4	Shade sets the main plot of the novel in motion when he impetuously defies that law, and inadvertently initiates a chain of events that lead to the destruction of his colony's home, forcing their premature migration, and his separation from them.	19.54	0.850
WR_0.4 LV_0.8	Shade sets the main plot of the novel in motion when he does not follow the law, and inadvertently starts a chain of events that lead to the destruction of his colony's home, forcing them to move away, and his separation from them.	39.60	0.775
WR_0.2 LV_0.8	Shade sets the main plot of the novel in motion when he does not follow the law, and inadvertently starts a chain of events that lead to the destruction of his colony's home, forcing their premature migration, and his separation from them.	32.22	0.817

Table 3.9. Effect of varying WordRank ratio and some other ratios with the others remaining at 1.0.

Source	The four canonical texts are the Gospel of Matthew, Gospel of Mark, Gospel of Luke and Gospel of John, probably written between AD 65 and 100 (see also the Gospel according to the Hebrews).	SARI	BERTScore
DTD_1.2	The four canonical texts are the Gospel of Matthew, Gospel of Mark and Gospel of Luke, probably written between AD 65 and AD 100 (see also the Gospel according to the Hebrews).	34.48	0.882
DTD_0.8	The four canonical texts are the Gospel of Matthew, Gospel of Mark and Gospel of Luke. They are probably written between AD 65 and 100 (see also the Gospel according to the Hebrews).	36.04	0.946
DTD_0.6	The four canonical texts are the Gospel of Matthew, Gospel of Mark and Gospel of Luke. The Gospel of John was probably written between AD 65 and 100 (see also the Gospel according to the Hebrews).	32.21	0.922
DTD_0.4	The four canonical texts are the Gospel of Matthew, Gospel of Mark and Gospel of Luke. The Gospel of John was probably written between AD 65 and 100 (see also the Gospel according to the Hebrews).	32.21	0.922

Table 3.10. Effect of varying DependencyTreeDepth ratio with the others remaining at 1.0.

In Table 3.10, we provide examples showcasing the effect of the DependencyTreeDepth ratio. The output sentence with DTD_1.2 remains the same as the input sentence. As we decrease the DependencyTreeDepth ratio to 0.8, the model splits the long sentence into two shorter sentences. With a lower DTD_0.6, the model goes even further in reducing syntactical simplicity by altering the demonstrative pronoun 'they'. However, when we further decrease the DependencyTreeDepth ratio to 0.4, no observable effect is seen. This example highlights the inconsistency in the impact of control tokens and underscores the importance of selecting proper values for control tokens, as they can significantly influence the resulting sentence.

3.5 Discussion and Future Work

3.5.1 Effect of Quantisation

One phenomenon found during the optimisation section in the original project is that the score of recommended optimisation is even lower than the default values of control tokens at 0.8. A hypothesis emerged that continuous optimisation is not an ideal option to maximise the score. As shown in the four rows in Table 3.3, the score in reimplementation is higher with equivalent values in the four control tokens. There could be several reasons: the algorithm is not working as expected or the optimisation budget is not large enough to find better optimisations. The default tokenization method in the MUSS project that breaks the control tokens into pieces brings more noise and probably lowers the performance. Apart from the verbosity in optimal values, the long tokenization of the control token is another concern of noisy input. Although the results above show no signs of such problems, they may become more serious with the increasing length of control tokens, especially for short sentences. It would be wiser to limit the unnecessary noise in the input to a lower level.

3.5.2 The effect of tokenization Strategy

In addition to the values, as shown in Table 3.6 and Figures 3.2 and 3.3, the tokenization methods can also affect the points and the other points in the curves. Given the null significant difference in SARI based on p-value, the marginal performance gap among the tokenization methods may be caused by the fine-tuned models on a lower training scale. However, in Figures 3.2b and 3.3b, the deviated curves among the three tokenization methods may still imply performance variations between tokenization methods for the Word Rank Ratio control token under certain values. Considering the various requirements of the target audiences, a mixed tokenization method based on the performance curve that maximises the model's performance at different points can be better than a single one. Although it remains unclear whether there will be the same effects in the combined control tokens, the mixed tokenization method can still be promising with the appearance of more different control tokens.

3.5.3 Similarities and Differences among Control Tokens

Figure 3.2 and Table 3.5 expose the reason for variation with the control token and provide a good illustration of nature in each control token. In single control tokens, the peak points mainly fall between 0.6 and 0.7, and the score decreases with the value deviating from the peak point. However, there are still some differences among the control tokens. In the *DependencyTreeDepth Ratio* and *Length Ratio*, the reduction is more dramatic than the other 2. In both graphs, the SARI_add decreases with the value deviating from the peak point and increases slowly when the value is bigger than 1. The SARI_keep and SARI_del fluctuate in the form of 2 half-phase shifted sine functions and the maximum sum is found in between the peaks. The graph of the *WordRank Ratio* shows some diversity in both Figure 3.2b and 3.3b among the tokenization methods. Although there is no explanation for the deviations, the deviations show the potential of combining different tokenization methods. When focusing on the main section from 0.5 to 1, the graph shows characteristics similar to the graphs in the previous 2 control tokens. As for the *ReplaceOnlyLevenshtein Ratio*, the slope is milder on the left side and it seems to have less effect on the SARI score. Unlike the other 3 control tokens, this control token can only indicate the intensity of change but not the direction of change. Although the combined effects are still under research, a more effective control token could be a better solution.

As for the optimal value, they are the optimal values within the budget. When rerunning the code, it is quite common to have a different set of optimal values. This is one limitation of the current SOTA system and we propose the predictor to improve this drawback. In addition, the optimal values for one control token and for the combined control tokens are different. The correlation among the control tokens presumably causes this variation. If the four control tokens can be designed to work independently, the graph on a single control token can be directly used to find the optimal value. However, the graph of combined control tokens is bound to have some distortions for now. Based on the details from Graph 3.2a to 3.3d, it is also clear that the value of control tokens can significantly affect the performance of the models trained in this way and should be treated carefully.

Another interesting finding between SARI and BERT in this paper is that most BERT score for optimal value is around 0.78 to 0.8. However, as shown in Figure 3.3b and 3.3d, there are more than 1 points that have such value, so the BERT score alone cannot be used to evaluate the text simplification results. It may be a necessary but not sufficient condition for a good simplification. Since the SARI score is not perfect and relies on references, it is important to build non-reference-based metrics to evaluate the model on a different genre of corpora. The BERT score may play a role in these new metrics. Thus, this guess is worth further verification in future work.

In addition to the values, as shown in Table 3.6, the tokenization methods can also affect the peak score. In the curves, there are different optimised methods for each certain point. Although the performance differences may be caused by the fine-tuned models on a lower training scale, they may still imply performance variations between tokenization methods. Considering the various requirements of lay users, a mixed tokenization method based on the performance curve may maximise the model's performance at different points better than a fixed one. Although it remains unclear whether there will be the same effects in the combined control tokens, the mixed tokenization method can be still promising with the appearance of more different control tokens. However, a more lightweight and efficient training method should be introduced to solve the problem of balancing cost and effect.

3.5.4 Future Work

Since the control tokens can be regarded as customised prompts to the models, the applicability of control tokens or similar ideas in other NLP tasks is worth doing in the future. With the popularity of prompt tuning, similar ideas can be applied to other tasks which need control ability to the outputs. In addition, as the tokenization strategy shows only marginal effects on the performance in the control token experiments, a further study on the longer prompts may be worth doing, especially in the prompt-driven generative models with templates or formats. Another future work is related to the SARI score. Since the SARI score highly relies on the quality and diversity of the references and can reflect only partially on the simplicity of the output. This could be overcome with a new reference-less metric to measure the readability in different audiences' perspectives or measure the similarity of the outputs with text genre targeting specific audience groups with large language models. As shown in Section 3.4, there is interference among the control tokens, impairing the usability and practicality of control tokens. One alternative solution is to break the system into several subsystems, build a pipeline to normalise the sentence and divide and conquer the subtasks in steps.

3.6 Conclusion

In the investigation, we have shown the results and importance of control tokens with different values and tokenization methods, which can be used to balance user intention and performance. We proposed some improvements in quantisation, compared the influences of different tokenization strategies of control tokens and proposed possible further improvement means. Although the proposed suggestions may improve text simplification tasks marginally, they may also be generalised to prompts designing on other controllable NLP tasks.

Chapter 4

Evaluation and Limitation of Controllable Text Simplification

As discussed previously, there are different user groups with different demands. In addition to the user groups, there are also different genres of texts, such as news, novels, academic articles and so on. In order to study the performance of current text simplification models in certain scenarios and make the expert-level information and knowledge (i.e. Medicine) more accessible to lay users, we present genre-specific text simplification research alongside a study on the effects of different genres.

In this chapter, we focus on the evaluation of the controllable text simplification methods. We compared text simplification models without control and controllable text simplification models in both general test scenarios and genre-specific scenarios. we leveraged the newly published text simplification dataset Simple-TICO19 (Shardlow and Alva-Manchego 2022), designed a test scenario for controlled text simplification with different genres, proved the effects of transfer learning on the genre-specific datasets, compared the performance of generic and expert models with different scales.

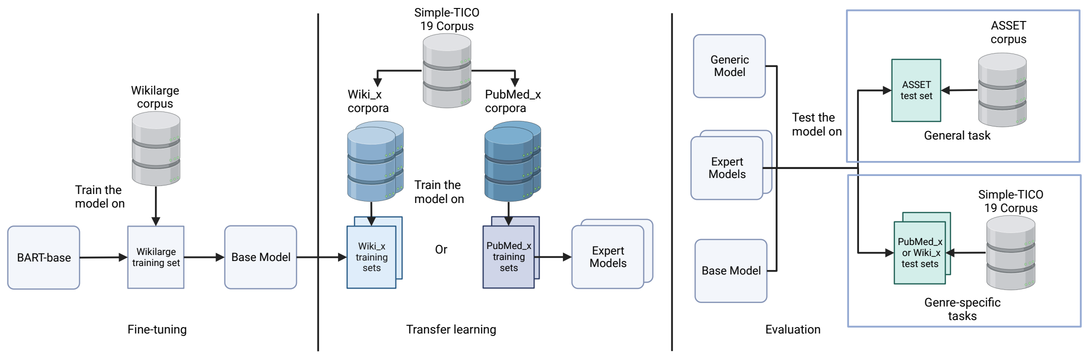


Figure 4.1. The methodology is represented in three sections. In the left section, we fine-tune BART-base on the WikiLarge training set to give the **base model**. In the middle section, we regard the task as transfer learning and further fine-tune the base model on our Wikipedia_x and PubMed_x training sets to generate the **expert model(s)**. In the right section, we add 2 zero-shot **generic models** through publicly available APIs. We then evaluate our base model, expert models and generic models in the generic simplification task (The Asset test set) and the genre-specific tasks (the Wikipedia_x and Pubmed_x test sets) and compare the results for the models.

4.1 Methodology

In this section, we describe the experiments that were undertaken. A visual representation of our methodology is provided in Figure 4.1, which is explained in further detail throughout the following subsections.

4.1.1 General and genre-specific task

As illustrated in figure 4.1, there are two different tasks. For the genre-specific tasks, we employed the Simple-TICO19 dataset (Shardlow and Alva-Manchego 2022), which consists of translations and simplifications related to COVID-19 from various resources representing distinct genres. This dataset encompasses a total of 3,173 parallel sentences available in both English and Spanish. In this project, we solely utilized the English section. We split this dataset based on genre and use the subsets as training and test sets for the expert models. To minimize any potential bias introduced by specific dataset partitions, we generated 30 distinct permutations or partitions from subsets from each genre. Consequently, we performed the same procedure for each partition, effectively creating 60 expert models for every partition. However, due to time and computational constraints, we limited our evaluation to 40 expert models derived from the two genres across the 60 partitions. This approach allowed us to provide a comprehensive analysis while managing the practical limitations associated with time and computational resources.

4.1.2 Transfer learning

In this experiment, there are three types of models: the base model, generic models and expert models. For the base model, we adopted the reimplemented MUSS model from Chapter 3. For general models, we leveraged the most powerful and accessible models at the time, such as ChatGPT and GPT-3 (text-davinci-003). As for the expert models, we further fine-tuned the base model on the training set of 30 partitions from the Simple-TICO19 dataset (Shardlow and Alva-Manchego 2022). Since the size of subsets for each genre is relatively smaller, we regard this process as transfer learning and verified the effect before and after transfer learning with the base model.

4.1.3 Prompt Design

Input	Please simplify this sentence for me: ”Asymptomatic or only mild symptoms were detected when bats were infected with CoVs, indicating the mutual adaptation between CoVs and bats.”
Output	When bats get infected with CoVs, they do not show symptoms or only mild symptoms, which shows that CoVs and bats have adapted to each other.

Table 4.1. Prompt design

In the experiment, we leveraged the GPT-3 (Brown et al. 2020) model and ChatGPT with hand-crafted prompts. Since the main focus is not on how to achieve the best performance of LLMs with prompts only, we didn’t apply prompt learning or prompt engineering on the prompt. Instead, we asked the ChatGPT for the best prompt and set it to ’Please simplify this sentence for me: ”(Target sentence)”’ and an example is shown in Table 4.1.

4.1.4 Evaluation and metrics

In this experiment, we applied the SARI score (Xu, Napoles, et al. 2016) and BERTScore (T. Zhang et al. 2019) as the main automatic metrics. We also conduct the human evaluation for the genre-specific experiments as SARI and BERTScore can only partially reflect performance differences. Furthermore, the lack of multiple reference sentences in the test set of Simple TICO-19 (Shardlow and Alva-Manchego 2022) will lead to a less trustworthy score, compared to the 10 references in the ASSET test set (Alva-Manchego, Martin, Bordes, et al. 2020).

4.2 Experiment design

4.2.1 Preprocessing

Following the MUSS implementation (Martin, Fan, et al. 2020), the four control tokens are introduced as follow:

- <DEPENDENCYTREEDEPTH_*x*> representing syntactic complexity
- <WORDRANK_*x*> representing lexical complexity
- <REPLACEONLYLEVENSHTEIN_*x*> representing the token difference ratio
- <LENGTHRATIO_*x*> representing the difference in length

Each control token is calculated by comparing the above ratios in complex-simple sentence pairs. After the calculation of the control tokens for the training set, the calculated value of complex sentences is added as a prompt to the beginning of the corresponding complex sentences. The value of these control tokens is rounded to 0.05 and limited in the range of 0.2 to 1.5, except for the LV, which is limited from 0.2 to 1. In Simple Tico-19 (Shardlow and Alva-Manchego 2022), due to the manual translation, there are some sentences that did not implement simplification or are unsuited for simplification. These unsimplified pairs were removed in the following derivative subsets that we used in our research. We further split the dataset according to the genre. The number of instances after the filtering of each subset is shown in table 4.2.

Data source	Number of instances
CMU	122
PubMed	809
Wikinews	76
Wikivoyage	206
Wikipedia	1224
Wikisource	101

Table 4.2. Number of instances in each data source

Taking into account various factors such as the literary style, intended readership, and the number of sentences, we selected the **PubMed** and **Wikipedia** subsets to serve as representative examples of two distinct genres for our genre-specific text simplification experiments involving expert and generic models. Specifically, the sentence pairs labelled as ’wiki’ and ’Wikipedia’ were exclusively sourced from Wikipedia, which led us to combine these two subsets into a single comprehensive **Wikipedia** dataset, comprising a total of 1224 instances.

To create training, validation, and test sets, we proceeded to perform random splits on the **PubMed** and **Wikipedia** datasets, ensuring that the data was divided in a ratio of 8:1:1. These splits were conducted using a predetermined random seed to maintain consistency and reproducibility. Consequently, each permutation of the dataset, generated under a specific random seed denoted as ***x***, was labelled as **PubMed*x*** and **Wikipedia*x***. For example, we have datasets such as **PubMed0** and **Wikipedia0**. This process resulted in the creation of three distinct sections for each dataset, containing 978, 122, and 124 sentence pairs in each **Wikipedia*x*** subset, and 647, 81, and 81 sentence pairs in each **PubMed*x*** subset, corresponding to the training, validation, and test sets, respectively. Such partitioning ensures a balanced distribution of data and allows for comprehensive evaluation and comparison of the models’ performance.

4.2.2 Models for Text Simplification

In this chapter, we propose to compare the performance among three versions of a text simplification model: the base model, the generic model and the expert model.

The **base model** is based on BART-base (Lewis et al. 2020) with 6 layers in both encoder and decoder and 140 million parameters. The base model is fine-tuned on the training set of Wikilarge (X. Zhang and Lapata 2017a) only with the above-mentioned 4 control tokens. The following hyper-parameters were employed: Learning rate: 2e-5, Weight Decay: 0.01, Training epochs: 10.

After fine-tuning, the training loss reaches 0.85 without overfitting. By comparing the SARI score of our model on the ASSET test set (Alva-Manchego, Martin, Bordes, et al. 2020) with the original results in MUSS (Martin, Fan, et al. 2020), it is reasonable to claim that it has reached to the designed performance level.

For **generic models**, we tested the GPT-3 (Brown et al. 2020) based model and ChatGPT via the API and online platform by OpenAI. Instead of training or fine-tuning, we leverage the 2 models by prompting. The prompt is set to ”Please simplify this sentence for me: ” and will be added to the beginning of each complex sentence, then the model will try to generate a simplified version of the input text after the colon. The exact model prompted in the GPT-3 is called ”text-davinci-003”, which is the latest version, the parameters are set as follows:

- temperature: 1 The sampling temperature, ranging from 0 to 2, influences the degree of randomness in the generated output. Increasing the temperature leads to a higher level of randomness, while decreasing it results in a more focused and deterministic output;
- frequency_penalty: 0 The Frequency penalty, ranging from -2 to 2, controls the variation between the output and input. Positive values penalize new tokens by their existing frequency in the text thus far, discouraging the model from repeating the same line verbatim and promoting diversity in the generated output. By assigning positive values, the model is encouraged to explore alternative phrasing or introduce fresh perspectives. On the other hand, negative values aim to preserve the original input to the greatest extent possible, allowing the model to adhere closely to the given text and minimize deviations. Striking a balance between these positive and negative values enables the model to exhibit both creativity and faithfulness to the source material, en-

hancing its overall performance in generating diverse and coherent text;

- presence_penalty: 0 The presence penalty, ranging from -2 to 2, controls the presence of new tokens. Positive values penalize new tokens by their appearance in the text thus far, thereby enhancing the model's inclination to introduce fresh subjects and discuss novel topics. Conversely, negative values aim to retain the original tokens to the greatest extent possible, discouraging excessive changes or introductions of new information. By fine-tuning these values, the model's behavior can be tailored to strike a balance between introducing novelty and maintaining consistency with the existing text.
- max_token: 2000 Max token controls the maximum number of tokens to generate in the completion.

As for ChatGPT, due to the fast iteration speed, the only information available is ”ChatGPT Jan 9 Version”. During our experiment, since there was no official API released, we accessed the ChatGPT via a fake web browser with session IDs to request responses in batches. The ChatGPT is then accessed on the online platform in the conversations automatically. There is no guarantee of performance compared to the results of API access and different versions of ChatGPT.

The **expert model(s)** are composed of base models after transfer learning on corresponding permutations of subsets. By fine-tuning the pre-trained model on the preprocessed Wikilarge training set (X. Zhang and Lapata 2017a), the base model learns how to generate simplifications based on the value of control tokens. To leverage the base model as an expert text simplification model, we further fine-tune the model on the preprocessed training set of **Wikipedia*x*** and **PubMed*x*** and then have the corresponding expert models for each permutation of **Wikipedia*x*** and **PubMed*x***. The fine-tuning hyper-parameters are the same as fine-tuning the base model. In the experiment, we build 30 expert models for both **Wikipedia*x*** and **PubMed*x*** and evaluate the performance for 20 permutations.

4.2.3 optimisation

Since the values of control tokens influence the quality of the generated output and overall model performance, it is necessary to find an optimal value of the control tokens for the model on the test sets. This is consistent with the previous state of the art, but it is also worth noting that the results reported are limited to specific datasets and alternative parameters may be optimal for different datasets. The value options of most control tokens fall between 0.2 to 1.5 (or 0 to 1 for Levenshtein), so there is only finite options are provided during optimisation, and the optimisation problem is reduced to finding the best value combination of control tokens within the optimisation budget. The optimisation budget limits the total number of attempts to find the set of values of control tokens to maximize the metric, which is set to the SARI score. The optimisation budget for the general tasks on the ASSET valid set (Alva-Manchego, Martin, Bordes, et al. 2020) is 128, while the value for genre-specific tasks on the valid sets of permutations of **Wikipedia*x*** and **PubMed*x*** is reduced to 64 for time-saving. We used Nevergrad (Rapin and Teytaud 2018) to find out the local optimal value within the budgets.

4.2.4 Genre-specific Experiments

To verify the effect of transfer learning, we computed the SARI score on the test set of **PubMed*x***s and **Wikipedia*x***s. Since there is only one reference sentence in the Simple TICO-19, the SARI score on these test sets is only applicable and comparable within the experiment. We tested the base model, generic model and expert models on the test sets from 20 permutations of **PubMed*x***s and **Wikipedia*x***s.

For expert models from the same genre of the test set, we only evaluate the expert model trained on the corresponding training set of the test set to avoid data leakage. The average of these models is reported as ‘Average corresponding genre models’ in Tables 4.5 and 4.6.

As for the expert models from the other type, we tested 30 models from permutations with different random seeds. The overall results are shown in table 4.5 and

4.6, and the details are shown in figure 4.2 and 4.3. The full results are available as a table in the Appendix.

4.2.5 Evaluation

In the evaluation of the general task, we conducted tests on both the base model and general models, and selected **Wikipedia0** and **PubMed0** as representative expert models. To ensure the integrity of our data, we implemented measures to prevent any potential data leakage during the evaluation of genre-specific tasks. Specifically, for each partition of the dataset, only the corresponding expert model was assessed and clearly marked as the ”corresponding model” in the tables. The expert models from other genres were evaluated as a collective group, and the mean values were recorded to provide an overview of their performance. As for the base model and general models, they underwent the same evaluation process as the general task.

For the human evaluation, we recruited 17 human annotators through Amazon Mechanical Turk, specifically selecting individuals who possessed the ”Master” qualification, indicating their reliability and expertise as trusted workers on the platform. All annotators reported having achieved an educational background at the undergraduate level or higher, ensuring a solid foundation for their evaluation. Among the annotators, twelve were non-native English speakers, while the remaining five were native English speakers. Each annotator was presented with 20 instances in the form of table 4.3, featuring an original sentence along with a pair of sentences derived from the generic and expert model (randomly assigned to avoid bias). The annotators were then requested to assess these provided sentences, responding to two questions using a 5-point Likert scale, which allowed for a comprehensive evaluation process.

Original Text	Simplified sentences	Meaning preservation	Simplicity
It is possible that many mammals including domestic animals are susceptible to SARS-CoV-2.	Many types of animals, including pets, may be able to get infected with SARS-CoV-2.	Disagree	Agree
	Many mammals including domestic animals may be susceptible to SARS-CoV-2.	Agree	Agree

Table 4.3. Sample of human evaluation questionnaire

4.3 Results

4.3.1 General task

Model		SARI↑	BERTScore↑
Base	BART-base	44.05	0.777
Generic	GPT-3	41.73	0.703
	ChatGPT	**46.42**	0.731
Expert	**Wikipedia0**	43.24	**0.835**
	PubMed0	43.67	0.812

Table 4.4. SARI and BERTScore on ASSET test

Table 4.4 presents the SARI scores and BERTScores on the ASSET test set. Among them, ChatGPT emerges as the top performer in terms of SARI score, indicating its proficiency in text simplification. However, when considering BERTScore, the expert model **Wikipedia0** takes the lead. It is noteworthy that both generic models demonstrate decent performance on the test set without any fine-tuning or prompt engineering. ChatGPT surpasses the base model by a large margin, while GPT-3 lags behind in the SARI score. Notably, the BERTScore is comparatively lower for both generic models when compared to the base model. These metrics showed the performance gap between GPT-3 and ChatGPT, a distinction that aligns with the differences in model structure and scale.

4.3.2 Genre-specific task

Table 4.5 shows the average SARI and BERTScores over all 20 permutations of test sets from different models. The first row shows the average SARI score of the base model, which is only fine-tuned on WikiLarge and based on the BART

	Model	SARI ↑	BERT↑
Base	BART-base	40.78	0.741
Generic	GPT-3	29.03	0.530
	ChatGPT	31.12	0.542
Expert	Average corresponding expert **Wikipedia** models	**44.30**	**0.756**
	Average **PubMed** models	42.75	0.741

Table 4.5. Average SARI and BERTScore on all **Wikipeida*x***

	Model	SARI↑	BERT↑
Base	BART-base	40.56	0.723
Generic	GPT-3	30.72	0.547
	ChatGPT	31.55	0.515
Expert	Average Corresponding expert **PubMed** models	**45.05**	**0.741**
	Average **Wikipedia** models	43.38	0.726

Table 4.6. Average SARI and BERTScore on all **PubMed*x***

base. The following two rows show the SARI scores of two generic models on the test sets. The last two rows show the SARI scores of all expert models. The corresponding **Wikipedia** or **Pubmed** models refer to the corresponding expert models after transfer learning on the training sets (e.g., model **Wikipedia0** to test set **Wikipedia0** and model **Pubmed19** to test set **Pubmed19**). The last row shows a combined average SARI score of expert models trained on the opposite genre. The detailed SARI score can be found in figure 4.2 and 4.3. The same rules also apply to table 4.6.

In both table 4.5 and 4.6, the corresponding expert models, which is the expert model transfer learnt on the corresponding training set, have the highest overall SARI score. Although the generic models show very competitive performance in the general task, the lack of fine-tuning led to lower performance in terms of SARI score in the genre-specific scenario. The fine-tuned models also take advantage of learning the text style in the training set. The overall performance gap between the two generic models is aligned to the gap in Table 4.4. As for the expert models, they have a much higher SARI score and appear to have a much higher performance, but the actual performance gap between the generic models and expert models needs further exploration. What the SARI score can tell is how they benefit from the transfer learning compared to the base model. It is surprising to see

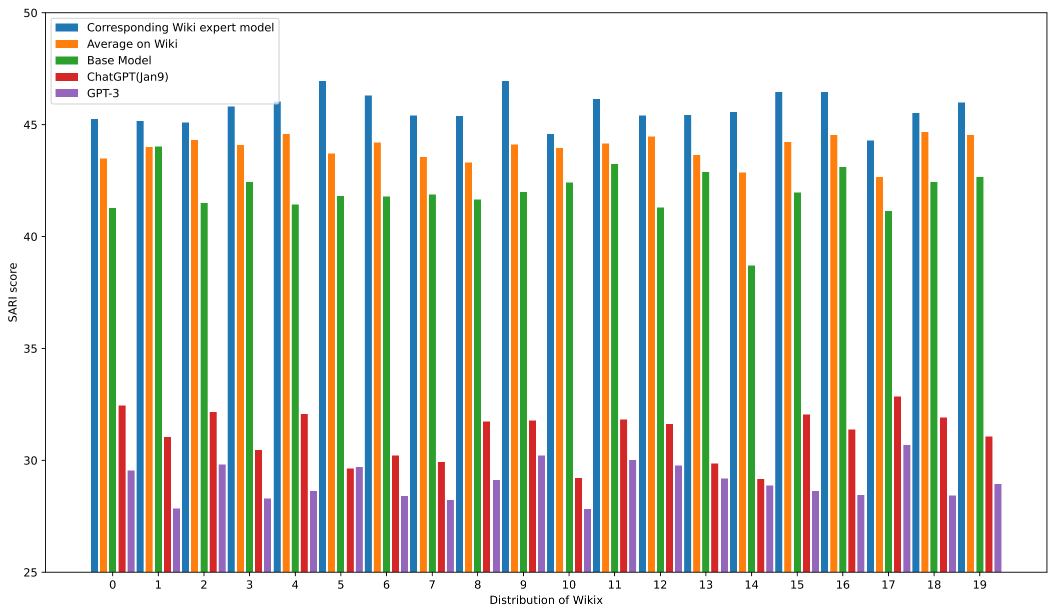


Figure 4.2. SARI score on 20 **Wikipedia*x*** permutations for different models

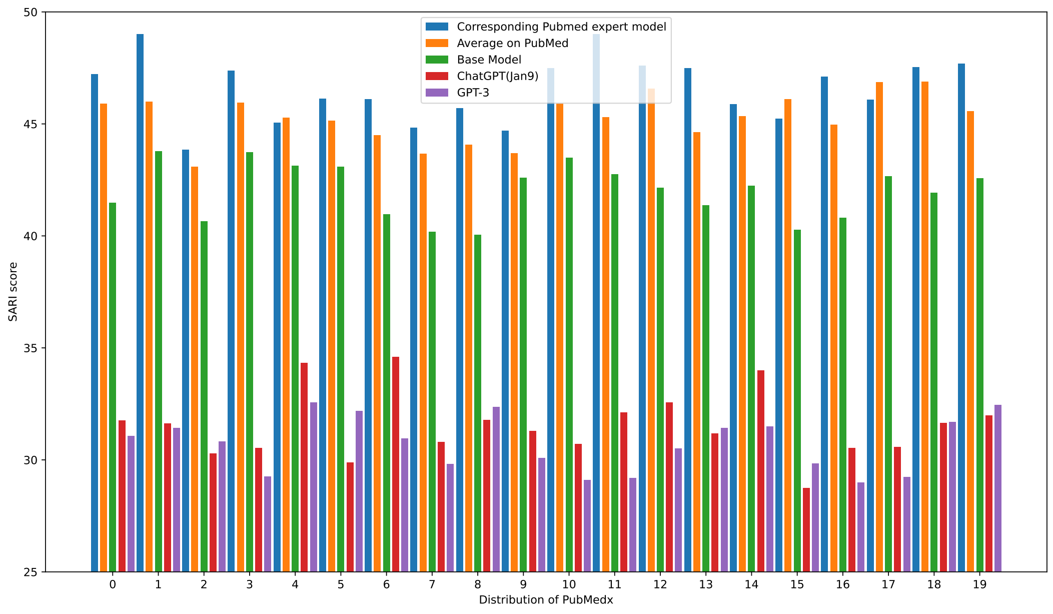


Figure 4.3. SARI score on 20 **PubMed*x*** permutations for different models

the improvement for both kinds of expert models, which is presumably caused by the sharing characteristics in the two subsets (both are related to Covid-19 information). As a result, the improvement of the overall SARI score for expert models shows the effectiveness of transfer learning for genre-adaptive text simplification.

We also evaluated BERT-score for our generic and expert models on the expert datasets. The BERTScore similarly shows that the simplifications produced by generic models in the expert setting are of worse quality than those produced by the expert models. For **Wikipediax**, we note that there is little improvement in BERTScore on the expert models, with the model trained on **PubMedx** performing marginally higher than the base model, or the model trained on Wikipedia. The base model was also fine-tuned on Wiki-large which is the same genre as the **Wikipediax** subsets coming from Simple TICO-19, which may explain why there was little performance gain. For **PubMedx**, the expert models both improve when measured against the Base model. The genre-specific PubMed expert models attain a higher BERTScore than those fine-tuned on the Wikipedia subsets.

Generally, the detailed SARI score is aligned with the overall performance. The corresponding expert model outperforms the other four models in the SARI score across all permutations and the generic models have a much lower SARI score than the base model. The SARI score also shows some similarities among models. We listed the detailed SARI and BERTScore in the appendix in figure 4.2 and 4.3 and the remaining tables.

In the SARI score of 20 **Wikipediax** test sets, nearly all models show a performance drop on **Wikipedia14** and **Wikipedia17**. However, the two generic models show a different phenomenon on **Wikipedia17** by increasing the SARI score. The fluctuation of the SARI score also demonstrates the effect of permutation in the genre-specific experiments and also shows that some of the permutations are not ideal distribution of training and test sets. As for BERTSCore, there are more inverted performances, which also align with the overall performance.

Unlike the detailed SARI score for **Wikipediax**, there are several divergences on certain permutations of **PubMedx**. In **PubMed15** and **PubMed17**, the average performance of expert models from **Wikipedia** outperforms the corresponding expert model. Similar incidents happen between the two generic models too. Considering the big performance gap between the GPT-3 and ChatGPT, the inconsistency is probably caused by the lack of more reference sentences or the poorly designed training and test sets. There is also some commonality in **PubMed**. In detailed SARI score, the three models share a common variation in performance on **Pubmed1**,**Pubmed2** and **Pubmed3**. In addition, there are also similar situations in BERTSCore.

Comparing the base models with the generic models, it is unclear why the generic models perform so poorly on the test sets in terms of SARI score. One possible reason is that both base models and expert models are fitted with the optimal value of control tokens to maximize the SARI score while the prompted generic models are not. The calculation method of the SARI score prefers the sentence that keeps the most of original content under the condition of lack of reference sentences.

Model	Simplicity ↑	Meaning Preservation ↑
Generic	2.55	2.86
Expert	2.46	3.17

Table 4.7. Human evaluation score on test set of **Wiki0** and **PubMed0** (out of 4)

Table 4.7 presents the results of the human evaluation, with scores ranging from 0 to 4, indicating the level of agreement or disagreement with the given statements. The evaluation primarily focused on two key aspects: the degree of sentence simplification and the preservation of meaning. Regarding simplicity, the generic model (ChatGPT) achieved a similar score to the expert models under evaluation (**Wiki0** and **PubMed0**), while with a slight advantage. However, when it comes to meaning preservation, ChatGPT received a lower evaluation compared to the expert models. This suggests that ChatGPT may have neglected important details that the expert models successfully retained. It is worth noting that, contrary to the SARI scores shown in Table 4.5 and 4.6, the performance of the generic model proves to be competitive with the expert model in terms of human evaluation.

4.4 Case Study

Table 4.8 presents a selection of examples generated by GPT-3, ChatGPT, and the corresponding expert model. These examples allow us to reveal the strengths and weaknesses of the tested models. In the first example, ChatGPT not only provides a definition of PoCT but also replaces specific terms and less common words like ”specimen” and ”puncture” with more commonly understood synonyms. In contrast, the other model follows closely to the original text without incorporating such simplifications. This example highlights ChatGPT’s proficiency in both lexical and conceptual simplification. However, it is important to note that while ChatGPT demon-

Source:	For PoCT a single specimen of blood is usually obtained by skin puncture.
Reference:	For PoCT, a single specimen of blood is usually obtained by skin puncture.
GPT-3:	A skin puncture usually gets one sample of blood for PoCT.
ChatGPT:	For Point of Care Testing (PoCT), a small sample of blood is usually taken by pricking the skin.
Expert:	For PoCT, a single specimen of blood is usually obtained through skin puncture.
Source:	It is possible that many mammals including domestic animals are susceptible to SARS-CoV-2.
Reference:	It is possible that many mammals, including domestic animals, are susceptible to SARS-CoV-2.
GPT-3:	Mammals, including domestic animals, may be vulnerable to SARS-CoV-2.
ChatGPT:	Many types of animals, including pets, may be able to get infected with SARS-CoV-2.
Expert:	Many mammals including domestic animals may be susceptible to SARS-CoV-2.
Source:	Personal protective equipment (PPE) is considered less effective than engineering and administrative controls, but can help prevent some exposures.
Reference:	Personal protective equipment (PPE) is considered less effective than engineering and administrative controls. However, it can help prevent some exposures.
GPT-3:	PPE is not as good as engineering and admin controls but can prevent some exposures.
ChatGPT:	PPE is not as good as other ways to prevent exposure, but it can help some.
Expert:	Personal protective equipment (PPE) is considered less effective than engineering and administrative controls, but can help prevent some exposures.

Table 4.8. Examples of simplifications from different models

strates these simplification capabilities, its output may not always align perfectly with the reference sentence, which may result in a lower SARI score for such simplifications.

In the second example, both GPT-3 and the expert model exhibit a consensus and produce similar simplifications. However, ChatGPT encounters a challenge as it simplifies ”domestic animals” solely as ”pets” and overlooks the inclusion of other mammals, which raises the concern of misunderstanding and misinformation in lexical simplification.

In the third example, both GPT-3 and ChatGPT removed the explanation of the abbreviation, which potentially decreased the readability of the sentence. The inconsistency of the performance of generic models can be an obstacle to applying such models to downstream tasks. In addition to that, the definition of simplicity for the generic models is also vague. We found that many of the outputs of ChatGPT are much shorter than the outputs of expert models. However, the short sentences don’t always align with the simplicity and better readability.

In general, ChatGPT demonstrates its ability in conceptual simplification with its enormous training data as the knowledge base. It also indicates that the expert models took a more conservative approach, preserving a significant portion of the original text, while GPT-3 and ChatGPT showed more extensive paraphrasing. However, the inconsistency in performance undermines the reliability of applying ChatGPT in professional fields like the medical domain.

4.5 Discussion and Future Work

The performance of generic models is impressive in the general task. The generic model can become the new SOTA in many natural language processing tasks with proper prompts. However, the scale of the parameters in LLM like ChatGPT makes it almost impossible to be deployed locally. In addition, it can hardly be fine-tuned by an individual or a small group of researchers due to the high requirement for computation power. Even though it can be leveraged by prompts, when it comes to the specialised domain or private information, data privacy prevents it from becoming a universal solution for all people, which limits the applications in real-life scenarios. Another issue is low BERTScore in both general and genre-specific tasks, which indicates deviation in meaning preservation.

When it comes to genre-specific tasks, the generic model is less competitive than it is in general tasks. Based on the human evaluation (Table 4.7), the expert model shows similar or higher performance than the generic model. Although the generic model trained with a much larger corpus contains more internal knowledge, it is hard to determine where to stop the simplification without quantitive prompts. Considering the lower BERTScore from the generic models, it seems that the ChatGPT over-paraphrased the input, compared to the expert models. Results from human evaluation also agree that the expert models appear to preserve the meaning better than the generic model. However, the expert models are equipped with control tokens and inevitably will delete some content or information based on the control token they are given as well. This result shows that human annotators may prefer the more conservative settings of the expert models as opposed to the paraphrasing of the generic models in terms of meaning preservation.

The results in Table 4.5 and 4.6 prove the effectiveness of transfer learning after fine-tuning. As mentioned in the results, both expert models benefit from transfer learning in the genre-specific task. One possible reason for the improvement of the other kind of expert models may be the common ground between the two subsets of different genres, the topic and context for example. Yet the performance gap between the two types of expert models shows that genre or text style still matters and causes performance differences. The genre-adapted model can be a potential solution to better fit the requirements of different groups of lay users.

Even with highly capable generic or expert models, there is still the possibility for the introduction of factual errors in the output. With the convincing performance of generic models like ChatGPT, the hallucination problem become more serious than ever before. When the task is related to a crucial area such as medicine or legal help, the introduction of misleading information may cause severe problems. To improve the robustness of the simplification system, it is necessary to build a factual evaluation system in the future (Devaraj et al. 2022; Ma, Seneviratne, and Daskalaki 2022). Unlike other text generation tasks, simplification maintains the essential information in the input, thus it is easier to judge whether there is misleading content or hallucinations. BERTScore, which measures the meaning preservation for the implications, could be extended into a tool to measure the deviation of original meanings in future work.

Another problem is the explanation of abbreviations. For lay users unfamiliar with the abbreviations and technical terms, it is important to explain the meaning of these unique words or phases. ChatGPT has a huge knowledge base to understand common abbreviations. However, technical terms in certain domains may be unknown for the generic model and the abbreviations may refer to different phrases in different contexts. To avoid the above problem, the model needs to have a genre-specific knowledge base in future work, which allows the model to identify and explain the abbreviations and terms. To achieve this goal, a model competitive with an external source of knowledge base is required. In addition, the knowledge base should be combined with lexical complexity evaluation to decide which term needs explanation.

4.6 Conclusion

In this chapter, we compared the performance differences between generic models and expert models on general and genre-specific simplification datasets. We showed the effect and practicality of transfer learning in genre-specific datasets with less amount of samples. The performance drop on general tasks after transfer learning is acceptable and may be further reduced in future studies. The performance, cost-effectiveness and portability of expert models prove themselves as one of the practical solutions for domains-specific or genres-specific tasks.

Chapter 5

Improvement on Controllable Text Simplification

As mentioned earlier, it is important to acknowledge the limitations of the optimisation method used for the MUSS (Martin, Fan, et al. 2020). One significant limitation is that the optimal value of control tokens determined at the corpus level may not necessarily be the optimal value at the sentence level. This discrepancy becomes more apparent as the size of the test set increases. Another constraint is the absence of a validation set in real-world application scenarios, which hinders the ability to identify the optimal values accurately. Despite the model's ability to learn from the input data, the incremental improvement achieved through this process may not necessarily lead to the best performance observed in the experiment. Moreover, as the data size increases, there is a potential risk of performance deterioration, adding to the challenges associated with the optimisation method.

In this chapter, we propose a different method to predict the control tokens, which serves as a replacement for the optimisation step utilized in previous settings. To examine the effectiveness of these predictors, we conducted tests using both regression and classification methods and subsequently compared the obtained results with those achieved through the traditional optimisation approach. Moreover, we explored various approaches to leverage the predictor models, aiming to fully unlock their potential and exhaustively assess their capabilities.

5.1 Methodology

In this section, we illustrate the methodology of the system. Figure 5.1 shows the flow chart of the whole system. Compared to the system implemented in Chapter 3, the major difference is the introduction of control token predictors, which are used to predict the value of control tokens. More details will be illustrated in the following subsections.

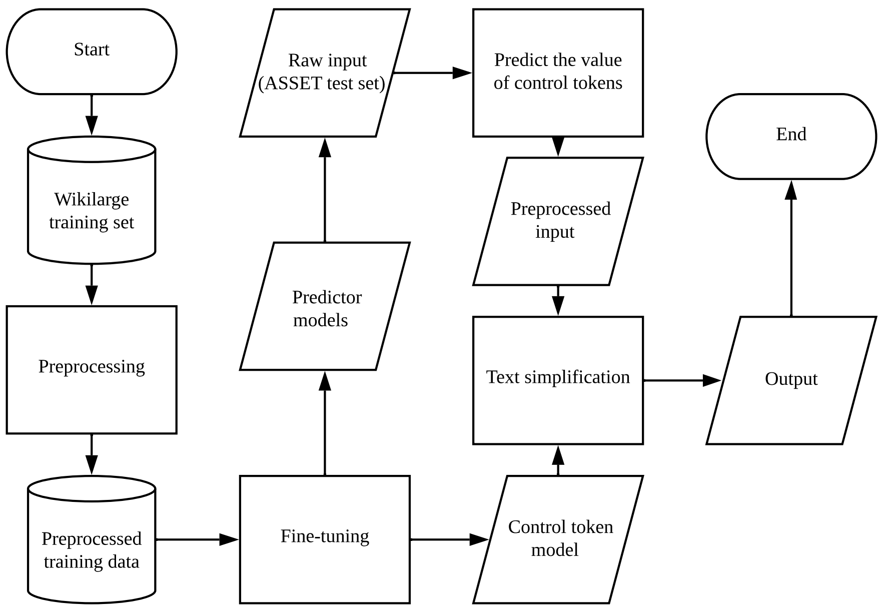


Figure 5.1. The methodology is represented as a flow chart. With the help of control token predictor models and controllable text simplification models, we implement controlled text simplification on the sentence level.

5.1.1 Prediction of Optimal Control tokens

In contrast to the method of finding an optimised set of control tokens at the corpus level adopted in MUSS (Martin, Fan, et al. 2020), we propose both regressive and multi-classification methods of predicting the value of each control token at the sentence level. As shown in the centre of Figure 5.1, there is an extra step of fine-tuning control token predictors, which will predict the value of each control token in each sentence. The predicted values will be combined with the raw input and achieve control on the sentence level.

However, this method is based on 2 premises: There is a unique ideal simplification for every sentence; By feeding the ideal value of control tokens from the ideal simplification, the model can generate the ideal simplification. Unlike optimising the value of control tokens on the test set, we assume the ideal value is an intrinsic property and can be predicted. In addition, the problem can be regarded as a regression problem to predict the optimal value of control tokens and how to fine-tune autoregression models to do so for every sentence. In the experiment, we chose the average value of multiple reference sentences as an approximation to the ideal value and compared it with the predicted values and the optimised value. Although the median value of reference sentences shows slightly higher SARI and BERTScore on the ASSET test set, the predicted median values are not as good as the calculated values. In addition, both the average and median meet the requirements as an approximation of ideal values, and choosing one or the other makes no difference in the conclusion.

5.1.2 Regression or multi-class classification

During the analysis of the preprocessed multiple reference sentences in the ASSET dataset, it was observed that the optional values associated with the multiple reference sentences for a single complex sentence exhibited a discrete pattern. Despite rounding the value of control tokens to the nearest 0.05, it was noted that these values were not adjacent. Specifically, for a given complex sentence, the optional value of LengthRatio may range from 0.5 to 1.05 but may only include values such as 0.5, 0.85, 0.9, 1.0, and 1.05 in the reference sentences. Notably, there existed a substantial gap between 0.5 and 0.85, highlighting a distinct characteristic of the control token's behaviour. The discrete nature of the operations performed on the sentences, as evident from the observed values, presents a challenge in finding suitable options that seamlessly align with natural expression and grammar while making minimal adjustments to the control token. This discrete characteristic highlights the difficulty in identifying incremental changes that maintain the coherence and fluency of the sentence while effectively altering the sentence with control tokens.

Therefore, we approach the prediction of ideal control tokens from two different perspectives: as either a regressive problem or a single-label multi-classification problem. The main distinction between these two methods lies in how we define the value of the control tokens. As mentioned earlier, the available options for the control token values are limited and discontinuous. However, it is important to note that these values are interconnected and exhibit relationships with one another. When we treat the problem as a multi-class classification task, we effectively disregard the interconnections among the values and treat them as independent labels. While this approach allows us to predict the value of control tokens while disregarding the influence of adjacent values, it may overlook the impact of these interconnections. On the other hand, the regressive method simply treats the control token as a numerical value. However, this method carries the risk of encountering difficulties in capturing the subtle distinctions between the possible options and potentially falling into the gap between them.

5.2 Experiment Design

5.2.1 Preprocessing

Training set for control token	Number of sentences
DEPENDENCYTREEDEPTHRATIO	262,429
WORDRANKRATIO	295,779
REPLACEONLYLEVENSHTEIN	294,588
LENGTHRATIO	253,436

Table 5.1. Number of sentences in the training set for each control token (The original number of sentence pairs in Wikilarge (X. Zhang and Lapata 2017a) is 296,402)

In the preprocessing step, we applied similar procedures on Wikilarge (X. Zhang and Lapata 2017b) as mentioned in chapters 3 and 4 to append the control tokens. However, in this task, instead of combining all four control tokens into a single training set, we created four training set and each training set contains only one type of control token, in which the complex sentences are the model's input, while the corresponding control token values are the training targets. To increase data quality, we implemented a filtering process that removes values below 0.2 and above 1.5 (below 0.2 and above 1.0 to replace only the Levenshtein ratio). The detailed sentence number is listed in Table 5.1. As a result, we filtered 19,844 sentences on average for each training set.

5.2.2 Fine-turning

In the training process of predictors for each control token, we fine-tuned the BERT-base-uncased model on the filtered Wikilarge dataset(X. Zhang and Lapata 2017b). For comparison, we also tried two different models, distilledBERT (Sanh et al. 2019) and Roberta-large (Y. Liu et al. 2019), the detailed results are shown in Table 5.3. In the regressive method, we apply the regressive model, tokenize the target as plain text and retrieve the score directly without implementing any function to the model output. During training, we implement mean absolute error(MAE), root mean square error(RMSE) and coefficient of determination (R^2) metrics on the validation set and choose RMSE as the loss function. After 10 epochs of training, we keep the model with the lowest RMSE.

In contrast, when utilizing the classification approach, we employ the single-label multi-classification model. In this method, the target was tokenized into a vector consisting of 0s and 1s, and the softmax function was applied at the final layer of the model. To assess the model's performance, we choose the cross-entropy loss as the loss function. Similar to the regressive method, we trained the model for 10 epochs and selected the model with the lowest cross-entropy loss for further analysis and evaluation.

The detailed parameters applied in the training process are listed as below:

- learning rate = 2e-5,
- train batch size = 16,
- evaluation batch size 16,
- weight decay = 0.01,
- number of epoch = 5,
- fp16 = True,

5.2.3 Evaluation

In the evaluation step, we evaluate the predictors in two different ways. One is directly comparing the predicted value with the expectation, which is the mean value from all reference sentences. The other is applying the control tokens with predicted values to complex sentences and testing the SARI score and BERTScore for the output.

In the direct evaluation, we report the three common metrics for regression tasks as the indicator for both regressive and single-label multi-classification models:

- MAE: It is calculated by taking the absolute difference between each predicted value and its corresponding actual value and then averaging these differences. The resulting value represents the average absolute deviation of the predictions from the ground truth, providing a measure of how close the predictions are to the actual values on average. The equation is listed below:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \tag{5.1}$$

  where $\hat{y}_i$ represents the predicted value, y_i represents the actual value, and n is the total number of samples.

- RMSE: It is calculated by taking the square root of the mean of the squared differences between predicted and actual values. This helps to penalize larger errors more heavily, as the squared term amplifies the effect of larger deviations. The equation is listed below:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{5.2}$$

  where $\hat{y}_i$ represents the predicted value, y_i represents the actual value, and n is the total number of samples.

- R^2: It is a statistical measure that represents the proportion of the variance in the dependent variable (target variable) that can be explained by the independent variables (predictor variables) in a regression model. The R^2 value can range from negative infinity to 1. When it is non-positive, it indicates that the regression model captures none of the patterns or relationships present in the data, and the predictions are no better than simply using the mean value. When it is between 0 and 1, it indicates that the model explains a portion of the variability in the dependent variable. The higher the value is, the better the model is at explaining the variability. A value of 1 indicates that the model perfectly predicts the dependent variable using the independent variables. The equation is listed below:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5.3}$$

where $\hat{y}_i$ represents the predicted value, y_i represents the actual value, $\bar{y}$ represents the mean value and n is the total number of samples.

In the case of regressive models, the output value is rounded to the nearest 0.05 to provide a more concise and interpretable representation. Conversely, for classification models, the output comprises 27 labels, each associated with a confidence level (17 labels for the predictor of the replace-only Levenshtein ratio). To obtain a single output value, we consider the expectation of all labels and round it to the nearest 0.05, ensuring consistency with the representation used by the regressive models.

For reference, in addition to the performance metrics of the control token predictors, we include the average variance among all sentences. This variance is calculated based on the values of one complex sentence and ten reference sentences and is essential for the calculation of the R^2 score. Furthermore, to provide a comparative analysis, we present the same metrics for a sample predictor that was trained on unfiltered training sets. This comparison allows us to assess the impact of filtering the training data on the performance of the control token predictors. By incorporating these additional measures and comparisons, we gain a comprehensive understanding of the control token predictors' performance and their deviation from the reference sentences, enabling a more nuanced evaluation of their effectiveness. We also dive into the distribution and box plot of the predictors in the two methods, mean values and all values.

In terms of evaluating the SARI score and BERTScore, we conduct tests by applying the predicted values to both the single control token models and the combined control token model. Specifically, we selected the models that were trained using the joint tokenization strategy in Chapter 3 as the test bench. We test the predicted values and include the mean value as the reference point during the evaluation process. Furthermore, we also include the results obtained from the optimisation method as an additional reference for comparison and analysis.

5.3 Results

5.3.1 Performance of control token predictors

Control Token Predictor	Strategy	MAE $\downarrow$	RMSE $\downarrow$	$R^2 \uparrow$	Average Variance
DTD_predictor	regressive	0.117	0.149	0.508	0.045
	expectation	0.108	0.140	0.565	
	median	0.124	0.160	0.431	
WR_predictor	regressive	0.048	0.063	-0.204	0.003
	expectation	0.046	0.063	-0.198	
	median	0.046	0.063	-0.198	
LV_predictor	regressive	0.086	0.115	0.213	0.017
	expectation	0.071	0.094	0.470	
	median	0.093	0.116	0.198	
LR_predictor	regressive	0.098	0.125	0.527	0.033
	expectation	0.094	0.121	0.557	
	median	0.113	0.143	0.384	

Table 5.2. Performance of regressive control token predictors on the average value of ASSET test set (Average Variance means the average value of the variances calculated from different sentence pairs in ASSET test set).

Table 5.2 shows the performance of regression and expectation and median of multi-classification predictors along with the average variance from the reference sentences. The expectation is the weight product of all predictions with probability, while the median is the prediction that splits the probability distribution into half. The value of the MAE and RMSE is low (0.063–0.149) because the unit distance between the two adjacent control tokens is 0.05.

Upon examining the performance of the predictors for different control tokens, notable variations can be observed. The DTD_predictor exhibits the highest R^2 score among all four predictors, indicating the strongest ability to capture the relationship between the value of the control token and the corresponding input sentences. Following closely is the LR_predictor, which also demonstrates a respectable R^2 score. In contrast, the LV_predictor in the regression method performs noticeably worse compared to its counterpart in the classification method. The R^2 scores are positive for the predictors of dependency tree depth ratio, replace-only Levenshtein ratio, and length ratio in both methods, indicating their proficiency in capturing the desired relationship. However, the R^2 score is negative for the predictor of word rank ratio in both regression and classification methods, suggesting the failure of the proposed predictor to effectively capture the corresponding feature. Lastly, the performance of all classification predictors surpasses that of the regression predictors in the direct evaluation.

5.3.2 Distribution of all control tokens

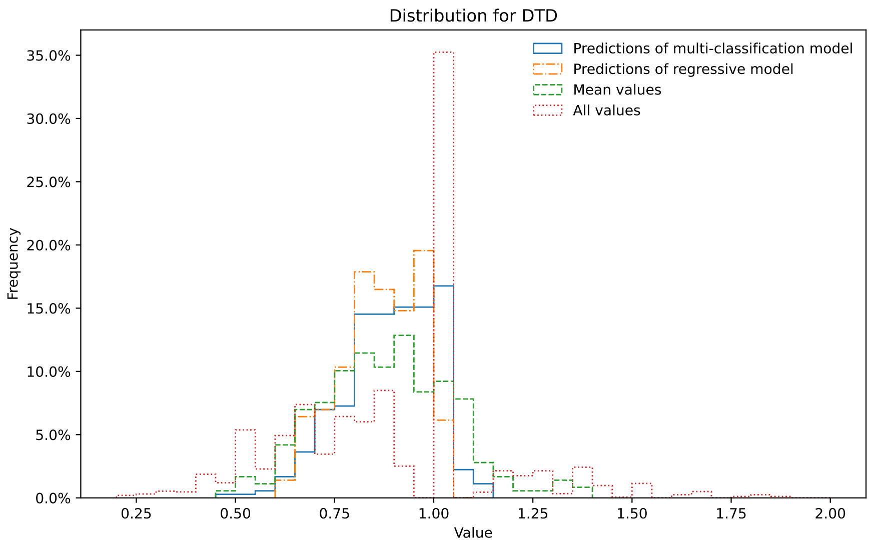


Figure 5.2. The density distribution of predictions, average and values of all reference sentences for DTD control token.

Figure 5.2 illustrates the density histograms of the predicted values, mean values, and all values derived from the reference sentences for the DTD. The histogram of all values in red dots exhibits a highly centralized impulse around the value of 1, with a noticeable gap on both sides. The remaining values predominantly occupy the left section of the distribution, ranging from 0.2 to 0.9. In contrast, the distribution of mean values, indicated by the green dashed line, demonstrates less centralization and a closer approximation to a normal distribution. The median value is approximately positioned near 0.85. Furthermore, by analyzing the predicted values, we observe that the distribution of the classification model, depicted by the blue solid line, shows a more evenly distributed pattern compared to the distribution of the regression model, illustrated by the orange dash-dot line. This may explain the reason for the better prediction of the mean values between the two methods.

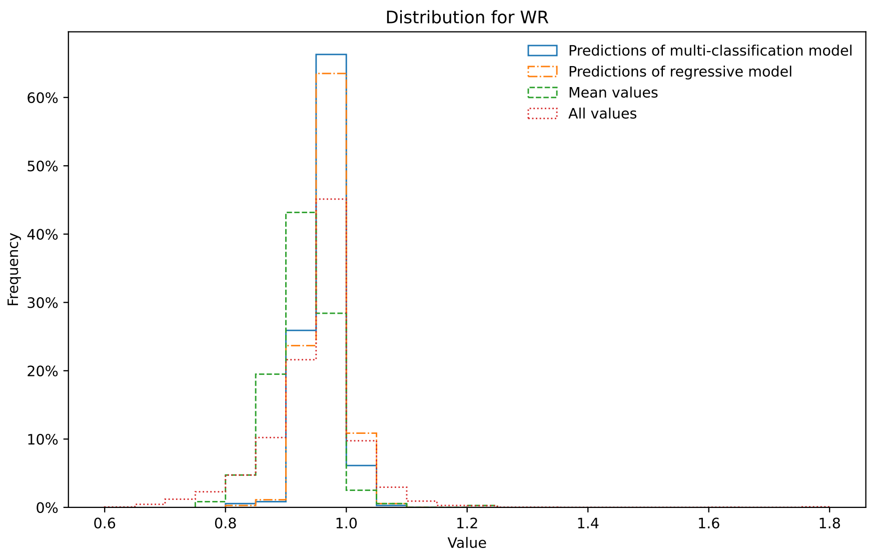


Figure 5.3. The density distribution of predictions, average and values of all reference sentences for WR control token.

Figure 5.3 shows the density histograms for the control token WR. The histograms of all values in the red dots and the mean values in the green dashed line exhibit a similar shape, with the median values tending to centralize around 0.9. However, both the distribution of the regression model and the classification model appear more concentrated around the value of 0.95. These observations regarding the distribution align with the R^2 scores presented in Table 5.2, indicating that the predictors struggle to capture the relationship between input sentences and the word rank ratio when compared to the overall means.

Figure 5.4 illustrates the density histograms for the control token LV. The histogram of all values in red dots exhibits a characteristic long tail, while the regression model and mean values share a similar shape. However, considering the R^2 scores provided in Table 5.2, it becomes evident that the classification model aligns more closely with the mean value. This discrepancy may be attributed to the left section

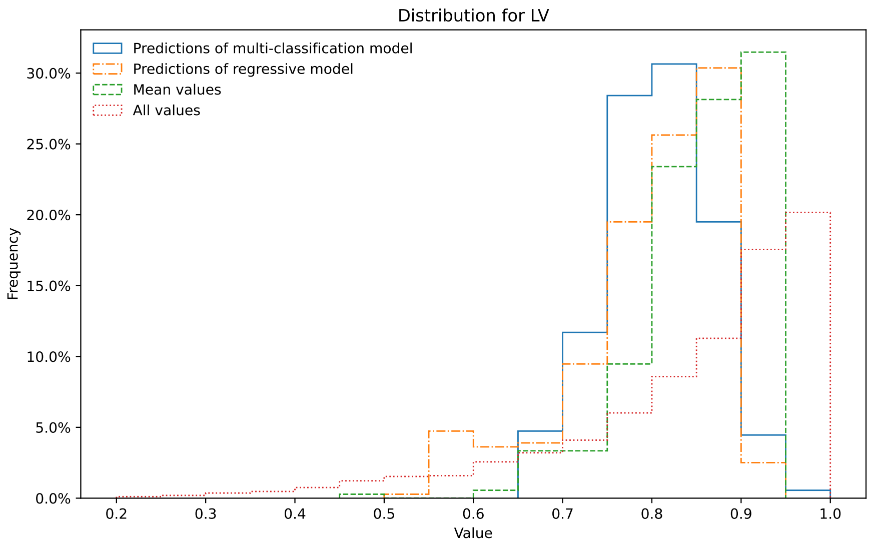


Figure 5.4. The density distribution of predictions, average and values of all reference sentences for LV control token.

of the regression model, which registers values lower than 0.7. The observed differences in the histograms clarify the performance of the predictors and their relation to the LV control token.

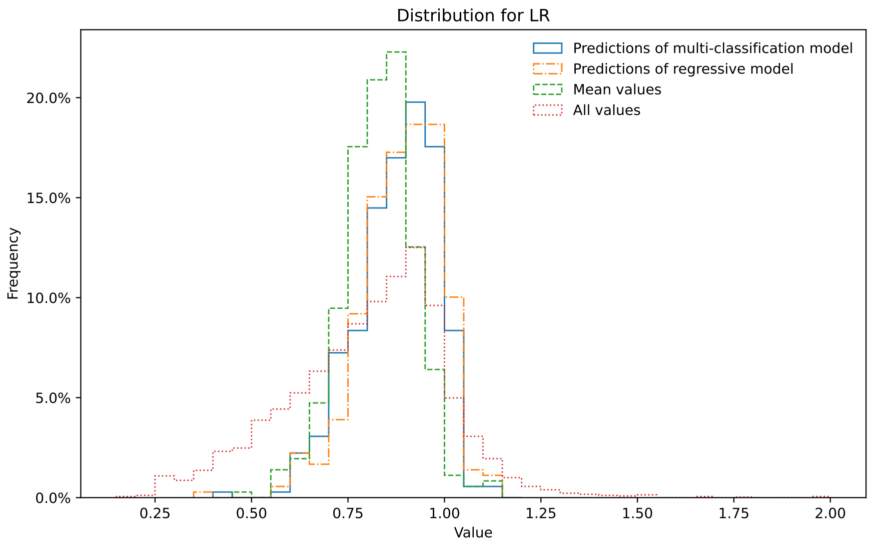
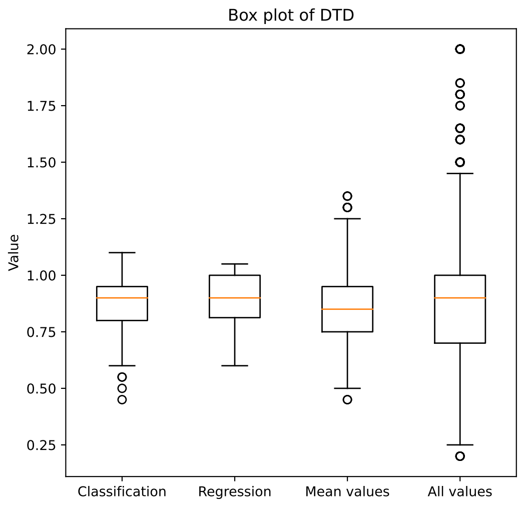
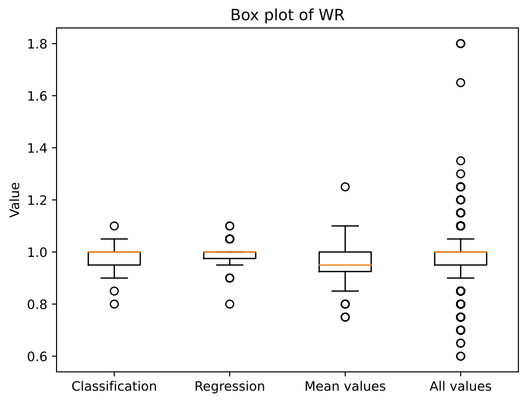


Figure 5.5. The density distribution of predictions, average and values of all reference sentences for LR control token.

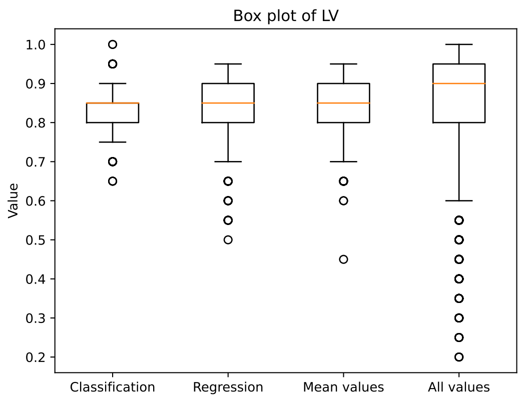
Figure 5.5 presents the density histograms depicting the control token LR. Analyzing the histogram of all means represented by the red dots, we observe that it exhibits a less centralized distribution in comparison to the other three histograms. However, despite this relatively dispersed nature, the mean value itself displays a more centralized shape compared to the predictors. Notably, the histogram of the classification model aligns closely with the mean value distribution, further reinforcing the consistency between these two distributions. This aligns with the distribution characteristics and the R^2 scores presented in Table 5.2.



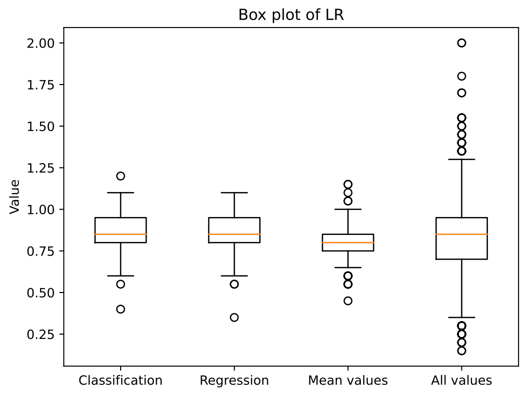
(a) The box plot of distributions for DTD control token.



(b) The box plot of distributions for WR control token.



(c) The box plot of distributions for LV control token.



(d) The box plot of distributions for LR control token.

Figure 5.6. The box plot of distributions of predictions, average and values of all reference sentences for the four control tokens.

In Figure 5.6, we display the box plots for different data groups of the four control tokens, illustrating various statistics such as maximum, upper quartile, median, lower quartile, minimum, and outliers. In Figure 5.6a, similar to Figure 5.2, all values have the most outliers and the largest range, while the predicted methods show higher concentration. The classification model overlaps more with mean values in the quartiles compared to the regression model.

Figure 5.6b shows box plots for control token WR, providing insights into its distribution. Similar to Figure 5.3, the box plot for all values overlaps more with mean values. However, there is a notable long tail due to outliers, as reflected in the box plots.

In Figure 5.6c, box plots for control token LV are examined. Consistent with Figure

5.4, the classification model has a more centralized distribution, closely overlapping with mean values. The regression model resembles the mean values, but the classification model performs better. There are numerous outliers in all values, indicating extreme data points.

Figure 5.6d presents box plots for control token LR. Although it is difficult to distinguish differences between the classification and regression models, the gap in performance compared to mean values is evident. The box plot for all values aligns with the histogram in Figure 5.5, showcasing diversity and variance.

5.3.3 Performance difference among models

Model	MAE $\downarrow$	RMSE $\downarrow$	$R^2 \uparrow$
BERT-base (Devlin et al. 2019)	**0.098**	**0.125**	**0.527**
distilBERT (Sanh et al. 2019)	0.108	0.134	0.458
Roberta-large (Y. Liu et al. 2019)	0.108	0.135	0.447
BERT-base with unfiltered training set	0.141	0.182	-0.008

Table 5.3. The performance of different regressive models on predicting Length Ratio on filtered or unfiltered training set.

In order to demonstrate the performance difference across various models, we present the results of regression models trained to predict the Length Ratio as an illustrative example. The reason for choosing the Length Ratio as the only example is that it has the highest average R^2 score and the most distinct distribution in all four control tokens. The performance comparison can be observed in Table 5.3, where it is evident that the BERT-base model (Devlin et al. 2019) outperforms the other two commonly used reference models in terms of both MAE and RMSE. To validate the efficacy of the data filtering process, we also include the performance of BERT trained on an unfiltered training set in the last row of the table. The observed performance improvement in the filtered dataset further confirms the effectiveness of the data filtering approach in enhancing the model's predictive capabilities.

	DTD	WR	LV	LR
Regression				
SARI ↑	36.10	32.75	40.64	37.19
BERTScore ↑	0.870	0.878	0.821	0.868
Expectation				
SARI ↑	36.66	32.67	40.95	37.69
BERTScore ↑	0.863	0.880	0.830	0.864
Median				
SARI ↑	35.73	32.31	38.92	36.48
BERTScore ↑	0.868	0.883	0.855	0.866
Average				
SARI ↑	37.66	35.78	41.16	39.88
BERTScore ↑	0.859	0.861	0.837	0.849
optimisation joint				
SARI ↑	40.71	41.42	42.86	40.64
BERTScore ↑	0.812	0.733	0.782	0.785

Table 5.4. Performance of single control token models with predictors on ASSET test (Regression and Classification methods refer to the single control models works with the control token predictors, average method means the control token value is the average value of reference sentences and optimisation method is the same as shown in Chapter 3).

Models	SARI ↑	BERTScore ↑
Regression	42.30	0.833
Expectation	42.76	0.828
Median (Predicted)	40.74	**0.850**
Median (Calculated)	44.56	0.836
Average (Calculated)	44.55	0.816
optimisation joint	**44.58**	0.794

Table 5.5. Performance of control token model with predictors on ASSET test. The top three rows are predicted values, and the bottom three rows are calculated or optimised values.

5.3.4 Performance with control token predictors

Table 5.4 shows the SARI and BERTScore of single control token models with different methods. It is clear that the performance gap between the classification models and regression models aligns with it in the direct evaluation. As a comparison, the scores of the average value calculated from the reference sentences and the optimised value found on the test set are added. Although the SARI scores of predictors and the average value are lower than the optimised ones, BERTScore remains higher. Among all the control tokens, the LV has the closest gap between the two types of methods.

Similar to Table 5.4, Table 5.5 shows the SARI and BERTScore with 4 predictors working cooperatively with the combined control token model. The performance difference still aligns with the previous table. However, the SARI score difference between the average value and optimisation method is very low, while the average value maintains a higher BERTScore, which shows the average value of multi-reference can be a proper approximation of the ideal value for each sentence.

5.3.5 Performance of hybrid method

Static control token	SARI ↑	BERTScore ↑
DTD fixed at 0.35	43.97	0.813
WR fixed at 0.7	43.62	0.809
LV fixed at 0.85	42.98	0.839
LR fixed at 0.9	41.30	0.846

Table 5.6. The performance with only one static value of the control token (The value is referred from Table 3.4).

Dynamic control token	SARI ↑	BERTScore ↑
DTD	42.75	**0.837**
WR	43.24	0.833
LV	44.16	0.816
LR	**44.61**	0.808

Table 5.7. The performance with only one dynamic value of the control token and the static value for the remaining control tokens are DTD:0.35, WR:0.7, LV:0.85 and LR:0.9 respectively (The value is referred from Table 3.4).

Considering the poor alignment and performance of single-control token models for some of the predictors, we conducted a hybrid method of both optimisation and prediction methods in Table 5.6 and 5.7. In Table 5.6, we replace the one predictor with the fixed value, referred from the optimisation results in Table 3.4, and found the impact of different control token predictors. It is clear that the DTD control token predictor has the biggest negative impact on the SARI score among all control token predictors, while the LR has the least. By replacing the DTD predictor with a fixed value of 0.35, we increased the SARI score from 42.76 to 43.97.

In Table 5.7, we replace one of the optimised values with predicted values from the classification method and found the performance differences with control token predictors. Notably, the one with the DTD predictor still shows the largest drop in the SARI score and the one with the LR predictor outperforms the optimisation method in both the SARI score and BERTScore.

5.3.6 Performance with Offsets

Model	SARI ↑	BERTScore ↑	Offsets
Prediction with offsets	**44.82**	0.784	DTD:-6 WR:-3 LV:-2 LR:0
Prediction method	42.76	**0.828**	N/A
optimisation method	44.58	0.794	N/A
Hybrid method with LR predictor	44.61	0.808	N/A

Table 5.8. The performance of prediction method with manual offset (x offset means adding $0.05 \cdot x$ to every predicted values of predictor models)

Compared to the optimisation method, we also conducted a manual offset on the predicted value to explore the potential peak performance of the prediction method. Along with the other referencing methods, the prediction method with manual offset outperforms the other three in the SARI score, which indicates the potential of this method and the limitation of some predictors as well. While the default prediction method maintains the best BERTScore. Similar to the results in 5.6 and 5.7, the DTD predictor has the highest offset value and we have to shift the value to left by 6 unit distance, which is minus 0.3 to every predicted value unless the shifted value is smaller than the lower limit at 0.2. As for the other three, the LR predictor works as intended, DTD and WR predictors require a reasonable offset only. One possible reason for the requirement of offset to reach the peak performance is that the model can hardly catch the relationship between the desired feature and the input sentences. However, it is also possible that these control tokens are poorly designed to learn or capture the feature of the sentences.

5.3.7 Migration to other datasets

Model	SARI ↑	BERTScore ↑	Offset
Prediction method with offset	45.77	0.624	DTD:-6 WR:-3 LV:-2 LR:0
Prediction method	**47.23**	**0.667**	N/A
optimisation method	44.19	0.664	N/A
Hybrid method with LR predictor	46.32	0.635	N/A

Table 5.9. The performance of multi-methods on PWKP test set

We tested the same setting without altering the offset values and optimised values in Table 5.8 on PWKP, Turk Corpus and Newsela test sets and reported the SARI

Model	SARI ↑	BERTScore ↑	Offset
Prediction method with offset	41.96	0.769	DTD:-6 WR:-3 LV:-2 LR:0
Prediction method	42.43	**0.832**	N/A
optimisation method	**42.90**	0.821	N/A
Hybrid method with LR predictor	42.01	0.798	N/A

Table 5.10. The performance of multi-methods on Turk Corpus test set

Model	SARI ↑	BERTScore ↑	Offset
Prediction method with offset	36.23	0.615	DTD:-6 WR:-3 LV:-2 LR:0
Prediction method	32.98	**0.644**	N/A
optimisation method	33.66	0.623	N/A
Hybrid method with LR predictor	**36.23**	0.622	N/A

Table 5.11. The performance of multi-methods on Newsela test set

and BERTScore in Table 5.9, 5.10 and 5.11. In Table 5.9, the prediction method holds the highest SARI score and BERTScore, while in Table 5.10, the optimisation method has the highest SARI score and prediction shows the highest BERTScore. Notably, the hybrid method performs worse than the optimisation method in the Turk Corpus test set. As for the Newsela test set, we found the best SARI score from both hybrid offset methods. In addition, the default prediction method holds the best BERTScore, which indicates the limitations of directly applying the optimisation method to different datasets and verifies the improvement on the practicability.

5.4 Case Study

5.4.1 Instances

Tables 5.12–5.14 show several examples of simplifications produced by optimisation and prediction methods and the output with average value as the reference on the test set of the ASSET. All three outputs are from the same model with different values of control tokens listed in the second column.

In Table 5.12, the optimisation method shows a typical factual error because of the fixed values in control tokens. There is no relationship between ’website’ and ’magazine’ related to ’taken over’ in the source sentence. However, due to the LR and LV ratios being fixed on the whole test set, the simplification from the optimi-

Source	The name survives as a brand for a related spin-off digital television channel, digital radio station, and website which have survived the demise of the printed magazine.			
Method	Control token	Output	SARI	BERTScore
optimisation	DTD_0.35 WR_0.85 LV_0.8 LR_0.85	The name survives as a brand for a related digital television channel, digital radio station, and website which have not been taken over by the magazine.	43.49	0.781
Prediction	DTD_0.6 WR_1.0 LV_0.8 LR_0.7	The name survives as a brand for a digital television channel, digital radio station, and website which have also survived.	43.48	0.752
Average	DTD_0.65 WR_0.95 LV_0.9 LR_0.85	The name survives as a brand for a related digital television channel, digital radio station, and website which have survived without the magazine.	41.37	0.779

Table 5.12. Examples of limitations of the optimisation method (changed the meaning in the third row).

Source	A few animals have a chromatic response, changing color in changing environments, either seasonally (ermine, snowshoe hare) or far more rapidly with chromatophores in their integument (the cephalopod family).			
Method	Control token	Output	SARI	BERTScore
optimisation	DTD_0.35 WR_0.85 LV_0.8 LR_0.85	A few animals have chromatic response. They change color in changing environments, either seasonally (ermine, snowshoe hare) or far more rapidly (the cephalopod family).	30.19	0.724
Prediction	DTD_0.85 WR_0.95 LV_0.9 LR_0.6	A few animals have chromatic response. They change color in changing environments, either seasonally (ermine, snowshoe hare).	36.61	0.621
Average	DTD_0.8 WR_0.8 LV_0.75 LR_0.8	A few animals have chromatic response, changing color in changing environments, either seasonally (ermine, snowshoe hare) or more rapidly (the cephalopod family).	25.36	0.680

Table 5.13. Examples with mispredicted values (missing 'or' clause in the third row).

Source	Since the end of the 19th century Eschelbronn is well known for its furniture manufacturing industry.			
Method	Control token	Output	SARI	BERTScore
optimisation	DTD_0.35 WR_0.85 LV_0.8 LR_0.85	Since the end of the 19th century, Eschelbronn has become very famous for its furniture.	43.86	0.822
Prediction	DTD_0.85 WR_1.05 LV_0.9 LR_0.8	Since the end of the 19th century Eschelbronn is famous for its furniture making.	39.85	0.920
Average	DTD_0.85 WR_0.95 LV_0.9 LR_0.8	Since the end of the 19th century Eschelbronn is famous for its furniture making.	39.85	0.920

Table 5.14. Examples with properly predicted values.

sation method has to maintain a longer and more different sequence than the prediction method in this case, which makes it tend to generate extra content to fulfil the requirements. As a result, it generates false content and changes the meaning of the source sentence, which will mislead readers. This could be more common in larger and more diverse datasets. Although the output of the prediction method also changed the original meaning, it just loses some meaning without adding false content. The output of the average value best preserves the meaning in this example.

In Table 5.13, we show the consequence of misprediction. The output of the optimisation method and the average value is quite similar except the length of the optimisation method is a bit longer. In the prediction method, the output sentence is incomplete due to the lower length ratio compared to the average value. Although there is also a gap in the DTD ratio in optimisation and prediction methods, there seems to be no obvious change in the syntactical complexity, which is aligned with the limitations mentioned in previous sections.

While in Table 5.14, we show an example of desired predictions. Although there are some small variances in the value of control tokens, the output of the prediction method is identical to the output of the average value, which is used as the reference sentence. However, the optimisation method changes the original meaning in the source sentence.

5.4.2 Mistakes and Analysis

We manually check the system outputs between the official system output of BART trained on the Wikilarge from MUSS and ours. In Table 5.15, we listed the findings and limitations of both models. The major differences include the following types:

- Shorter and separate sentences: We find many examples showing that our system prefers to split a long sentence into several shorter sentences. This may benefit aphasia people, who find it hard to follow very long sentences. On the other hand, it may also set obstacles to people having difficulties understanding the pronouns.
- Hallucination and false content: Both systems still generate hallucination and false content, which are inconsistent with the inputs. These false contents will

undermine the practicality of the simplification systems. However, some of the false contents are caused by the control tokens themselves and can be alleviated by giving proper input value or a new control mechanism.

- Omitting contents: We also found that the two systems might have different preferences in deleted contents. For now, there lack of proper control mechanisms in control tokens to decide which part is the key information in the sentence and needs further improvement.

Insights	System outputs (MUSS followed by the best hybrid method)
Our system's output tends to generate shorter sentences with demonstrative pronouns.	They are castrated so that the animal may be more docile and put on weight more quickly. They are castrated so that the animal may be more docile. <u>It</u> may put on weight more quickly.
Both systems show hallucination and false content in some way. In the example, MUSS falsely predicted the meaning of 'It'.	<u>Stralsund</u> is located on the coast of the Baltic Sea, near the city of Stralsund. It is located on the Baltic Sea. The city of Stralsund is nearby.
Both systems show hallucination and false content in some way. In the example, our system fails to simplify the 'extremely competitive' with 'important'.	Admission to Tsinghua is very difficult. Admission to Tsinghua is very <u>important</u>.
In this example, our system shows a more readable output by sentence splitting and replacing 'Public Broadcasting Service' with 'television'.	She performed for President Reagan in 1988's Great Performances at the White House series on the Public Broadcasting Service. She performed for President Ronald Reagan in 1988's Great Performances at the White House series. <u>The series</u> was shown on <u>television</u>.
Both system outputs omitted some information, 'motor racing championship' in MUSS and 'Brecia' in ours.	The first Italian Grand Prix took place on September 4, 1921 at <u>Brescia</u> in Italy. The first Italian Grand Prix <u>motor racing championship</u> was held on September 4, 1921.

Table 5.15. More insights in the two systems

5.5 Discussion and Future Work

5.5.1 Overall Performance

In Table 5.2, the prediction of WR is less applicable in the four control tokens. In addition, in Table 5.4, both the regression model and classification model for DTD and WR show a much lower SARI score than the optimisation method, which indicates the average value may not be an ideal approximation to the ideal values or the models fail to generate ideal simplification with the approximated ideal values. While both methods for the other control tokens demonstrate decent performance in the SARI score.

In Table 5.5, we show that the SARI score of the predictor methods and average value for control tokens are lower than the optimisation method in the SARI score. However, the BERTScore is significantly higher, which is reasonable since the goal set in the optimisation method is to maximise the SARI score only. However, we would like to emphasise the limitations of the SARI score. A sentence with a higher SARI score is not necessarily more meaningful at the sentence level, because the SARI score focuses on the word level operations. It is possible for a sentence full of similar add, keep and delete operations to the reference with complete unreadable order will still have a high SARI score. If the only goal is to chase the SARI score, the model may generate some meaningless content.

5.5.2 Hybrid Method

In Table 5.6 and 5.7, we tested the hybrid methods with either the prediction method or optimisation method to evaluate the effect of certain control token predictors. Both tables show poor performance with the DTD predictor and WR predictor. However, unlike the poor alignment of WR in Table 5.2, the DTD demonstrates good performance in the R^2 score. The main reason for the poor performance with the DTD predictor may be that this control token is not well designed to reflect the attribute of sentences or it is not suitable for the model to learn. In addition, the optimisation method with the LR predictor outperforms the original optimisation method.

In Table 5.8, we tried to manually adjust the predicted value to maximise the SARI score. The offsets of four control tokens reflect similar results to the four control token predictors that the DTD predictor deviates from the expectation the most. However, with manual tweaks, the prediction method can outperform the other methods, indicating the potential of this method.

5.5.3 Migration to other datasets

Due to the lack of multi-reference and the small size of the PWKP, the SARI score can be more diverse and extreme. In addition, there might be an overlap between the training data for the predictor and the test set, so that the predicted value is closer to the ideal value. Nonetheless, the prediction method outperforms the optimisation method by a large margin. The lower performance of the offset method shows the predictors can better predict the value on the Wikipedia-based test set. However, in Turk Corpus, due to the similarity between this test set and the ASSET, the prediction method fails to outperform the optimisation method in the SARI score, but it still maintains a higher BERTScore. Lastly, the Newsela test set with its unique genre and manual simplifications shows the universality of the prediction method, which outperforms the optimisation method with offset. However, it also proves that the predictors need more diverse training materials to better adapt to different datasets. Tables 5.9, 5.10 and 5.11 demonstrate the adaptation ability by directly applying the several methods with control token predictors on different datasets.

5.5.4 Future Work

As shown in previous sections, both the predictors and the average values of DTD and WR improves the SARI score marginally, which might be because of the design of these two control token. If some of the control tokens can hardly fulfil the design purpose, it might be worth building some new control tokens or fulfilling the designed goal differently. Another task is to extend the application scenario and training material of control token predictors. For now, the predictor learns only from the Wikilarge dataset, and can hardly serve the purpose of giving precise predictions based on the needs of different user groups. To better achieve the goal of customised simplification for different user groups, further improvement in the control token predictor mechanism is needed.

5.6 Conclusion

In this chapter, we proposed a new method with control token predictors to improve the generalization ability of the controllable text simplification systems. As a conclusion, we tested the performance of predictors of different control tokens, pointed the limitation of current control token predictors, increased the performance of current controllable text simplification system with control token predictors with and without optimised values.

Chapter 6

Final Remarks

6.1 Conclusion

In this thesis, we answered the three research questions in Chapters 3, 4 and 5. In Chapter 3, we give a thorough analysis of the single control tokens in SARI score, BERTScore and case study to find how they affect the results in different aspects. We reported the limitation of single control tokens and the effectiveness of the combination of all control tokens. In addition, during the reimplementation of the SOTA, we optimised the tokenization strategy and quantisation method and hence improved the SARI score by 0.5 on the ASSET test set.

In Chapter 4, we designed a genre-specific test scenario for text simplification systems from the Simple-TICO 19 dataset and conducted detailed experiments on the two types of models. In the experiment, we verified the effect of transfer learning of controllable text simplification systems in certain domains and compared these expert models with LLM such as ChatGPT. As a result, we analysed the results and thought that the genre-specific or domain-specific models still play an important role in the corresponding tasks.

In Chapter 5, we proposed a new approach to leverage the control tokens to increase the practicability of the controllable text simplification systems. We investigated the performance of control token predictors and compared them with the optimization method. We found that the newly proposed method can generate comparable or even better output than the previous system without requiring further information for the new dataset, which is crucial in real-world applications.

As a conclusion, we made the following contributions in this thesis: providing an easy-to-use implementation of the SOTA text simplification model; investigating the effects of control tokens in both evaluations and practice; optimising the tokenization strategy to improve the performance; proposing a genre-specific test scenario for text simplification system; proving the effectiveness of transfer learning on the domain-specific task with control token mechanism; comparing the performance of the genre-specific fine-tuned model with LLMs in different situations; proposing a different pattern of levering the control tokens for text simplification systems; verifying the generalization and adaptation ability of newly-proposed system.

6.2 Discussion and Future Work

In this section, we would like to point out the problems of current seq-2-seq text simplification systems and discuss potential solutions for the problems. During our experiments, we found two major issues with the current test simplification systems:

One major issue to address is the hallucination and misinformation problem, as shown in table 3.9, the enforced controlling of the output sentence may lead to misinformation, especially when given improper value of control tokens. Similar issues happen to the LLMs as well (M. Zhang et al. 2023; Lee, Bubeck, and Petro 2023), which could cause severe problems and hinder the application and popularity of NLP systems. Many of the hallucination problems are presumably caused by the lack of knowledge base in the system, which was solely trained on a general dataset. However, it is impractical to train an LLM in every domain, thus we think one of the ways to alleviate the problem is by combining the external knowledge base with current LLMs.

The other issue is the lack of conceptual simplifications in the current text simplification systems. Although it is possible to ask the model about the definition or explanation of terms or jargon or force the model to extend the simplification with control tokens and expect to give a proper explanation to the complex words, these operations can hardly be done in one run and require further identification or tagging on the target word. Given the condition, we consider one solution is decomposing the sentences into triplets and implementing text simplification on the triplets level to identify the target word better.

Based on the discussion mentioned above, we plan to build a controllable text simplification system with a knowledge base focusing on the triplets level. In order to achieve this goal, we plan to introduce the knowledge graph as the external knowledge base and develop a decompose-recompose system to cooperate with the knowledge graph. There are three steps in the system, the first is to break the sentences into triplets, the second is to combine the triplets with an external knowledge graph and implement the lexical and conceptual simplification, and the last step is to recover the modified triplets into sentences with a syntactical simplification. With the help of this system, the two major problems can be alleviated and solved with further research and development.

References

Alva-Manchego, Fernando, Louis Martin, Antoine Bordes, et al. (July 2020). “ASSET: A Dataset for Tuning and Evaluation of Sentence Simplification Models with Multiple Rewriting Transformations”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 4668–4679. DOI: 10.18653/v1/2020.acl-main.424. URL: https://aclanthology.org/2020.acl-main.424.

Alva-Manchego, Fernando, Louis Martin, Carolina Scarton, et al. (Nov. 2019). “EASSE: Easier Automatic Sentence Simplification Evaluation”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*. Hong Kong, China: Association for Computational Linguistics, pp. 49–54. DOI: 10.18653/v1/D19-3009. URL: https://aclanthology.org/D19-3009.

Alva-Manchego, Fernando, Carolina Scarton, and Lucia Specia (2020). “Data-Driven Sentence Simplification: Survey and Benchmark”. In: *Computational Linguistics* 46.1, pp. 135–187. DOI: 10.1162/coli_a_00370. URL: https://aclanthology.org/2020.cl-1.4.

– (Dec. 2021). “The (Un)Suitability of Automatic Evaluation Metrics for Text Simplification”. In: *Computational Linguistics* 47.4, pp. 861–889. DOI: 10.1162/coli_a_00418. URL: https://aclanthology.org/2021.cl-4.28.

Amancio, Marcelo and Lucia Specia (Apr. 2014). “An Analysis of Crowdsourced Text Simplifications”. In: *Proceedings of the 3rd Workshop on Predicting and Im-*

proving Text Readability for Target Reader Populations (PITR). Gothenburg, Sweden: Association for Computational Linguistics, pp. 123–130. DOI: 10.3115/v1/W14-1214. URL: https://aclanthology.org/W14-1214.

Anastasopoulos, Antonios et al. (Dec. 2020). “TICO-19: the Translation Initiative for COvid-19”. In: *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*. Online: Association for Computational Linguistics. DOI: 10.18653/v1/2020.nlpcovid19-2.5. URL: https://aclanthology.org/2020.nlpcovid19-2.5.

Barzilay, Regina and Noemie Elhadad (2003). “Sentence Alignment for Monolingual Comparable Corpora”. In: *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pp. 25–32. URL: https://aclanthology.org/W03-1004.

Brown, Tom et al. (2020). “Language Models are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., pp. 1877–1901. URL: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Bruce, Bertram, Andee Rubin, and Kathleen Starr (1981). “Why readability formulas fail”. In: *IEEE Transactions on Professional Communication* 1, pp. 50–52.

Callison-Burch, Chris, Miles Osborne, and Philipp Koehn (Apr. 2006). “Re-evaluating the Role of Bleu in Machine Translation Research”. In: *11th Conference of the European Chapter of the Association for Computational Linguistics*. Trento, Italy: Association for Computational Linguistics, pp. 249–256. URL: https://aclanthology.org/E06-1032.

Carroll, John et al. (1998). “Practical simplification of English newspaper text to assist aphasic readers”. In: *Proceedings of the AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*. Association for the Advancement of Artificial Intelligence, pp. 7–10.

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

Chandrasekar, Raman and Bangalore Srinivas (1997). “Automatic induction of rules for text simplification”. In: *Knowledge-Based Systems* 10.3, pp. 183–190.

Coleman, Meri and Ta Lin Liau (1975). “A computer readability formula designed for machine scoring.” In: *Journal of Applied Psychology* 60.2, p. 283.

Coster, William and David Kauchak (June 2011). “Simple English Wikipedia: A New Text Simplification Task”. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, pp. 665–669. URL: https://aclanthology.org/P11-2117.

Crystal, David (1987). “The Cambridge encyclopedia oflanguage”. In: *UK: Cambridge University*.

De Belder, Jan and Marie-Francine Moens (2010). “Text simplification for children”. In: *Prroceedings of the SIGIR workshop on accessible search systems*. ACM; New York, pp. 19–26.

Devaraj, Ashwin et al. (May 2022). “Evaluating Factuality in Text Simplification”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, pp. 7331–7345. DOI: 10.18653/v1/2022.acl-long.506. URL: https://aclanthology.org/2022.acl-long.506.

Devlin, Jacob et al. (June 2019). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. 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)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

Dolch, Edward William (1928). “Vocabulary burden”. In: *The Journal of Educational Research* 17.3, pp. 170–183.

Dong, Yue et al. (2019). “EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing”. In: *arXiv preprint arXiv:1906.08104*.

DuBay, William H (2004). “The principles of readability.” In: *Online Submission*.

Engzell, P, A Frey, and M Verhagen (2020). “The collateral damage to children’s education during lockdown”. In: *VOXeu CEPR Policy Portal*.

Fedus, William, Barret Zoph, and Noam Shazeer (2021). “Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity”. In: *CoRR* abs/2101.03961. arXiv: 2101.03961. URL: https://arxiv.org/abs/2101.03961.

Feng, Yutao et al. (2023). “Sentence simplification via large language models”. In: *arXiv preprint arXiv:2302.11957*.

Flesch, Rudolph (1948). “A new readability yardstick.” In: *Journal of applied psychology* 32.3, p. 221.

Floridi, Luciano and Massimo Chiriatti (2020). “GPT-3: Its nature, scope, limits, and consequences”. In: *Minds and Machines* 30.4, pp. 681–694.

Ganitkevitch, Juri, Benjamin Van Durme, and Chris Callison-Burch (June 2013). “PPDB: The Paraphrase Database”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, pp. 758–764. URL: https://aclanthology.org/N13-1092.

Grave, Edouard et al. (2018). “Learning Word Vectors for 157 Languages”. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Guo, Han, Ramakanth Pasunuru, and Mohit Bansal (2018). “Dynamic Multi-Level Multi-Task Learning for Sentence Simplification”. In: *CoRR* abs/1806.07304. arXiv: 1806.07304. URL: http://arxiv.org/abs/1806.07304.

Harris, Zellig S (1954). “Distributional structure”. In: *Word* 10.2-3, pp. 146–162.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780.

Honnibal, Matthew et al. (2020). “spaCy: Industrial-strength natural language processing in python”. In.

Hwang, William et al. (2015). “Aligning Sentences from Standard Wikipedia to Simple Wikipedia”. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, pp. 211–217. DOI: 10.3115/v1/N15-1022. URL: https://aclanthology.org/N15-1022.

Kajiwara, Tomoyuki and Mamoru Komachi (Dec. 2016). “Building a Monolingual Parallel Corpus for Text Simplification Using Sentence Similarity Based on Alignment between Word Embeddings”. In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 1147–1158. URL: https://aclanthology.org/C16-1109.

Karreman, Joyce, Thea Van der Geest, and Esmee Buursink (2007). “Accessible website content guidelines for users with intellectual disabilities”. In: *Journal of applied research in intellectual disabilities* 20.6, pp. 510–518.

Kauchak, David (Aug. 2013). “Improving Text Simplification Language Modeling Using Unsimplified Text Data”. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Sofia, Bulgaria: Association for Computational Linguistics, pp. 1537–1546. URL: https://aclanthology.org/P13-1151.

Klare, George Roger et al. (1963). “Measurement of readability”. In.

Klein, Guillaume et al. (2017). “Opennmt: Open-source toolkit for neural machine translation”. In: *arXiv preprint arXiv:1701.02810*.

L’ALLIER, James J (1981). “AN EVALUATIVE STUDY OF A COMPUTER-BASED LESSON THAT ADJUSTS READING LEVEL BY MONITORING ON-TASK READER CHARACTERISTICS.” In.

Lee, Peter, Sebastien Bubeck, and Joseph Petro (2023). “Benefits, Limits, and Risks of GPT-4 as an AI Chatbot for Medicine”. In: *New England Journal of Medicine* 388.13, pp. 1233–1239.

Lester, Brian, Rami Al-Rfou, and Noah Constant (2021). *The Power of Scale for Parameter-Efficient Prompt Tuning*. arXiv: 2104.08691 [cs.CL].

Lewis, Mike et al. (July 2020). “BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703. URL: https://aclanthology.org/2020.acl-main.703.

Linderholm, Tracy et al. (2000). “Effects of causal text revisions on more-and less-skilled readers’ comprehension of easy and difficult texts”. In: *Cognition and Instruction* 18.4, pp. 525–556.

Liu, Pengfei et al. (2023). “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing”. In: *ACM Computing Surveys* 55.9, pp. 1–35.

Liu, Xiao et al. (2022). “P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 61–68.

Liu, Yinhan et al. (2019). “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692*.

Lively, Bertha A and Sidney L Pressey (1923). “A method for measuring the vocabulary burden of textbooks”. In: *Educational administration and supervision* 9.7, pp. 389–398.

Lu, Junru et al. (2023). “Napss: Paragraph-level medical text simplification via narrative prompting and sentence-matching summarization”. In: *arXiv preprint arXiv:2302.05*

Lu, Xinyu et al. (Nov. 2021a). “An Unsupervised Method for Building Sentence Simplification Corpora in Multiple Languages”. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 227–237. DOI: 10.18653/v1/2021.findings-emnlp.22. URL: https://aclanthology.org/2021.findings-emnlp.22.

– (2021b). *An Unsupervised Method for Building Sentence Simplification Corpora in Multiple Languages*. arXiv: 2109.00165 [cs.CL].

Ma, Yuan, Sandaru Seneviratne, and Elena Daskalaki (Dec. 2022). “Improving Text Simplification with Factuality Error Detection”. In: *Proceedings of the Workshop on Text Simplification, Accessibility, and Readability (TSAR-2022)*. Abu Dhabi, United Arab Emirates (Virtual): Association for Computational Linguistics, pp. 173–178. URL: https://aclanthology.org/2022.tsar-1.16.

Maldonado, Joana Elisa and Kristof De Witte (2022). “The effect of school closures on standardised student test outcomes”. In: *British Educational Research Journal* 48.1, pp. 49–94.

Martin, Louis, Angela Fan, et al. (2020). “Multilingual Unsupervised Sentence Simplification”. In: *CoRR* abs/2005.00352. arXiv: 2005.00352. URL: https://arxiv.org/abs/2005.00352.

Martin, Louis, Samuel Humeau, et al. (2019). “Reference-less quality estimation of text simplification systems”. In: *arXiv preprint arXiv:1901.10746*.

Martin, Louis, Benoı□t Sagot, et al. (2019). “Controllable Sentence Simplification”. In: *CoRR* abs/1910.02677. arXiv: `1910.02677`. URL: `http://arxiv.org/abs/1910.02677`.

Mc Laughlin, G Harry (1969). “SMOG grading-a new readability formula”. In: *Journal of reading* 12.8, pp. 639–646.

McNamara, Danielle S et al. (1996). “Are good texts always better? Interactions of text coherence, background knowledge, and levels of understanding in learning from text”. In: *Cognition and instruction* 14.1, pp. 1–43.

Mikolov, Tomas et al. (2013). “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781*.

Munkhdalai, Tsendsuren and Hong Yu (2017). “Neural semantic encoders”. In: *Proceedings of the conference. Association for Computational Linguistics. Meeting*. Vol. 1. NIH Public Access, p. 397.

Nisioi, Sergiu et al. (July 2017). “Exploring Neural Text Simplification Models”. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Vancouver, Canada: Association for Computational Linguistics, pp. 85–91. DOI: `10.18653/v1/P17-2014`. URL: `https://aclanthology.org/P17-2014`.

Noordman, Leo GM and Wietske Vonk (1992). “Readers’ knowledge and the control of inferences in reading”. In: *Language and Cognitive Processes* 7.3-4, pp. 373–391.

OECD (Oct. 2013). *OECD Skills Outlook 2013*. OECD. DOI: `10.1787/9789264204256-en`. URL: `https://doi.org/10.1787/9789264204256-en`.

– (Dec. 2019). *PISA 2018 Results (Volume I)*. OECD. DOI: `10.1787/5f07c754-en`. URL: `https://doi.org/10.1787/5f07c754-en`.

Ogden, Charles Kay (1930). “Basic English: A general introduction with rules and grammar”. In.

Omelianchuk, Kostiantyn, Vipul Raheja, and Oleksandr Skurzhanskyi (2021). “Text Simplification by Tagging”. In: *CoRR* abs/2103.05070. arXiv: 2103.05070. URL: https://arxiv.org/abs/2103.05070.

Orăsan, Constantin, Richard Evans, and Ruslan Mitkov (2018). “Intelligent text processing to help readers with autism”. In: *Intelligent Natural Language Processing: Trends and Applications*, pp. 713–740.

Paetzold, Gustavo Henrique (2016). “Lexical simplification for non-native english speakers”. PhD thesis. University of Sheffield.

Papineni, Kishore et al. (July 2002). “Bleu: a Method for Automatic Evaluation of Machine Translation”. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://aclanthology.org/P02-1040.

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

Pavlick, Ellie, Pushpendre Rastogi, et al. (July 2015). “PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, pp. 425–430. DOI: 10.3115/v1/P15-2070. URL: https://aclanthology.org/P15-2070.

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

Peters, Matthew E. et al. (June 2018). “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: `10.18653/v1/N18-1202`. URL: `https://aclanthology.org/N18-1202`.

Petersen, Sarah E and Mari Ostendorf (2007). “Text simplification for language learners: a corpus analysis”. In: *Workshop on speech and language technology in education*. Citeseer.

Qi, Peng et al. (2020). *Stanza: A Python Natural Language Processing Toolkit for Many Human Languages*. arXiv: `2003.07082 [cs.CL]`.

Radford, Alec et al. (2019). “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8, p. 9.

Raffel, Colin et al. (2019). “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *CoRR* abs/1910.10683. arXiv: `1910.10683`. URL: `http://arxiv.org/abs/1910.10683`.

– (2020). “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *Journal of Machine Learning Research* 21.140, pp. 1–67. URL: `http://jmlr.org/papers/v21/20-074.html`.

Rajpurkar, Pranav et al. (Nov. 2016). “SQuAD: 100,000+ Questions for Machine Comprehension of Text”. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 2383–2392. DOI: `10.18653/v1/D16-1264`. URL: `https://aclanthology.org/D16-1264`.

Rapin, J. and O. Teytaud (2018). *Nevergrad - A gradient-free optimization platform*. https://GitHub.com/FacebookResearch/Nevergrad.

Rello, Luz et al. (2013). “Frequent words improve readability and short words improve understandability for people with dyslexia”. In: *Human-Computer Interaction–INTERACT 2013: 14th IFIP TC 13 International Conference, Cape Town, South Africa, September 2-6, 2013, Proceedings, Part IV 14*. Springer, pp. 203–219.

Robert, Gunning (1952). “The technique of clear writing”. In: *New York*.

Sanh, Victor et al. (2019). “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *arXiv preprint arXiv:1910.01108*.

Scarton, Carolina and Lucia Specia (July 2018). “Learning Simplifications for Specific Target Audiences”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 712–718. DOI: 10.18653/v1/P18-2113. URL: https://aclanthology.org/P18-2113.

Schwartz, Lane et al. (June 2014). “Machine Translation and Monolingual Postediting: The AFRL WMT-14 System”. In: *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Baltimore, Maryland, USA: Association for Computational Linguistics, pp. 186–194. DOI: 10.3115/v1/W14-3321. URL: https://aclanthology.org/W14-3321.

Scialom, Thomas et al. (2021). “Rethinking Automatic Evaluation in Sentence Simplification”. In: *CoRR* abs/2104.07560. arXiv: 2104.07560. URL: https://arxiv.org/abs/2104.07560.

Shardlow, Matthew and Fernando Alva-Manchego (June 2022). “Simple TICO-19: A Dataset for Joint Translation and Simplification of COVID-19 Texts”. In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, pp. 3093–3102. URL: https://aclanthology.org/2022.lrec-1.331.

Sheang, Kim Cheng and Horacio Saggion (Aug. 2021). “Controllable Sentence Simplification with a Unified Text-to-Text Transfer Transformer”. In: *Proceedings of the 14th International Conference on Natural Language Generation*. Aberdeen, Scotland, UK: Association for Computational Linguistics, pp. 341–352. URL: `https://aclanthology.org/2021.inlg-1.38`.

Sherman, Lucius Adelno (1893). *Analytics of literature: A manual for the objective study of English prose and poetry*. Ginn.

Siddharthan, Advaith (2006). “Syntactic simplification and text cohesion”. In: *Research on Language and Computation* 4, pp. 77–109.

Stajner, Sanja (Aug. 2021). “Automatic Text Simplification for Social Good: Progress and Challenges”. In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Online: Association for Computational Linguistics, pp. 2637–2652. DOI: `10.18653/v1/2021.findings-acl.233`. URL: `https://aclanthology.org/2021.findings-acl.233`.

Sulem, Elior, Omri Abend, and Ari Rappoport (2018). “BLEU is Not Suitable for the Evaluation of Text Simplification”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 738–744. DOI: `10.18653/v1/D18-1081`. URL: `https://aclanthology.org/D18-1081`.

Surya, Sai et al. (July 2019). “Unsupervised Neural Text Simplification”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 2058–2068. DOI: `10.18653/v1/P19-1198`. URL: `https://aclanthology.org/P19-1198`.

Tanprasert, Teerapaun and David Kauchak (Aug. 2021). “Flesch-Kincaid is Not a Text Simplification Evaluation Metric”. In: *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*. Online: Association for Computational Linguistics, pp. 1–14. DOI: 10.18653/v1/2021.gem-1.1. URL: https://aclanthology.org/2021.gem-1.1.

Thoppilan, Romal et al. (2022). “Lamda: Language models for dialog applications”. In: *arXiv preprint arXiv:2201.08239*.

Tomasik, Martin J, Laura A Helbling, and Urs Moser (2021). “Educational gains of in-person vs. distance learning in primary and secondary schools: A natural experiment during the COVID-19 pandemic school closures in Switzerland”. In: *International Journal of psychology* 56.4, pp. 566–576.

Vaswani, Ashish et al. (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30.

Vogel, Mabel and Carleton Washburne (1928). “An objective method of determining grade placement of children’s reading material”. In: *The Elementary School Journal* 28.5, pp. 373–381.

Vu, Tu et al. (2018). “Sentence simplification with memory-augmented neural networks”. In: *arXiv preprint arXiv:1804.07445*.

Wang, Alex et al. (Nov. 2018). “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, pp. 353–355. DOI: 10.18653/v1/W18-5446. URL: https://aclanthology.org/W18-5446.

Williams, Ronald J (1992). “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. In: *Reinforcement learning*, pp. 5–32.

Woodsend, Kristian and Mirella Lapata (July 2011). “Learning to Simplify Sentences with Quasi-Synchronous Grammar and Integer Programming”. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, pp. 409–420. URL: https://aclanthology.org/D11-1038.

Wubben, Sander, Antal van den Bosch, and Emiel Krahmer (July 2012). “Sentence Simplification by Monolingual Machine Translation”. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Jeju Island, Korea: Association for Computational Linguistics, pp. 1015–1024. URL: `https://aclanthology.org/P12-1107`.

Xu, Wei, Chris Callison-Burch, and Courtney Napoles (2015). “Problems in Current Text Simplification Research: New Data Can Help”. In: *Transactions of the Association for Computational Linguistics* 3, pp. 283–297. DOI: `10.1162/tacl_a_00139`. URL: `https://aclanthology.org/Q15-1021`.

Xu, Wei, Courtney Napoles, et al. (2016). “Optimizing Statistical Machine Translation for Text Simplification”. In: *Transactions of the Association for Computational Linguistics* 4, pp. 401–415. DOI: `10.1162/tacl_a_00107`. URL: `https://aclanthology.org/Q16-1029`.

Yang, Zhilin et al. (2019). “Xlnet: Generalized autoregressive pretraining for language understanding”. In: *Advances in neural information processing systems* 32.

Zakaluk, Beverly L and S Jay Samuels (1988). *Readability: Its Past, Present, and Future.* ERIC.

Zhang, Muru et al. (2023). “How language model hallucinations can snowball”. In: *arXiv preprint arXiv:2305.13534*.

Zhang, Tianyi et al. (2019). “BERTScore: Evaluating Text Generation with BERT”. In: *CoRR* abs/1904.09675. arXiv: `1904.09675`. URL: `http://arxiv.org/abs/1904.09675`.

Zhang, Xingxing and Mirella Lapata (Sept. 2017a). “Sentence Simplification with Deep Reinforcement Learning”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 584–594. DOI: 10.18653/v1/D17-1062. URL: https://aclanthology.org/D17-1062.

Zhang, Xingxing and Mirella Lapata (2017b). “Sentence simplification with deep reinforcement learning”. In: *arXiv preprint arXiv:1703.10931*.

Zhao, Sanqiang et al. (2018). “Integrating Transformer and Paraphrase Rules for Sentence Simplification”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 3164–3173. DOI: 10.18653/v1/D18-1355. URL: https://aclanthology.org/D18-1355.

Zhu, Zhemin, Delphine Bernhard, and Iryna Gurevych (Aug. 2010a). “A Monolingual Tree-based Translation Model for Sentence Simplification”. In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Beijing, China: Coling 2010 Organizing Committee, pp. 1353–1361. URL: https://aclanthology.org/C10-1152.

– (2010b). “A monolingual tree-based translation model for sentence simplification”. In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pp. 1353–1361.

Chapter A

Appendix

Detailed SARI and BERTScore for system decribed in Chapter 4.

SARI score / Test set / Model		Wiki0	Wiki1	Wiki2	Wiki3	Wiki4	Wiki5	Wiki6	Wiki7	Wiki8	Wiki9	Wiki10	Wiki11	Wiki12	Wiki13	Wiki14	Wiki15	Wiki16	Wiki17	Wiki18	Wiki19	**Overall**
Base	BART-base	39.58	43.13	40.20	40.82	40.99	40.67	39.33	40.14	40.53	40.88	42.18	42.43	43.30	41.17	37.74	41.03	41.99	39.70	39.34	40.48	40.78
Generic	GPT-3	29.54	27.85	29.82	28.29	28.62	29.70	28.41	28.22	29.12	30.22	27.83	30.01	29.76	29.18	28.88	28.62	28.44	30.68	28.42	28.95	29.03
	ChatGpt	32.46	31.05	32.16	30.45	32.07	29.63	30.21	29.93	31.73	31.79	29.21	31.83	31.62	29.85	29.16	32.04	31.37	32.86	31.92	31.07	31.12
Expert	Corresponding **Wiki** model	43.99	44.53	42.56	43.33	43.30	44.88	43.83	45.36	44.33	45.23	45.24	43.22	46.46	44.62	44.25	43.56	44.41	42.74	43.76	46.39	**44.30**
	PubMed0	43.52	43.36	42.62	44.94	44.33	41.31	41.79	42.81	42.02	43.25	41.68	41.51	44.38	43.41	42.90	43.36	43.03	41.57	45.32	44.14	43.06
	PubMed1	41.28	42.73	42.70	41.37	44.00	42.15	43.62	40.72	42.36	43.18	44.35	43.65	43.34	43.23	41.94	42.69	42.15	42.33	41.42	40.74	42.50
	PubMed2	42.72	43.78	43.01	43.16	41.49	44.52	40.69	41.39	42.03	42.19	40.90	43.48	45.08	42.52	41.60	43.95	44.92	40.46	43.88	41.42	42.66
	PubMed3	40.90	43.55	43.09	42.51	44.50	42.13	43.98	41.93	41.92	41.23	42.43	43.70	43.71	42.74	41.82	42.96	45.02	39.59	42.79	43.29	42.69
	PubMed4	40.04	43.34	44.72	43.26	43.14	43.85	43.09	40.27	41.62	44.31	41.43	42.11	42.56	42.43	42.25	41.42	43.33	41.50	41.79	42.54	42.45
	PubMed5	40.53	43.62	44.39	40.24	41.17	43.00	41.95	43.09	42.73	43.81	41.05	44.11	42.80	43.42	42.48	43.08	44.77	41.62	40.68	42.69	42.56
	PubMed6	43.11	42.98	43.84	42.94	40.59	42.03	43.91	40.63	40.95	42.81	44.03	43.80	43.05	42.59	41.02	43.49	44.00	42.00	42.28	42.95	42.65
	PubMed7	41.15	42.38	42.30	43.87	43.19	40.27	42.03	43.76	40.98	43.93	44.08	42.72	44.52	41.97	42.32	42.48	43.38	42.22	45.01	40.97	42.68
	PubMed8	43.59	43.58	41.92	43.81	42.54	42.74	44.77	41.77	42.25	43.69	40.03	43.96	44.83	42.11	41.88	44.40	44.89	41.48	42.48	43.39	43.01
	PubMed9	42.51	41.79	45.22	42.24	41.55	41.35	42.12	42.41	42.72	41.80	43.03	43.69	44.03	43.58	40.82	41.90	42.43	41.87	41.66	42.18	42.45
	PubMed10	42.11	44.12	45.41	42.17	42.80	41.11	42.83	43.29	42.80	41.20	43.50	43.33	42.80	43.29	42.55	42.22	43.43	41.15	42.33	42.58	42.75
	PubMed11	43.47	44.30	44.72	43.85	43.13	42.05	43.93	43.40	40.71	42.67	44.10	43.44	43.38	39.46	42.08	42.87	42.54	42.17	44.31	43.42	43.00
	PubMed12	42.17	44.73	43.65	43.23	42.29	40.96	41.82	41.15	43.31	44.12	42.38	44.00	42.25	40.82	41.67	43.07	43.19	40.36	42.72	40.23	42.41
	PubMed13	40.95	41.53	42.46	41.70	42.44	43.36	41.62	43.52	41.59	43.72	43.18	44.42	45.20	41.20	42.96	42.90	43.84	42.39	42.88	45.04	42.84
	PubMed14	41.49	43.38	44.17	42.49	42.83	42.85	44.60	42.53	41.09	44.83	43.16	42.87	44.32	42.02	43.41	43.98	44.91	43.21	44.16	41.56	43.19
	PubMed15	42.00	43.81	43.83	42.49	41.53	42.69	41.97	41.16	44.18	40.93	41.33	44.95	43.23	42.90	41.28	41.99	44.42	41.84	44.49	42.35	42.67
	PubMed16	43.38	41.63	42.97	42.11	41.39	42.59	42.32	43.22	42.85	45.12	41.27	42.06	44.34	43.17	41.33	42.46	44.40	42.24	44.35	39.71	42.65
	PubMed17	40.74	43.04	41.75	41.40	41.53	44.47	42.83	38.51	43.79	43.70	41.13	43.03	42.70	43.21	43.61	43.85	41.05	41.35	45.01	41.20	42.40
	PubMed18	42.27	43.16	45.59	41.33	44.20	41.83	43.75	42.24	41.86	43.91	41.39	43.87	44.77	44.06	41.01	43.57	43.79	41.16	44.14	43.23	43.06
	PubMed19	43.03	43.61	42.38	40.75	44.17	42.53	43.99	42.73	42.33	43.94	42.24	43.39	43.03	42.67	42.67	43.00	43.83	42.96	45.40	41.66	43.02
	PubMed20	42.67	42.60	42.72	42.02	42.76	42.95	43.42	42.08	43.34	39.79	43.97	42.78	44.28	42.11	41.81	42.49	43.82	42.77	42.26	41.22	42.59
	PubMed21	39.71	42.36	43.71	40.99	41.71	41.45	43.12	42.68	42.69	43.52	44.16	44.25	44.72	42.29	39.49	45.07	42.72	42.25	44.75	40.79	42.62
	PubMed22	42.09	41.96	42.83	41.27	41.09	43.91	42.91	41.61	42.06	42.85	42.64	42.87	44.37	43.12	42.54	43.15	42.30	42.59	43.21	41.46	42.54
	PubMed23	42.96	45.08	44.38	44.48	40.52	40.83	44.83	42.74	43.85	44.42	43.57	42.65	44.62	43.88	41.23	42.52	42.42	42.45	42.81	41.99	43.11
	PubMed24	43.37	43.43	43.09	40.50	40.55	42.57	43.25	41.98	43.70	43.95	43.88	43.19	45.14	42.82	42.61	43.41	44.45	42.12	44.37	41.20	42.98
	PubMed25	42.62	42.82	43.36	41.35	42.47	44.20	44.17	42.58	42.96	44.46	44.03	43.38	42.76	42.41	40.80	41.88	44.23	42.25	44.01	43.94	43.03
	PubMed26	42.76	44.63	42.64	43.09	42.91	42.57	43.57	42.71	41.95	42.63	44.82	42.51	42.08	41.11	40.34	43.53	44.59	41.85	44.50	41.92	42.84
	PubMed27	41.20	45.43	41.81	42.15	41.46	41.58	43.78	39.74	43.47	41.50	42.22	43.24	43.07	42.06	43.63	43.27	43.10	41.66	43.36	41.43	42.46
	PubMed28	39.86	41.25	43.70	44.96	42.41	41.11	42.69	43.74	40.87	42.40	41.69	44.37	44.64	41.88	43.59	42.84	44.57	41.20	44.66	41.58	42.70
	PubMed29	41.65	44.38	42.92	42.73	43.82	41.81	42.48	42.51	42.31	44.10	41.01	44.26	43.33	43.17	42.54	43.37	44.67	40.71	42.29	42.17	42.81
Average of **PubMed**		44.48	43.28	43.40	42.45	42.42	42.36	43.06	42.10	42.38	43.13	42.62	43.39	43.78	42.52	42.01	43.04	43.67	41.78	43.44	42.10	42.75

Table A.1. Detailed SARI score on **Wikipeidax** (We use **Wiki** to refer the **Wikipedia** in the columns)

BERTScore Test set Model		Wiki0	Wiki1	Wiki2	Wiki3	Wiki4	Wiki5	Wiki6	Wiki7	Wiki8	Wiki9	Wiki10	Wiki11	Wiki12	Wiki13	Wiki14	Wiki15	Wiki16	Wiki17	Wiki18	Wiki19	**Overall**
Base	BART-base	0.743	0.797	0.723	0.722	0.755	0.804	0.721	0.766	0.747	0.781	0.737	0.735	0.753	0.698	0.733	0.760	0.736	0.768	0.602	0.733	0.741
Generic	GPT-3	0.524	0.533	0.541	0.532	0.515	0.557	0.545	0.524	0.523	0.508	0.525	0.518	0.514	0.506	0.536	0.546	0.533	0.548	0.535	0.533	0.530
	ChatGpt	0.559	0.568	0.551	0.547	0.549	0.546	0.555	0.549	0.542	0.521	0.521	0.531	0.552	0.495	0.539	0.552	0.534	0.538	0.548	0.538	0.542
Expert	Corresponding **Wiki** model	0.763	0.756	0.805	0.736	0.745	0.772	0.764	0.771	0.738	0.796	0.749	0.748	0.774	0.727	0.770	0.726	0.711	0.755	0.749	0.763	**0.756**
	PubMed0	0.759	0.742	0.786	0.755	0.746	0.735	0.710	0.741	0.774	0.754	0.700	0.792	0.769	0.717	0.748	0.731	0.742	0.755	0.761	0.741	0.748
	PubMed1	0.669	0.771	0.776	0.731	0.746	0.724	0.781	0.736	0.745	0.784	0.794	0.733	0.772	0.713	0.785	0.744	0.764	0.715	0.786	0.708	0.749
	PubMed2	0.740	0.780	0.760	0.706	0.763	0.754	0.687	0.788	0.744	0.765	0.749	0.751	0.745	0.722	0.673	0.773	0.763	0.767	0.783	0.685	0.745
	PubMed3	0.670	0.781	0.750	0.680	0.747	0.758	0.810	0.769	0.753	0.730	0.756	0.769	0.715	0.710	0.646	0.703	0.759	0.673	0.675	0.729	0.729
	PubMed4	0.647	0.797	0.756	0.776	0.766	0.790	0.779	0.788	0.735	0.759	0.752	0.661	0.727	0.736	0.743	0.727	0.742	0.729	0.732	0.725	0.743
	PubMed5	0.649	0.799	0.736	0.676	0.704	0.763	0.763	0.753	0.762	0.756	0.737	0.737	0.720	0.726	0.743	0.660	0.750	0.735	0.745	0.705	0.731
	PubMed6	0.736	0.748	0.740	0.736	0.688	0.756	0.781	0.696	0.679	0.740	0.764	0.738	0.692	0.728	0.685	0.725	0.752	0.759	0.768	0.757	0.733
	PubMed7	0.734	0.769	0.740	0.788	0.751	0.745	0.756	0.764	0.730	0.761	0.786	0.718	0.745	0.720	0.759	0.691	0.759	0.727	0.771	0.705	0.746
	PubMed8	0.758	0.761	0.795	0.745	0.745	0.757	0.775	0.775	0.701	0.770	0.721	0.768	0.769	0.741	0.723	0.731	0.747	0.740	0.748	0.739	0.750
	PubMed9	0.752	0.811	0.758	0.723	0.778	0.719	0.780	0.758	0.766	0.683	0.751	0.742	0.730	0.736	0.707	0.707	0.747	0.754	0.787	0.709	0.745
	PubMed10	0.735	0.770	0.756	0.730	0.741	0.615	0.810	0.751	0.769	0.682	0.763	0.758	0.711	0.715	0.712	0.732	0.769	0.702	0.754	0.726	0.735
	PubMed11	0.738	0.766	0.773	0.786	0.750	0.762	0.782	0.769	0.660	0.714	0.781	0.759	0.767	0.574	0.761	0.729	0.724	0.762	0.763	0.739	0.743
	PubMed12	0.733	0.770	0.758	0.750	0.729	0.719	0.773	0.760	0.753	0.752	0.746	0.758	0.699	0.615	0.688	0.742	0.755	0.696	0.771	0.692	0.733
	PubMed13	0.726	0.673	0.748	0.713	0.732	0.773	0.726	0.759	0.755	0.756	0.770	0.705	0.764	0.685	0.762	0.713	0.746	0.735	0.754	0.739	0.737
	PubMed14	0.735	0.757	0.747	0.707	0.752	0.773	0.798	0.793	0.711	0.771	0.791	0.783	0.774	0.711	0.753	0.738	0.751	0.726	0.760	0.691	0.751
	PubMed15	0.695	0.784	0.765	0.735	0.703	0.746	0.741	0.726	0.771	0.782	0.715	0.743	0.728	0.737	0.762	0.705	0.783	0.761	0.782	0.727	0.745
	PubMed16	0.756	0.723	0.791	0.732	0.683	0.762	0.702	0.763	0.757	0.758	0.700	0.779	0.750	0.746	0.680	0.683	0.786	0.747	0.771	0.686	0.738
	PubMed17	0.654	0.771	0.701	0.640	0.753	0.750	0.785	0.625	0.758	0.774	0.714	0.743	0.740	0.720	0.752	0.728	0.685	0.725	0.749	0.701	0.723
	PubMed18	0.709	0.757	0.763	0.706	0.757	0.794	0.795	0.758	0.684	0.772	0.752	0.745	0.732	0.748	0.734	0.759	0.742	0.742	0.790	0.726	0.748
	PubMed19	0.750	0.776	0.791	0.692	0.762	0.773	0.793	0.748	0.723	0.755	0.777	0.770	0.698	0.728	0.768	0.720	0.727	0.748	0.768	0.688	0.748
	PubMed20	0.745	0.753	0.789	0.733	0.769	0.779	0.784	0.723	0.740	0.693	0.797	0.775	0.752	0.710	0.668	0.724	0.769	0.730	0.767	0.637	0.742
	PubMed21	0.723	0.812	0.743	0.693	0.761	0.707	0.798	0.748	0.724	0.735	0.800	0.748	0.753	0.713	0.701	0.740	0.711	0.729	0.765	0.681	0.739
	PubMed22	0.753	0.780	0.802	0.726	0.678	0.791	0.773	0.671	0.718	0.756	0.791	0.745	0.711	0.714	0.738	0.729	0.734	0.687	0.782	0.725	0.740
	PubMed23	0.734	0.796	0.774	0.770	0.676	0.698	0.780	0.776	0.766	0.735	0.788	0.739	0.757	0.751	0.707	0.743	0.774	0.759	0.768	0.720	0.750
	PubMed24	0.745	0.789	0.718	0.678	0.683	0.719	0.778	0.779	0.734	0.781	0.784	0.726	0.771	0.715	0.734	0.710	0.743	0.732	0.739	0.696	0.738
	PubMed25	0.768	0.780	0.749	0.753	0.740	0.787	0.784	0.766	0.709	0.759	0.806	0.744	0.757	0.756	0.731	0.714	0.772	0.739	0.765	0.749	0.756
	PubMed26	0.770	0.782	0.758	0.716	0.756	0.720	0.769	0.764	0.745	0.775	0.743	0.753	0.734	0.713	0.701	0.740	0.759	0.748	0.769	0.705	0.746
	PubMed27	0.626	0.790	0.697	0.705	0.762	0.723	0.779	0.632	0.770	0.774	0.756	0.749	0.695	0.744	0.744	0.729	0.744	0.753	0.761	0.691	0.731
	PubMed28	0.602	0.810	0.753	0.744	0.724	0.771	0.783	0.746	0.648	0.760	0.772	0.744	0.772	0.711	0.751	0.712	0.756	0.760	0.782	0.680	0.739
	PubMed29	0.630	0.779	0.749	0.734	0.742	0.772	0.762	0.774	0.751	0.783	0.683	0.733	0.715	0.744	0.742	0.727	0.748	0.745	0.646	0.719	0.734
Average of **PubMed**		0.715	0.773	0.757	0.725	0.736	0.748	0.771	0.747	0.735	0.752	0.758	0.747	0.739	0.717	0.727	0.724	0.750	0.736	0.759	0.711	0.741

Table A.2. Detailed BERTScore on **Wikipeida*x*** (We use **Wiki** to refer the **Wikipedia** in the columns)

	SARI score Test set Model	Pub0	Pub1	Pub2	Pub3	Pub4	Pub5	Pub6	Pub7	Pub8	Pub9	Pub10	Pub11	Pub12	Pub13	Pub14	Pub15	Pub16	Pub17	Pub18	Pub19	**Overall**
Base	BART-base	40.10	42.76	37.38	41.81	39.44	41.44	43.60	39.19	40.35	41.15	40.70	41.93	39.67	39.64	36.94	40.57	41.26	41.18	41.03	41.17	40.56
Generic	GPT-3	31.06	31.43	30.82	29.25	32.56	32.19	30.96	29.82	32.36	30.09	29.09	29.18	30.50	31.42	31.50	29.84	28.99	29.23	31.70	32.45	30.72
	ChatGpt	31.75	31.63	30.28	30.52	34.32	29.89	34.59	30.80	31.79	31.28	30.71	32.12	32.57	31.18	33.99	28.75	30.53	30.57	31.65	31.99	31.55
Expert	Corresponding **PubMed** model	46.04	44.02	41.77	45.95	44.83	43.98	43.60	43.61	43.45	44.97	47.79	42.92	48.14	46.05	44.45	45.97	42.66	46.41	48.24	46.11	**45.05**
	Wikipedia0	46.06	43.34	40.63	43.85	43.65	42.51	41.89	41.80	43.76	41.32	43.87	42.84	46.94	44.14	44.05	43.79	42.93	45.37	45.34	43.25	43.57
	Wikipedia1	43.39	42.57	40.29	45.24	44.20	45.11	43.38	41.63	41.96	40.52	45.66	42.03	45.38	44.98	43.79	43.27	44.64	45.83	46.75	42.95	43.68
	Wikipedia2	45.31	43.26	40.92	45.10	42.84	44.82	42.80	40.55	43.14	42.57	44.47	43.55	42.91	43.91	42.59	44.53	42.33	44.46	43.98	43.53	43.38
	Wikipedia3	44.41	42.11	40.26	43.42	43.47	44.51	42.58	42.72	42.73	39.74	43.53	43.33	46.24	42.25	43.68	43.69	41.94	46.19	41.94	44.19	43.15
	Wikipedia4	44.70	40.70	41.02	43.48	42.02	43.26	42.17	41.30	43.34	39.49	43.39	44.76	46.35	42.92	41.51	46.17	40.74	46.15	46.13	42.63	43.11
	Wikipedia5	46.16	41.24	40.83	44.85	41.77	42.62	41.47	41.97	42.22	41.37	44.00	43.46	44.24	41.83	41.15	45.41	42.23	47.27	44.74	44.21	43.15
	Wikipedia6	42.23	44.98	40.99	44.10	41.80	42.62	43.02	41.68	42.84	40.82	45.20	44.38	45.67	41.93	42.06	45.43	43.81	45.50	46.08	44.49	43.48
	Wikipedia7	44.93	44.30	40.96	43.90	44.42	44.90	40.82	43.56	42.10	40.39	42.95	43.19	43.69	43.02	41.53	44.58	42.79	43.82	45.91	43.77	43.28
	Wikipedia8	43.85	42.81	41.95	43.02	42.90	41.77	39.76	42.84	41.78	40.11	45.05	44.83	45.78	43.62	42.97	45.73	43.42	44.07	46.07	44.24	43.33
	Wikipedia9	45.88	42.68	41.28	44.54	45.69	43.88	43.34	42.39	42.65	41.62	44.99	45.20	44.93	42.61	42.95	43.88	42.75	43.72	44.58	42.70	43.61
	Wikipedia10	45.14	42.52	40.63	45.48	43.21	45.29	43.57	43.63	43.68	42.19	44.79	44.51	42.59	41.15	40.73	44.78	45.95	44.84	45.45	43.80	43.70
	Wikipedia11	43.05	44.73	42.57	44.13	42.20	42.16	43.28	42.06	43.18	39.55	45.14	44.82	47.42	41.86	44.28	45.07	42.27	44.12	44.45	44.25	43.53
	Wikipedia12	43.74	42.67	41.42	45.30	41.43	41.50	41.39	41.15	43.90	40.54	42.88	44.52	46.35	41.43	42.16	44.76	44.87	46.92	43.65	43.51	43.20
	Wikipedia13	42.95	42.03	41.47	43.44	45.47	42.90	41.41	41.84	42.90	39.49	40.63	42.90	43.59	42.72	41.57	43.80	43.22	46.46	45.58	43.94	42.92
	Wikipedia14	41.49	44.27	42.48	44.06	42.46	43.58	42.98	43.13	42.39	40.88	44.66	42.15	42.47	42.40	44.49	44.27	42.60	45.02	41.65	44.04	43.07
	Wikipedia15	44.64	44.71	40.12	44.81	43.36	42.47	42.25	43.47	42.77	39.17	42.31	44.99	45.34	40.44	43.39	44.06	42.94	45.17	44.63	43.21	43.21
	Wikipedia16	43.01	45.84	41.05	44.86	42.96	44.49	43.00	41.09	43.01	39.56	44.48	43.54	44.58	44.12	42.18	45.60	39.94	45.22	44.71	43.64	43.34
	Wikipedia17	43.33	42.32	40.62	44.18	44.55	43.13	42.40	41.62	43.21	39.36	44.05	41.84	43.14	43.73	44.08	44.30	40.57	44.06	47.49	43.21	43.06
	Wikipedia18	44.74	44.14	40.04	45.04	42.26	41.52	44.70	42.95	43.10	39.69	44.52	42.64	42.61	41.88	41.25	46.20	45.05	45.41	44.07	41.83	43.18
	Wikipedia19	44.72	44.90	40.79	43.83	43.62	44.99	41.02	42.08	43.39	41.20	41.86	43.90	46.57	43.21	44.47	46.07	43.44	46.22	45.68	43.80	43.79
	Wikipedia20	45.78	45.79	39.14	44.39	41.65	44.54	40.85	43.16	42.07	40.49	43.16	43.40	45.53	42.08	42.91	44.46	42.23	43.54	46.00	44.26	43.27
	Wikipedia21	42.83	45.37	42.21	43.96	41.52	44.55	42.12	42.13	43.92	41.05	43.88	42.76	46.10	42.38	40.89	44.05	42.10	46.18	42.55	43.93	43.22
	Wikipedia22	43.98	43.17	41.61	43.62	43.51	43.34	41.60	42.12	42.86	39.97	44.90	43.50	44.82	44.72	43.07	44.65	42.30	44.55	43.78	42.76	43.24
	Wikipedia23	45.41	42.82	40.86	43.01	42.46	42.87	42.14	41.89	45.22	40.58	41.75	43.27	43.70	42.96	43.21	45.18	45.62	47.94	44.51	42.69	43.40
	Wikipedia24	45.29	43.69	41.22	45.13	45.94	45.19	43.28	42.91	44.35	42.44	43.45	43.45	44.02	42.62	44.39	44.78	43.45	47.87	44.89	43.92	44.11
	Wikipedia25	45.31	44.64	43.61	46.07	40.68	42.70	42.50	42.01	44.01	41.90	46.28	45.02	43.01	41.03	43.60	44.47	43.90	42.92	45.81	43.83	43.66
	Wikipedia26	45.61	42.98	39.72	44.76	43.94	42.53	40.98	42.57	43.08	41.54	42.74	41.86	44.59	41.40	43.18	45.23	44.35	43.14	41.99	42.43	42.93
	Wikipedia27	45.89	44.58	41.34	44.53	44.02	41.21	41.74	41.48	42.74	41.51	45.32	43.03	47.85	43.81	41.63	45.04	43.74	48.39	45.90	43.68	43.87
	Wikipedia28	45.80	42.73	40.32	43.46	44.22	41.83	41.92	42.16	41.86	41.03	43.89	43.94	43.00	44.00	41.16	45.65	40.53	46.38	43.78	42.17	42.99
	Wikipedia29	44.68	46.07	40.86	43.56	43.52	45.32	43.29	42.78	42.91	41.47	44.80	44.57	43.20	43.57	43.92	45.95	42.82	43.40	45.21	45.02	43.85
	Average of **Wikipedia**	44.48	43.60	41.04	44.30	43.19	43.40	42.26	42.22	43.04	40.72	43.95	43.61	44.75	42.76	42.76	44.83	42.98	45.34	44.78	43.53	43.38

Table A.3. Detailed SARI score on ***PubMedx***(We use **Pub** to refer the **PubMed** in the columns)

	BERTScore / Test set / Model	Pub0	Pub1	Pub2	Pub3	Pub4	Pub5	Pub6	Pub7	Pub8	Pub9	Pub10	Pub11	Pub12	Pub13	Pub14	Pub15	Pub16	Pub17	Pub18	Pub19	**Overall**
Base	BART-base	0.739	0.773	0.691	0.725	0.680	0.702	0.735	0.732	0.681	0.756	0.720	0.778	0.646	0.728	0.710	0.741	0.725	0.730	0.731	0.743	0.723
Generic	GPT-3	0.542	0.557	0.549	0.536	0.546	0.565	0.538	0.509	0.550	0.525	0.545	0.549	0.568	0.543	0.519	0.573	0.561	0.572	0.540	0.556	0.547
	ChatGpt	0.517	0.537	0.505	0.511	0.535	0.497	0.527	0.507	0.513	0.509	0.510	0.511	0.508	0.501	0.521	0.512	0.533	0.527	0.491	0.521	0.515
Expert	Corresponding **PubMed** model	0.767	0.774	0.675	0.749	0.725	0.705	0.751	0.748	0.724	0.739	0.758	0.711	0.782	0.768	0.737	0.782	0.701	0.770	0.776	0.688	**0.741**
	Wikipedia0	0.742	0.743	0.723	0.737	0.714	0.658	0.702	0.724	0.709	0.712	0.702	0.723	0.742	0.764	0.686	0.760	0.731	0.722	0.714	0.722	0.721
	Wikipedia1	0.683	0.761	0.713	0.746	0.742	0.755	0.704	0.650	0.720	0.772	0.745	0.681	0.711	0.748	0.670	0.781	0.726	0.750	0.721	0.659	0.722
	Wikipedia2	0.743	0.754	0.711	0.748	0.656	0.756	0.730	0.664	0.758	0.758	0.756	0.754	0.654	0.743	0.668	0.774	0.718	0.748	0.709	0.731	0.727
	Wikipedia3	0.748	0.729	0.720	0.709	0.723	0.750	0.743	0.724	0.712	0.757	0.698	0.710	0.778	0.718	0.647	0.741	0.710	0.745	0.689	0.739	0.725
	Wikipedia4	0.731	0.732	0.688	0.711	0.704	0.716	0.718	0.700	0.740	0.748	0.728	0.773	0.738	0.687	0.689	0.774	0.667	0.757	0.738	0.719	0.723
	Wikipedia5	0.769	0.745	0.704	0.729	0.700	0.696	0.732	0.676	0.732	0.744	0.749	0.771	0.754	0.730	0.684	0.800	0.741	0.762	0.762	0.744	0.736
	Wikipedia6	0.777	0.751	0.725	0.712	0.689	0.674	0.699	0.681	0.747	0.733	0.726	0.741	0.756	0.717	0.707	0.762	0.759	0.761	0.729	0.735	0.729
	Wikipedia7	0.744	0.746	0.650	0.712	0.760	0.761	0.719	0.704	0.697	0.743	0.687	0.707	0.713	0.757	0.672	0.778	0.732	0.758	0.728	0.723	0.725
	Wikipedia8	0.715	0.744	0.730	0.743	0.749	0.701	0.727	0.714	0.695	0.737	0.737	0.765	0.758	0.757	0.698	0.765	0.724	0.731	0.720	0.751	0.733
	Wikipedia9	0.760	0.731	0.688	0.712	0.754	0.731	0.748	0.715	0.746	0.768	0.748	0.794	0.735	0.734	0.680	0.790	0.722	0.626	0.685	0.733	0.730
	Wikipedia10	0.741	0.749	0.699	0.724	0.747	0.754	0.743	0.740	0.707	0.740	0.748	0.760	0.646	0.780	0.629	0.775	0.777	0.770	0.741	0.751	0.736
	Wikipedia11	0.723	0.725	0.734	0.713	0.656	0.724	0.725	0.725	0.655	0.713	0.725	0.755	0.778	0.756	0.691	0.769	0.746	0.759	0.715	0.713	0.725
	Wikipedia12	0.726	0.716	0.692	0.766	0.708	0.715	0.734	0.709	0.723	0.726	0.681	0.792	0.735	0.705	0.723	0.766	0.743	0.750	0.729	0.668	0.725
	Wikipedia13	0.784	0.786	0.700	0.713	0.749	0.711	0.714	0.647	0.755	0.764	0.638	0.771	0.692	0.743	0.664	0.727	0.728	0.754	0.729	0.737	0.725
	Wikipedia14	0.779	0.743	0.747	0.711	0.703	0.693	0.747	0.731	0.686	0.727	0.709	0.711	0.644	0.737	0.668	0.749	0.702	0.743	0.656	0.718	0.715
	Wikipedia15	0.769	0.783	0.684	0.713	0.710	0.699	0.716	0.690	0.723	0.741	0.695	0.803	0.749	0.676	0.705	0.745	0.743	0.765	0.708	0.742	0.728
	Wikipedia16	0.749	0.751	0.695	0.759	0.701	0.726	0.693	0.714	0.737	0.704	0.740	0.727	0.714	0.751	0.662	0.755	0.666	0.764	0.720	0.733	0.723
	Wikipedia17	0.761	0.764	0.700	0.703	0.735	0.711	0.746	0.681	0.737	0.759	0.698	0.685	0.664	0.733	0.665	0.762	0.682	0.734	0.729	0.716	0.718
	Wikipedia18	0.758	0.780	0.686	0.732	0.705	0.681	0.735	0.703	0.744	0.736	0.743	0.707	0.686	0.715	0.678	0.791	0.785	0.736	0.733	0.642	0.724
	Wikipedia19	0.733	0.748	0.681	0.735	0.644	0.750	0.675	0.692	0.727	0.690	0.674	0.704	0.745	0.726	0.684	0.799	0.736	0.747	0.705	0.739	0.717
	Wikipedia20	0.761	0.725	0.676	0.718	0.671	0.743	0.698	0.704	0.722	0.727	0.703	0.748	0.751	0.755	0.698	0.765	0.728	0.751	0.738	0.735	0.726
	Wikipedia21	0.743	0.759	0.735	0.691	0.648	0.731	0.742	0.697	0.774	0.729	0.755	0.697	0.763	0.715	0.700	0.745	0.711	0.765	0.672	0.752	0.726
	Wikipedia22	0.759	0.764	0.692	0.683	0.706	0.707	0.691	0.694	0.681	0.744	0.762	0.726	0.709	0.749	0.681	0.769	0.750	0.734	0.741	0.766	0.725
	Wikipedia23	0.753	0.752	0.701	0.719	0.661	0.733	0.703	0.688	0.763	0.718	0.705	0.749	0.716	0.756	0.641	0.794	0.790	0.743	0.720	0.738	0.727
	Wikipedia24	0.738	0.759	0.723	0.728	0.708	0.756	0.718	0.713	0.734	0.737	0.705	0.664	0.719	0.682	0.657	0.729	0.737	0.746	0.731	0.700	0.719
	Wikipedia25	0.731	0.740	0.743	0.770	0.766	0.688	0.740	0.702	0.737	0.657	0.743	0.737	0.719	0.703	0.730	0.779	0.718	0.716	0.732	0.726	0.729
	Wikipedia26	0.760	0.680	0.691	0.731	0.709	0.680	0.734	0.735	0.766	0.734	0.659	0.724	0.725	0.689	0.671	0.787	0.722	0.736	0.642	0.646	0.711
	Wikipedia27	0.734	0.742	0.694	0.718	0.739	0.728	0.737	0.655	0.718	0.734	0.741	0.685	0.777	0.783	0.685	0.774	0.756	0.769	0.719	0.721	0.730
	Wikipedia28	0.752	0.792	0.696	0.732	0.703	0.746	0.738	0.700	0.729	0.767	0.767	0.759	0.725	0.723	0.685	0.792	0.669	0.759	0.701	0.740	0.734
	Wikipedia29	0.765	0.756	0.678	0.741	0.712	0.768	0.711	0.715	0.722	0.729	0.747	0.757	0.746	0.766	0.723	0.777	0.698	0.750	0.743	0.722	0.736
Average of **Wikipedia**		0.748	0.748	0.703	0.725	0.709	0.721	0.722	0.700	0.727	0.735	0.721	0.736	0.725	0.733	0.681	0.769	0.727	0.745	0.717	0.722	0.726

Table A.4. Detailed BERTScore on ***PubMedx***(We use **Pub** to refer the **PubMed** in the columns)