A Syntactic Rule-based Framework for Parallel Data Synthesis in Japanese GEC

by

Alex Kimn

S.B., Massachusetts Institute of Technology (2019)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2020

Author .....
Department of Electrical Engineering and Computer Science
May 12, 2020

Certified by .....
Takako Aikawa
Senior Lecturer, Global Languages
Thesis Supervisor

Accepted by .....
Katrina LaCurtis
Chair, Master of Engineering Thesis Committee

A Syntactic Rule-based Framework for Parallel Data Synthesis in Japanese GEC

by

Alex Kimn

Submitted to the Department of Electrical Engineering and Computer Science
on May 12, 2020, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

A lack of parallel training data is the greatest barrier to progress in developing models for grammatical error correction (GEC). This data scarcity issue is especially pronounced for Japanese GEC, where very few parallel error-correct corpora are publicly available.

Thus, in this thesis, we propose a syntactic rule-based data synthesis framework designed to generate large parallel training corpora for Japanese GEC. This framework involves two components. The first is a set of syntactic rules that each characterize a common Japanese grammatical error and a corresponding correction by using syntactic information extracted from a small seed corpus. The second is a process for systematically generating error-correct sentence pairs by applying these syntactic rules to any arbitrary corpora of correct Japanese text.

To test this framework, we train a standard neural machine translation (NMT) system for Japanese GEC on a training dataset that combines the publicly available Lang-8 corpus and our synthesized data. We evaluate this trained model on a novel parallel training corpus designed to mimic Japanese learners' writing in the classroom setting and compare it's performance to a similar model trained on the Lang-8 corpus alone. Using this comparison, we show that our synthesised data can significantly improve the performance of Japanese GEC systems.

Thesis Supervisor: Takako Aikawa
Title: Senior Lecturer, Global Languages

Acknowledgments

Here, I would like to express my gratitude to those who have helped have made this thesis possible.

First, I would like to thank my advisor Takako Aikawa for inviting me to work on this project, and for guiding and mentoring me throughout the last few years. Thank you for challenging me with this problem, exposing me to some fascinating aspects of machine learning and natural language processing, and for motivating me to complete this project every step of the way.

Moreover, I would also like to thank Dr. Tetsuro Takahashi of Fujitsu Laboratories, Kawasaki, Japan and Yiqun Hu for lending me their knowledge and expertise. Your thoughtful feedback and advice was critical in shaping this thesis, and this project would not have been remotely as successful were it not for your collaboration.

Additionally I would like to thank the Japan Foundation, Los Angeles as well as MIT's Computer Science and Artificial Intelligence Laboratory for their generous financial support. In particular, I would like to thank Professor Dorothy Curtis for helping us acquire the administrative and hardware resources necessary to complete this project.

MIT would have been a lot less fun and far more difficult were it not for my friends. I would like to thank all of you for providing me with some great times and unforgettable memories.

And last, but of course not least, I would like to thank my parents. Without their never-ending support and encouragement, I would never had the chance to come to MIT, let alone complete this thesis. I will always be grateful for your love, which has pushed me forward every day.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the era of the internet and increasing globalization, the number of foreign language learners around the world has increased dramatically. As a result, there is an ever-growing demand for a system that can provide quick and reliable corrective feedback to foreign language students as they learn to write. In response, much recent research has been conducted in grammatical error correction (GEC), the natural language processing (NLP) task that addresses this need.

GEC is defined as the problem of automatically detecting and correcting grammatical errors within written text. In most recent GEC systems, this is typically interpreted as a translation task, in which the source language consists of potentially erroneous learners' writing, and the target language consists of corrections to that writing. The translation is then accomplished by employing machine translation (MT) models adapted from the NLP task of the same name.

Currently, the state-of-the-art in GEC [33, 2] consists of systems containing neural machine translation (NMT) models. These NMT-based approaches use large, complex neural networks to solve the GEC problem end-to-end. However, due to their large parameter count, the performance of NMT models is often restricted by a lack of available training data.

As a result, several methods that aim to generate artificial errors and increase the amount of usable training data have recently been proposed. These include several model-based approaches that involve the training of new "noising" models to pur-

posefully introduce grammatical errors into correct sentences [32, 25]. Unsupervised approaches that generate errors in a context-independent manner have also been considered [10]. Such methods have shown great promise in improving model performance and robustness. Indeed, in the recent 2019 Building Educational Applications (BEA) shared task [1], many of the top-ranking GEC systems incorporated some form of artificial error generation (AEG) to expand or augment existing training corpora.

1.1 Proposed Approach

Therefore, in this thesis, we propose a robust syntactic rule-based framework for synthesizing parallel training data for Japanese GEC. The proposed data synthesis framework consists of two primary components: (i) a set of user-constructed **syntactic rules**, and (ii) an error generation process dictated by those syntactic rules.

1.1.1 Syntactic Rules

Each syntactic rule is a representation of a particular Japanese grammatical error and a corresponding correction. Which error each rule represents can be inferred from a single pair of error and correct **representative phrases** extracted from a seed corpus. These representative phrases serve as an example error/correction pair for that grammatical error. For instance, the rule with the pair of representative phrases[1]:

Error: 新しいの本 (“new NO book”)

Correct: 新しい本 (“new book”)

represents a commonly encountered particle insertion error.

[1]The inserted particle の of the error phrase is a possession marker. In this paper, we translate の and all other Japanese particles to their capitalized rōmaji equivalents (e.g. の → “NO”)

1.1.2 Error Generation

New error-correct sentence pairs can then be generated using each rule's **error mapping**. This error mapping is comprised of a set of relations between the tokens of the two representative phrases of the rule, and provides a template for introducing errors into correct Japanese phrases, that are syntactically similar to the rule's representative correct phrase. These similar phrases are considered to "match" to the rule.

The error generation process therefore consists of two steps. In the first step, correct Japanese phrases that match to a rule are extracted from an arbitrary corpus of correct Japanese text. Then, the relations of the error mapping are applied to the selected correct phrases to generate corresponding error phrases, thus creating error-correct sentence pairs.

We validate the efficacy of the proposed data synthesis framework by training three models that use a standard NMT architecture. Each model is trained on one of three training datasets: a set containing only data synthesized by our framework, a set derived from the publicly available Lang-8 corpus [18], and a set created by combining both synthesized and Lang-8 data. We evaluate the trained models on a novel evaluation set that was created specifically to mimic Japanese language learners' writing in the classroom setting, and compare the performance of each model.

Our results show that our data synthesis framework can be used to increase the performance of Japanese GEC systems through the augmentation of existing training corpora with our synthesized data. Additionally, we demonstrate that our framework can be used to train models that target specific grammatical errors by showing that the models trained using synthetic data successfully correct most examples of the error types that were represented within that synthetic data. Finally, we show that our data synthesis framework can be used to train reasonable Japanese GEC systems even in the absence of publicly available training corpora.

1.2 Thesis Outline

The remainder of this thesis is structured as follows. Chapter 2 presents additional background information on GEC and AEG. Chapter 3 outlines the primary motivations for our framework and introduces the Japanese morphological analyzer MeCab that enables it to function. Chapter 4 introduces the various language resources used in the development and testing of this work. Chapters 5 and 6 describe in detail the proposed rule-based data synthesis framework and its individual components. Chapter 7 contains an overview of our experiments. Chapter 8 provides a discussion of the results of these experiments. Finally, Chapter 9 contains some concluding analysis, identifies the current limitations and challenges of our approach, and proposes avenues for future research.

Chapter 2

Related Work

In this chapter, we overview prior work done in the natural language processing (NLP) task of grammatical error correction (GEC). We begin in Section 2.1 with an overview of GEC as a whole, and a discussion of the current state-of-the-art. Then, in Section 2.2, we discuss the problem of data scarcity in depth and consider recent proposed approaches for artificial error generation (AEG). In Section 2.3, we discuss the various metrics commonly used to gauge the performance of GEC systems. Finally, in Section 2.4, we briefly discuss Japanese GEC specifically and the additional challenges that it brings when compared to English GEC.

2.1 Grammatical Error Correction

Grammatical error correction, or GEC, is the NLP task that addresses the problem of detecting and correcting grammatical errors within written text. In an ideal GEC system, a potentially erroneous sentence is given as an input, and a related grammatically correct sentence is expected as the output.

The earliest attempts at GEC systems were rule-based approaches consisting of a set of hand-crafted rules. Initially, these rules were primarily informed by simple pattern analysis and string replacement [16]; however, eventually these rules evolved to incorporate more complex features like syntactic analysis [19]. Nevertheless, despite being relatively easy to implement, rule-based GEC systems are limited in perfor-

mance, as it is difficult to define enough rules to account for the entire set of possible grammatical errors.

As such, more complex, data-driven machine learning approaches for GEC were developed when larger, annotated corpora specific to the task became available. The first major class of these were classifier-based approaches capable of distinguishing and correcting various error types. In these systems, GEC was formulated as a multi-class classification problem, in which each class represented a different type of grammatical error. For example, [11] and [30] each developed maximum-entropy classifiers capable of detecting English article and English preposition errors, respectively. Unfortunately, these early classifier-based systems were often unable to account for multiple co-dependent errors, as each classifier typically targeted corrections for a single word within an error type. Thus, solutions involving cascading multiple classifiers together or hybridizing the system with rule-based models [26] were eventually developed.

More recently, GEC systems inspired by existing machine translation (MT) models have risen to dominate the state-of-the-art. In these MT-based systems, GEC is treated as a translation problem between a source language of potentially erroneous text and a target language of correct text. The translation is then completed using an adapted MT model. Compared to the complex model pipelines of later classifier-based GEC systems, this formulation of GEC has the advantage of allowing a single model to perform the end-to-end task.

The 2014 Conference on Natural Language Learning (CoNLL) shared task [21], which aimed to standardize the evaluation of GEC and establish the state-of-the-art, was the impetus for the rise of MT-based GEC. In this competition, two of the top three highest-ranked systems [7, 13] incorporated statistical machine translation (SMT) models to a significant extent. Since then, additional SMT-based [14, 3] and later neural machine translation (NMT)-based systems [15, 4, 8] have continued to establish the state-of-the-art. Most recently, in the 2019 Building Educational Applications (BEA) shared task [1], NMT systems [33, 2] using the Transformer architecture of neural network [31] dominated the performance rankings. This, combined with the absence of any rule-based or classifier-based submissions to the 2019 BEA task

show that, at least for the time being, NMT-based systems are the unquestionable state-of-the-art in GEC.

Thus, in this work, we choose to evaluate our data synthesis framework using NMT models. Specifically, we use a convolutional sequence-to-sequence (seq2seq) architecture implemented with the open-source library *fairseq* [23] as the basis for our trained models. This type of NMT model has previously been shown to be competitive with the state-of-the-art in English GEC [4]. Additionally, while English and Japanese are quite different languages, MT models used in English GEC have shown to be relatively effective when adapted for Japanese GEC [34]. As such, we consider the selected seq2seq model to be a reasonable candidate to represent potential Japanese GEC systems that may be trained using our synthesized data.

2.2 Artificial Error Generation

One drawback of the use of NMT models in GEC is that their performance is heavily dependent on the amount of available training data, which exists in the form of parallel corpora of pairs of error and correct sentences. Typical NMT models contain millions of parameters; thus, they often fail to generalize adequately unless the training corpora is quite large. While this is not a serious problem for most traditional MT problems, where large parallel corpora of millions of sentence pairs are readily available, the same cannot be said of GEC. Even for English, the least resource-scarce GEC language, only a few parallel corpora of significant size, such as the NUS Corpus of Learner English (NUCLE) [6] and the Lang-8 corpus [18], are publicly available. Thus, data scarcity is one of the the primary roadblocks in improving the performance of current GEC systems.

To address this data scarcity issue, much recent work in GEC has considered the process of artificial error generation (AEG). This process seeks to increase the amount of available training data by synthesizing artificial error-correct sentence pairs. Several of these methods utilize probabilistic algorithms to create new data for specific errors. For example, [28] and [27] entertain the idea of using confusion distributions

from English as a Second Language (ESL) corpora to generate article and preposition errors that mimic real-world learner mistakes.

Approaches inspired by the MT idea of back-translation [29] have also been proposed. These involve the training of essentially reversed GEC systems, in which "noising" MT models that translate correct sentences into error sentences are employed as a means of generating new grammatical errors. For example, [25] proposes training a phrase-based SMT error generation model by reversing the source and target corpora of an existing GEC problem. Similarly, [32] trains an encoder/noising NMT model in the same fashion. This noising model is then used to augment sentence pairs while a corresponding decoder/denoising model, which is used to solve the original GEC problem, is trained.

Additionally, some more recent approaches make use of existing linguistic resources. Specifically, [10] proposes using confusion sets from a spell-checker to perform token substitution and introduce synthetic errors into sentences in a unsupervised and context-independent fashion. [17] proposes two error generation processes: one based on round-trip translations of correct text through bridge languages and another using Wikipedia edit histories.

Each of the aforementioned AEG methods has shown promise in improving model performance and robustness when used to expand or augment existing parallel training corpora. Hence, it is clear that generating synthetic training data is an effective method at tackling the data scarcity issue in GEC.

However, the proposed AEG approaches are not universally applicable. In particular, the aforementioned model-based and probabilistic AEG methods require the prior existence of significant amounts of parallel error-correct data. This is problematic for most GEC languages, including Japanese, where few, if any, parallel corpora of significant size exist. Moreover, most of the above AEG approaches offer little control over what types of error-correct sentence pairs can be generated. This makes it difficult to train or update GEC models to correct for specific grammatical errors. Though not necessary in academic GEC systems, such a feature would be quite useful in keeping publicly deployed GEC systems current and up-to-date.

Our framework is thus designed specifically to address the aforementioned use cases for which current AEG methods are not well suited. Specifically, the proposed approach is able to function when the amount of available parallel training data is extremely small, as each rule of the framework can be informed by just a single example pair of sentences. Thus, given only a small parallel seed corpus and any arbitrary corpus of correct Japanese text, our framework can be used to generate large quantities of new error-correct sentence pairs. Moreover, the rule-based nature of our framework allows specific grammatical errors to be targeted during training. This can be done by synthesizing data using rules crafted specifically to represent those errors.

2.3 Metrics

GEC systems are primarily evaluated by comparing their outputs to reference corrections made by human experts. There are several metrics that have been proposed to evaluate GEC systems; we cover three of the commonly used metrics: (i) the MaxMatch scorer, (ii) the BLEU score, and (iii) the the GLEU score, in the following sections.

2.3.1 MaxMatch

The current standard metric used in evaluating GEC models is the MaxMatch scorer [5]. This metric was used to rank the 2013 and 2014 CoNLL shared tasks [22, 21], and thus has been used by many recent works to compare model performance and establish the state-of-the-art.

The MaxMatch scorer works by calculating the F-score, the harmonic mean of precision and recall, in a manner that takes into account phrase-level edits. Specifically, given a set of gold-standard hand-annotated edits that detail the location and type of each error, the scorer computes the set of phrase-level edits made by the model that overlaps the most with the gold-standard. The scorer than computes and outputs the F-score made on those edits.

2.3.2 BLEU

The Bilingual Evaluation Understudy (BLEU) score [24] is one of the most popular and widely-used metrics for evaluating MT systems. As many GEC systems draw inspiration from MT, the BLEU score has thus also been used in evaluating GEC. The score is determined by computing a modified form of precision on n-grams of a model's output against multiple human references.

However, since the input and output languages of GEC are identical, and since most grammatical errors are fairly localized, the BLEU score of GEC models is typically misleadingly high. Nevertheless, the BLEU score is still used, as unlike the MaxMatch scorer, it does not require a set of detailed annotations.

2.3.3 GLEU

The Generalized Language Evaluation Understanding (GLEU) score [20] is derived from the BLEU score, and is designed specifically to improve upon previous GEC metrics. Unlike the BLEU score, the GLEU score takes into account the source sentence, placing greater weight on n-grams that exist within the references but not within the source. This modification has been shown to improve correlation of scores with respect to human rankings of GEC system outputs.

Like the BLEU score, the GLEU score does not require any detailed annotations. Thus, given the lack of the gold-standard edits as required by the MaxMatch scorer within Japanese GEC corpora, we use the GLEU score as our primary metric in this work.

2.4 Japanese GEC

The vast majority of existing GEC research has been conducted for English specifically. Indeed, almost all of the aforementioned research above has concerned English GEC. Thus, unlike English GEC, which has had the 2013 and 2014 CoNLL [22, 21] shared tasks as well as the 2019 BEA [1] shared task to establish baseline models and

shared evaluation criterion, there exists no formal standard for training or evaluating Japanese GEC systems.

This lack of standard evaluation corpora motivates our use of the novel Teacher corpus introduced in Section 4.2 as our primary evaluation set. Though it is not representative of learner sentences as a whole, the Teacher corpus is designed to reflect learner writing in the classroom setting, which we believe is a reasonable target for a real-world Japanese GEC system.

Moreover, to our best knowledge, the Lang-8 multilingual corpus [18] is the only publicly available corpus containing a significant number of parallel error-correct Japanese sentence pairs. Thus, the data scarcity issue is even more pronounced for Japanese GEC than it is for English GEC. As a result, the potential benefits of a successful approach for AEG in Japanese particularly high.

This is especially true given the lack of precedent for AEG in Japanese GEC. We know of only one previous attempt; [12] proposes adapting the probabilistic method given in [28] for English prepositions to instead generate errors involving Japanese particles. Thus, to our best knowledge the proposed approach is the first attempt at AEG for general Japanese grammatical errors.

Chapter 3

Motivation

In this chapter we overview the primary motivations for the proposed data synthesis framework. We begin in Section 3.1 with the main intuition behind our use of syntactic rules as the basis of our framework. Then, in Section 3.2, we introduce MeCab, the Japanese morphological analyzer that enables our entire framework to work.

3.1 Syntactic Rules

The original idea for our framework is largely inspired by the original rule-based systems of English GEC. As in English, many simple grammatical errors in Japanese can be explained by using only syntactic information. For example, the correct conjugation form of adjectives (e.g. 速い, "fast") usually depends only on the part-of-speech of the following token. For example, if an adjective is followed by a noun, (e.g. 車, "car") the correct conjugation of the adjective is the root form. However, if the subsequent token is a verb, the correct conjugation is the adverbial (-く) form. In this way, given an arbitrary phrase containing an adjective followed by some other token, we can determine whether or not the phrase is grammatically correct just from the conjugation of the adjective and the part-of-speech of the other token.

This implies that it is possible to construct "syntactic rules" to represent these grammatical errors using only syntactic information. Specifically, given an example error phrase **e** and a corresponding correction **c**, we can construct a mapping (i.e. a

rule) from the former to the latter such that any arbitrary phrase $\mathbf{e}'$ containing the same grammatical error as $\mathbf{e}$ can be corrected in a similar fashion. For instance, any error phrase containing an adjective in the adverbial form followed by a noun can be corrected by converting the adjective to its root form (e.g. 速く車, "quickly car", can be corrected to 速い車, "fast car"). We call the pair of example phrases ($\mathbf{e}$ and $\mathbf{c}$) the **representative phrases** of the syntactic rule; these can be thought of as one realization of the error-correction pairs that the rule represents.

However, the key intuition is that we can also "reverse" these syntactic rules. More specifically, a reverse mapping can be created in the opposite direction as described above, from $\mathbf{c}$ to $\mathbf{e}$. Such a mapping is essentially an error generation rule, having the effect of converting any phrase syntactically similar to $\mathbf{c}$ into a new error phrase containing the same grammatical error as $\mathbf{e}$. We again consider the same adjective and noun example. Using the reversed mapping, we can convert any arbitrary phrase containing an adjective in the root form followed by a noun to a related error phrase containing the adjective in the adverbial form. For example, 速い車 ("fast car") can be converted to 速く車 ("quickly car"); similarly, 優しい先生 ("kind teacher") can be changed to 優しく先生 ("kindly teacher"), and so on.

Moreover, corpora containing correct sentences, unlike parallel error-correct corpora, can readily be obtained for most languages, including Japanese. Thus, each of the above syntactic rules can easily be used to generate large amounts of new parallel error-correct sentence pairs, exactly what is necessary for training GEC systems. This can be done by first using the syntactic information of $\mathbf{c}$ to construct a set of constraints that define what a syntactically similar phrase to $\mathbf{c}$ must contain. Then, the correct corpus can be systematically scanned for any phrases $\mathbf{c}'$ that match those constraints. Finally, the reversed mapping can then be used to convert each matched phrase $\mathbf{c}'$ into a corresponding error.

In this way, a single syntactic rule informed by a lone pair of representative phrases can be used to generate many thousands of new parallel sentence pairs. This is the core idea of our approach.

3.2 MeCab

The proposed framework is largely enabled by the morphological analyzer MeCab[1], which serves as our sentence tokenizer and as the source of the syntactic information used in the construction of the framework's syntactic rules. MeCab is commonly used for sentence segmentation and tagging in Japanese-related NLP problems. In this work, we use MeCab with the custom, open-source, *mecab-ipadic-NEologd* morpheme dictionary[2], which contains many new words and named entities that help improve the tokenization output of MeCab.

3.2.1 Output Structure

Given an input sentence, MeCab outputs a list of tokens that comprise the sentence. For each token, Mecab also outputs a set of nine syntactic tags[3].

An example of MeCab's output is shown below in Figure 3-1. The input sentence 今日は寒いです ("Today is cold.") is shown on the second line. Each subsequent line except the last EOS line contains a token (e.g. 今日, "today") followed by a tab character and a comma-delimited list of syntactic tags.

```
~ $mecab
今日は寒いです。
今日    名詞,副詞可能,*,*,*,*,今日,キョウ,キョー
は      助詞,係助詞,*,*,*,*,は,ハ,ワ
寒い    形容詞,自立,*,*,形容詞・アウオ段,基本形,寒い,サムイ,サムイ
です    助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
。      記号,句点,*,*,*,*,。,。,。
EOS
```

Figure 3-1: Sample output produced by MeCab on the command line.

The syntactic tags outputted by MeCab are used in our data synthesis framework to distinguish phrases that are "matched" by a particular syntactic rule from those that are not[4]. An overview of the features that each syntactic tag corresponds to is given

[1] https://taku910.github.io/mecab/

[2] https://github.com/neologd/mecab-ipadic-neologd

[3] Additional (Japanese) documentation on the output formatting of MeCab is given at http://taku910.github.io/mecab/#parse

[4] See Section 5.2.

in Table 3.1. The features (and their corresponding English translations) represented within Mecab's output are shown in the column labeled as "**Feature**", while the outputted syntactic tags and their corresponding English translations[5] are given in the columns labeled "**Mecab Output**" and "**English Translation**, respectively. Note that the character "**＊**" is outputted when no syntactic tag is given for a particular feature.

Feature	**MeCab Output**	**English Translation**
品詞 (part-of-speech)	形容詞	adjective
品詞細分類1 (part-of-speech subcategory 1)	自立	independent
品詞細分類2 (part-of-speech subcategory 2)	＊	N/A
品詞細分類3 (part-of-speech subcategory 3)	＊	N/A
活用型 (inflection form)	形容詞・アオウ段	adjective (aou-columns)
活用形 (conjugated form)	基本形	basic form
原型 (lemma)	寒い	cold
読み (reading)	サムイ	SAMUI
発音 (pronunciation)	サムイ	SAMUI

Table 3.1: Analysis of the syntactic tags outputted by MeCab for the token 寒い ("cold") when the string 今日は寒いです。 ("Today is cold.") is inputted.

In this work, we only use syntactic tags that correspond to the following features: 品詞 ("part-of-speech"), 品詞細分類1 ("part-of-speech subcategory 1"), 活用型 ("inflection form"), 活用形 ("conjugated form"), and 原形 ("lemma") to inform our syntactic rules. We select these features as they are most relevant to the grammatical characteristics of a token. For example, substituting tokens with different part-of-speech tags (e.g. replacing a noun with an adjective) will almost always result in a grammatically incorrect sentence. Similarly, substituting two tokens with different conjugations (e.g. a root form adjective with an adverbial form adjective) will also cause grammatical errors.

On the other hand, the features 品詞細分類2 ("part-of-speech subcategory 2")

[5]We translate the values given for the reading and pronunciation rows as their capital rōmaji equivalents.

and 品詞細分類3 (“part-of-speech subcategory 3”) are not used as tags for them are rarely outputted. Moreover, distinctions between two tokens that with differing tags for either of these features are usually grammatically irrelevant. The features 読み (“reading”) and 発音 (“pronunciation”) are similarly ignored as the values of tags corresponding to them do not have any relation to the grammatical characteristics of a token.

Chapter 4

Data Resources

In this chapter, we overview the Japanese corpora used in the development and evaluation of our data synthesis framework. In Section 4.1, we introduce the Yourei corpus, which is used as the source of correct Japanese text for our error generation process. In Section 4.2, we present the teacher corpus, a novel parallel corpus that serves as our primary evaluation set. Finally, in Section 4.3, we overview the Lang-8 corpus, a publicly available parallel corpus that serves as the training set of our baseline model.

4.1 Yourei Corpus

The Yourei corpus is the primary source of the correct Japanese sentences we use to test the error generation component of our data synthesis framework. This corpus consists of over 12 million sentences scraped from the keyword search website *yourei.jp*[1], which hosts a large database of modern Japanese text[2]. *yourei.jp* functions as follows: when a user inputs a keyword into the search bar, the website searches its database and returns a set of example sentences containing the queried keyword or some conjugated form of that keyword.

We construct the Yourei corpus by first aggregating $7,489$ vocabulary terms from the Japanese-Language Proficiency Test (JLPT)[3] vocabulary lists to use as input

[1] http://yourei.jp/

[2] In general, this text is sourced from written work such as novels and newspaper articles.

[3] https://www.jlpt.jp/e/

keywords. For each of these keywords we then extract the first 2000 example sentences returned by the website for each word. We then apply several pre-processing steps to reduce data noise. First, all duplicate sentences are removed, along with sentences containing any characters that are neither Japanese nor alphanumeric. Additionally, any sentence containing more than 50 or fewer than 10 tokens when parsed by MeCab are also excluded. The final corpus size after these pre-processing steps are applied is $6,623,362$ unique sentences.

We select *yourei.jp* as our source of correct text as its database contains text that is relatively similar to what might be found in learner's writing. Specifically, its database contains many sentences that are written in the informal polite form[4] of Japanese. This is the form that is typically taught first to beginning Japanese language learners, and is stylistically more similar to their writing than the formal impolite form[5] commonly seen in other online or published sources of Japanese text (e.g. Wikipedia). Moreover, the *yourei.jp* database also contains a large number of examples of conversational Japanese. Though not necessarily stylistically close to learner's writing, these sentences consist mostly of simple vocabulary that learners also commonly use. Due to these similarities, we hypothesize that models trained on data generated using the *yourei.jp* data will generalize relatively well to learner's writing.

The aforementioned similarities can be seen in the sample data from the Yourei corpus shown below in Table 4.1. In the table, Sentence 1 is written in the in the informal polite form while Sentences 2 and 3 are excerpts from conversations. Note that while the sentence structure of the latter two is significantly more complex than would be expected in language learners' writing, the vocabulary present is quite simple.

[4]This form is commonly characterized with sentences ending in ～です and verbs being conjugated in the ～ます form.

[5]This form is characterized by the use of verbs conjugated in the short form and phrases like である.

#	**Sentence**	**English Translation**
1	でもエリーザはじぶんになんの罪もないことを知っていました。	But, Eliza new that she had done nothing wrong.
2	あなたはそれに少しでも打たれなかったか。	Did you even get hit by that at all?
3	でもあなた、きっと日本なんかいやだって外国にでも行っちまうんでしょう。	But you'll probably be sick of Japan and leave for abroad.
4	こんなくらいなら定子の所にでもいるほうがよほどましだった。	It would have been better for me to stay at Sadako's place if this was the case
5	一度でもこのくらい憎むべき言葉が、人間の口を出た事があろうか？	Have such detestable words ever been spoken by a human?

Table 4.1: Sample data from the Yourei corpus file containing the website's responses to the query keyword でも ("but; even")

4.2 Teacher Corpus

The teacher corpus[6] serves as the seed corpus for the syntactic rules of our data synthesis framework. It consists of $4,375$ error sentences sourced from Japanese language teachers. The teachers also provided each error sentence with one or more reference corrections, resulting in the corpus having a total of $6,345$ unique error-correct sentence pairs.

These error-correct sentence pairs were created to mimic mistakes commonly made by Japanese language learners in the classroom, under the condition that each sentence pair contain a single grammatical error[7]. Due to this constraint, most of the errors present in the corpus are simple and localized. However, as a whole, they cover the entire range of mistakes commonly seen in Japanese learners' writing. This includes not only spelling and syntactic errors, but also more complex errors, such as stylistic or semantic errors.

Thus, we also use the teacher corpus as our primary evaluation set. As mentioned in Section 2.4, to our best knowledge, no standard method of evaluation exists for

[6]This corpus is publicly available on Github at https://github.com/ahkimn/japanese_gec_corpus

[7]Note that this constraint was somewhat vague; some error sentences thus contain multiple, co-dependent errors.

Japanese GEC. Since the teacher corpus is designed to resemble students' writing in the classroom, a potential real-world application for Japanese GEC, we believe that it is a reasonable candidate to serve as such a standard, and thus use it to evaluate our trained models.

Sample data from the teacher corpus are shown below in Table 4.2. Note that for the purposes of evaluation, the boundaries of the error phrase within each error sentence is demarcated by angled brackets, "<and ">". The corresponding correction in the correct sentence is demarcated by parentheses, "(" and ")". Also, note that not all of the error sentences are unique; Sentences 3 and 4 show one error sentence that is given multiple possible corrections.

#	**Error Sentence**	**Correction**	**English Translation**
1	音楽を<聞きた>。	音楽を(聞いた)。	I listened to music
2	昨日、夕食にピザを<食びた>。	昨日、夕食にピザを(食べた)。	I ate pizza for dinner yesterday.
3	<昨日雪が降る>。	(昨日雪が降った)。	It snowed yesterday.
4	<昨日雪が降る>。	(昨日雪が降りました)。	It snowed yesterday.
5	学校は８時に<始めます>。	学校は８時に(始まります)。	School begins at 8 o'clock.

Table 4.2: Sample data from the teacher corpus.

4.3 Lang-8 Corpus

The publicly available NAIST Lang-8 Learner Corpora [18] serves as the training corpus of the baseline model we use in evaluating our data synthesis framework. The original, multilingual corpus consists of 580,549 conversations from the language exchange SNS site Lang-8, of which 185,991 are labeled as having a learner of Japanese.

We apply several pre-processing steps to retrieve the Japanese text in the original corpus. First, we select the conversations labeled as having a learner of Japanese and extract each error/correction sentence pair in each conversation[8]. Then, we use

[8]The exact format of the Lang-8 data is given at https://sites.google.com/site/naistlang8corpora/home/readme-raw.

the open source library *langid*[9] to select for only those pairs in which both the error and correction are classified as being written in Japanese. Next, we remove sentence pairs in which the Levenshtein distance between the two sentences is greater than the length (in characters) of the error sentence. Finally, we remove any duplicate pairs (e.g. where both the error and correct sentences of each pair are equivalent), and any sentences with a high ratio of alphanumeric to Japanese text. This results in a final corpus containing $1,788,054$ unique error-correct pairs.

Sample paired data from the Lang-8 corpus is given below in 4.3. This data is representative of the fact that the Lang-8 corpus is quite unstructured, containing many incomplete/short sentences. This is expected of informal conversations, but is not necessarily representative of learners' writing in the classroom setting. Moreover, note that the corpus also contains sentence pairs in which no corrections are made. Sentences 4 and 5 are examples of this; in both examples the error sentence is grammatically correct and no correction is made.

#	**Error Sentence**	**Correction**	**English Translation**
1	続いて探します！	引き続き探します！	I will continue to search (for it)!
2	大学の来年は「上級へ行こう」のコースを勉強しなければなりません。	来年は大学で「上級へ行こう」のコースを勉強しなければなりません。	Next year, I must take the "Let's advance" course at the university.
3	だれでもはわたしにおしえていただけませんか。	だれかわたしに教えていただけませんか。	Can anyone tell me?
4	でも、これは一時的だと思っている。	でも、これは一時的だと思っている。	But, I think this is temporary.
5	今日のイベント	今日のイベント	Today's event

Table 4.3: Sample data from the Lang-8 corpus.

[9]https://github.com/saffsd/langid.py

Chapter 5

Syntactic Rules

In this chapter, we overview the structure and construction of the first primary component of our framework: **syntactic rules**. Each syntactic rule is intended to represent a common Japanese grammatical error and a corresponding correction. These rules are informed by example error-correct phrase pairs from a seed corpus, and dictate the second component of our framework, the data synthesis process described in Chapter 6.

Each syntactic rule consists of three main components: (i) a pair of error and correct **representative phrases** used to provide the syntactic information for (ii) a matrix of **requisite syntactic tags**, which defines the set of phrases that the rule matches to using this information. Finally, the rule also contains (iii) a set of relations, the **error mapping**, that defines how the tokens of the representative error phrase are derived from those of the representative correct phrase. This error mapping is then used to direct the error generation process described in Section 6.2.

We explain the derivation of each of these components in Sections 5.1, 5.2, and 5.3, respectively. Additionally, in Section 5.4, we present a natural extension of the syntactic rule, called a **character syntactic rule**. This additional type of rule allows grammatical errors involving character-level edit operations to also be represented in our framework.

5.1 Representative Phrases

The representative phrases of a syntactic rule are a pair of two tokenized phrases: a **representative error phrase** and a **representative correct phrase**. The former is representative of a common Japanese grammatical error, while the latter is representative of a possible correction to that error. Together, the two phrases act as a template for correcting erroneous phrases that contain the grammatical error represented by the representative error phrase. We call this grammatical error the **rule error** of the syntactic rule.

Each pair of representative phrases is derived from an example error-correct phrase pair from a parallel seed corpus. Let s_e be the string value of the example error phrase and let s_c be the string value of the example correct phrase. Then, we represent the representative error and correct phrases of the rule as the vectors $\mathbf{e}$ and $\mathbf{c}$, respectively, which are formulated as follows:

$$\mathbf{e} = [e_0, e_1, \cdots, e_{l'-1}] \quad \mathbf{c} = [c_0, c_1, \cdots, c_{l-1}]$$

Here, e_j is the jth token outputted by MeCab when s_e is parsed, and c_i is the ith token outputted by MeCab when s_c is parsed; l' and l represent the number of tokens in s_e and s_c, respectively.

An example pair of representative phrases is shown in the colored boxes of the left-hand side of Figure 5-1[1]. Using the above notation, these are as follows[2]:

$$\mathbf{e} = [\text{楽しい, な, ゲーム}] \quad \mathbf{c} = [\text{楽しい, ゲーム}]$$

Equivalently, in English[3]:

$$\mathbf{e} = [\text{fun, NA, game}] \quad \mathbf{c} = [\text{fun, game}]$$

[1]Note that English translations are given in parentheses beneath each Japanese word. This format is used to present English translations in all similar figures in this work.

[2]The token な is a copula particle used after a type of noun known as 形容動詞 (literally "adjective verb", also known as a ～な adjective) to allow them to function as adjectives.

[3]As の is a particle we translate it to its capital rōmaji equivalent, "NA".

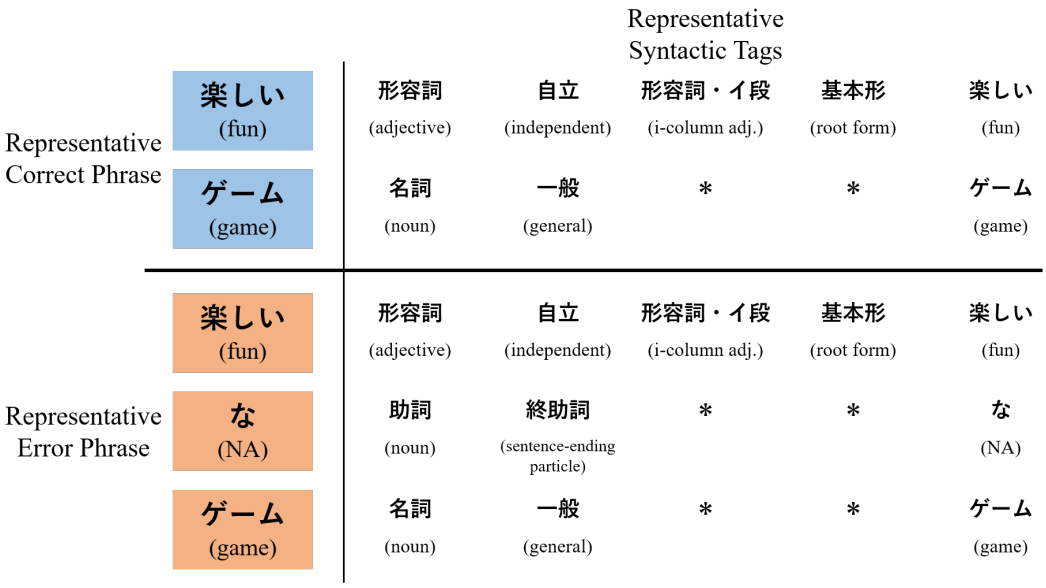

Figure 5-1: Representative phrases and syntactic tags of an example rule that represents a common particle insertion error.

5.1.1 Representative Syntactic Tags

Each representative phrase is accompanied by a set of corresponding **representative syntactic tags**, which are derived from the MeCab output produced by the tokens of the phrase.

For the representative correct phrase, $\mathbf{c}$, this set of tags is formulated as the matrix $\mathbf{C}_{tag}$. This is constructed jointly with $\mathbf{c}$ as follows: for each token c_i of the representative correct phrase, let

$$\mathbf{t}_i = [t_{i,0}, t_{i,1}, \cdots, t_{i,4}]$$

be a vector of length five consisting of the corresponding syntactic tags outputted by MeCab. Specifically, let $t_{i,0}$, $t_{i,1}$, $t_{i,2}$, $t_{i,3}$, and $t_{i,4}$ be the tags outputted for the following features:

- $t_{i,0} \rightarrow$ 品詞 ("part-of-speech")
- $t_{i,1} \rightarrow$ 品詞細分類1 ("part-of-speech sub-category 1")

- $t_{i,2} \rightarrow$ 活用型 (“inflection form”)
- $t_{i,3} \rightarrow$ 活用形 (“base form”)
- $t_{i,4} \rightarrow$ 原形 (“lemma”)

As mentioned in Section 3.2, the syntactic tags of these features are used as they are the most relevant to the grammatical characteristics of each token.

In Figure 5-1, we can derive $\mathbf{t}_0$ and $\mathbf{t}_1$ from the syntactic tags shown to the right of c_0 =楽しい (“fun”) and c_1 =ゲーム (“game”) as follows:

$$\mathbf{t}_0 = [\text{形容詞, 自立, 形容詞・イ段, 基本形,楽しい}]$$
$$\mathbf{t}_1 = [\text{名詞, 一般, ＊, ＊, ゲーム}],$$

Equivalently, in English[4]:

$$\mathbf{t}_0 = [\text{adjective, independent, adjective (i-column), root form, fun game}]$$
$$\mathbf{t}_1 = [\text{noun, general, *, *, game}],$$

These $\mathbf{t}_i$ are then concatenated to form $\mathbf{C}_{tag}$ as:

$$\mathbf{C}_{tag} = [\mathbf{t}_0, \mathbf{t}_1, \cdots \mathbf{t}_{l-1}]^T$$

where l is as before the length in tokens of $\mathbf{c}$.

Thus, for the example of Figure 5-1, we have that:

$$\mathbf{C}_{tag} = \begin{bmatrix} \text{形容詞} & \text{自立} & \text{形容詞・イ段} & \text{基本形} & \text{楽しい} \\ \text{名詞} & \text{一般} & ＊ & ＊ & \text{ゲーム} \end{bmatrix}$$

The construction of the matrix $\mathbf{E}_{tag}$ of the representative syntactic tags corresponding to the representative error phrase is nearly identical, but uses the syntactic tags

[4]Note, the character representing no output, ＊, is replaced with the asterisk, as ＊ is a non-English character.

outputted for the tokens e_j instead. Therefore, for the same example, we have:

$$\mathbf{E}_{tag} = \begin{bmatrix} \text{形容詞} & \text{自立} & \text{形容詞・イ段} & \text{基本形} & \text{楽しい} \\ \text{助詞} & \text{終助詞} & * & * & \text{な} \\ \text{名詞} & \text{一般} & * & * & \text{ゲーム} \end{bmatrix}$$

In English, the example $\mathbf{C}_{tag}$ and $\mathbf{E}_{tag}$ are:

$$\mathbf{C}_{tag} = \begin{bmatrix} \text{adjective} & \text{independent} & \text{adjective (i-column)} & \text{root form} & \text{fun} \\ \text{noun} & \text{general} & * & * & \text{game} \end{bmatrix}$$

$$\mathbf{E}_{tag} = \begin{bmatrix} \text{adjective} & \text{independent} & \text{adjective (i-column)} & \text{root form} & \text{fun} \\ \text{particle} & \text{sentence-ending particle} & * & * & \text{NA} \\ \text{noun} & \text{general} & * & * & \text{game} \end{bmatrix}$$

The representative syntactic tag matrices are used to inform other components of the syntactic rule. Specifically, $\mathbf{C}_{tag}$ is used as the starting point of the requisite syntactic tag matrix described in Section 5.2. The differences between $\mathbf{C}_{tag}$ and $\mathbf{E}_{tag}$ are used to derive the correction/error mapping introduced in Section 5.3.

5.1.2 Phrase Selection

As previously noted, the representative phrases are derived from example error/correct phrases in a parallel seed corpus. This derivation begins by selecting a sentence pair at random from the seed corpus, and determining the string difference (if one exists) between the error and correct sentences. The set of tokens containing this string difference serves as the starting point of the representative phrases.

For example, consider the following sentence pair that contains the representative

phrases shown in Figure 5-1:

Error: これ | は | 楽しい | な | ゲーム | です | 。 (This is a fun NA game.)

Correct: これ | は | 楽しい | ゲーム | です | 。 (This is a fun game.)

Here, the token boundaries have been demarcated by the character "|". In this case, the string difference is simply the inserted character 'な' of the error sentence. Since only one token (the な) contains any part of this string difference, representative phrases could be constructed as follows:

$$\mathbf{e} = [\text{な}] \quad \mathbf{c} = [\ \ ]$$

However, the tokens of the string difference by themselves are usually not enough to allow a specific rule error to be resolved. As is described below in Section 5.2, the rule error is largely specified by constraints placed on individual syntactic tags of $\mathbf{C}_{tag}$. Thus, for the above pair of $\mathbf{e}$ and $\mathbf{c}$, the rule error can only be ambiguous, as an empty $\mathbf{c}$ results in the matrix $\mathbf{C}_{tag}$ being empty; such a $\mathbf{C}_{tag}$ cannot provide any constraints on the context in which the insertion of な occurs. In other words, the rule error would represent the insertion of な between any two arbitrary tokens. While the arbitrary insertion of な into correct Japanese will almost always produce an error, an ideal rule error should be representative of a specific type of learner's mistake.

Therefore, to obtain the final representative phrases of a syntactic rule, the initial string difference is incrementally expanded with adjacent tokens. For example, for the initial $\mathbf{e}$ and $\mathbf{c}$ above, one token on each side of the string difference, namely 楽しい ("fun") and ゲーム ("game"), is added. In this way, we obtain the representative phrases of Figure 5-1:

$$\mathbf{e} = [\text{楽しい}, \text{な}, \text{ゲーム}] \quad \mathbf{c} = [\text{楽しい}, \text{ゲーム}]$$

These additional tokens provide the information necessary to define the rule error

as the common learner mistake of inserting な between an adjective (e.g. 楽しい) and the noun it modifies (e.g. ゲーム).

In this work, the phrase expansion process is done by hand, as given a pair of parallel phrases, it is difficult to define formal criterion for what an "optimal" pair of representative phrases should be. The pair of informal guidelines we use is as follows:

1. The representative phrases must include sufficient context to define the rule error as a specific Japanese learner mistake. Specifically, it should contain enough tokens so that $\mathbf{C}_{tag}$, and the requisite syntactic tags of Section 5.2 constructed from it, can be used to constrain the rule to a specific grammatical error.

2. However, the representative phrases should be of minimum length to ensure that the rule matches[5] to as many sentences containing the rule error as possible.

5.2 Requisite Syntactic Tags

The applicability of a syntactic rule to an arbitrary correct phrase is determined by comparing the syntactic information of the phrase to that of the representative correct phrase of the rule. To accomplish this, each rule contains a matrix of requisite syntactic tags, $\mathbf{R}_{tag}$. This matrix defines what specific syntactic tags each token of a phrase matched to the rule must contain, and is primarily used to filter potential matches during the data synthesis process described in Chapter 6.

$\mathbf{R}_{tag}$ is constructed by applying a binary mask, $\mathbf{B}_{tag}$, element-wise to the values of $\mathbf{C}_{tag}$. If $\mathbf{B}_{tag}[i,k] = 1$, then $\mathbf{R}_{tag}[i,k] = \mathbf{C}_{tag}[i,k]$, and the syntactic tag $\mathbf{R}_{tag}[i,k]$ is **requisite** for a correct phrase $\mathbf{c}'$ to match to the rule. In other words, the kth tag of the ith token of $\mathbf{c}'$ must be the same as that of the representative correct phrase. On the other hand, if $\mathbf{B}_{tag}[i,k] = 1$, then $\mathbf{R}_{tag}[i,k] = 0$, and no restriction is placed on the value of the kth tag of the ith token of $\mathbf{c}'$.

[5]Note that the matching process is also controlled by the requisite syntactic tags and is described in the following section.

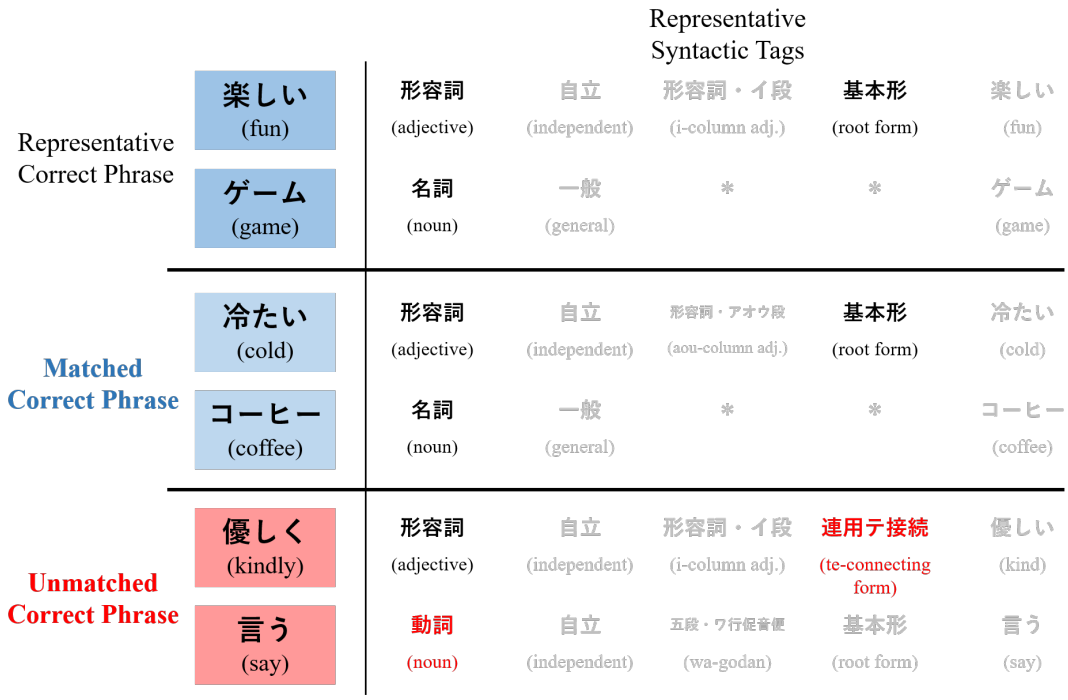

Figure 5-2: Phrase matching for the example syntactic rule representing the insertion of the token な between an adjective and the noun it modifies.

For example, consider the requisite syntactic tags for the rule shown in Figure 5-2[6], where we have:

$$\mathbf{C}_{tag} = \begin{bmatrix} \text{形容詞} & \text{自立} & \text{形容詞・イ段} & \text{基本形} & \text{楽しい} \\ \text{名詞} & \text{一般} & * & * & \text{ゲーム} \end{bmatrix}$$

or equivalently:

$$\mathbf{C}_{tag} = \begin{bmatrix} \text{adjective} & \text{independent} & \text{adjective (i-column)} & \text{root form} & \text{fun} \\ \text{noun} & \text{general} & * & * & \text{game} \end{bmatrix}$$

Noting the requisite and non-requisite (faded-out) syntactic tags, we obtain the binary

[6]This rule is identical to the one represented in Figure 5-1.

mask $\mathbf{B}_{tag}$ as:

$$\mathbf{B}_{tag} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Applying this to $\mathbf{C}_{tag}$, we obtain:

$$\mathbf{R}_{tag} = \begin{bmatrix} \text{形容詞} & 0 & 0 & \text{基本形} & 0 \\ \text{名詞} & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \text{adjective} & 0 & 0 & \text{root form} & 0 \\ \text{noun} & 0 & 0 & 0 & 0 \end{bmatrix}$$

Examining the vectors $\mathbf{r}_i = \mathbf{R}_{tag}[i, :]$ corresponding to the ith token of a matched phrase, we see that:

$$\mathbf{r}_0 = [\text{形容詞}, 0, 0, \text{基本形}, 0] = [\text{adjective}, 0, 0, \text{root form}, 0]$$

This imposes the constraint that the first token of a phrase that matches to the rule must be an 形容詞 (adjective) in the 基本形 (root form). Similarly, we have:

$$\mathbf{r}_1 = [\text{名詞}, 0, 0, 0, 0] = [\text{noun}, 0, 0, 0, 0]$$

which specifies that the second token of a match must be a 名詞 (noun).

In other words, given an arbitrary phrase $\mathbf{c}'$ with corresponding syntactic tags $\mathbf{C}'_{tag}$[7], we have that $\mathbf{c}'$ matches to the rule if $\mathbf{C}'_{tag} \circ \mathbf{B}_{tag} = \mathbf{R}_{tag}$[8].

This is shown in Figure 5-2 with the two candidate phrases shown below the representative correct phrase. The phrase 冷たいコーヒー ("cold coffee"), shown in light-blue, matches to the rule as both tokens adhere to the above constraints. Specifically:

$$\mathbf{C}'_{tag} \circ \mathbf{B}_{tag} = \begin{bmatrix} \text{形容詞} & 0 & 0 & \text{基本形} & 0 \\ \text{名詞} & 0 & 0 & 0 & 0 \end{bmatrix} = \mathbf{R}_{tag}$$

[7] $\mathbf{C}'_{tag}$ is constructed from $\mathbf{c}'$ in the same manner that $\mathbf{C}_{tag}$ and $\mathbf{E}_{tag}$ are constructed from $\mathbf{c}$ and $\mathbf{e}$ respectively.

[8] Note that $\circ$ denotes the element-wise or Hadamard product between the two matrices.

On the other hand, the phrase 優しく言う (“kindly say”), shown in light orange, does not match to the rule, as the first token is not an adjective in the root form, and because the second token is a verb and not a noun. This is shown as follows:

$$\mathbf{C'}_{tag} \circ \mathbf{B}_{tag} = \begin{bmatrix} \text{形容詞} & 0 & 0 & \text{連用テ接続} & 0 \\ \text{動詞} & 0 & 0 & 0 & 0 \end{bmatrix} \neq \mathbf{R}_{tag}$$

In this work, the values of the matrix $\mathbf{B}_{tag}$ are determined manually, as formulating an algorithm to optimize for the best combination of representative tags is quite difficult. In general, as more syntactic tags are more ones are used in $\mathbf{B}_{tag}$, the more constraints are placed on each token of a matched phrase $\mathbf{c'}$, and the syntactic rule becomes more restrictive for matching phrases. Conversely, if more zeros are used in $\mathbf{B}_{tag}$, the rule becomes less specific. Therefore, much like the selection of representative phrases, the “optimal” selection of requisite syntactic tags is defined by a set of two competing criterion:

1. The requisite syntactic tags should be restrictive enough so that the set error phrases $\mathbf{e'}$ that can be generated correspond to a specific Japanese learner mistake.

2. However, the requisite syntactic tags should be permissive enough for the rule to apply to as many sentences containing the rule error as possible.

5.3 Error Mapping

The final component of each syntactic rule is the error mapping, a set of relations that maps the tokens of the representative correct phrase, $\mathbf{c}$, to those of the representative error phrase, $\mathbf{e}$. These relations show how the rule error can be introduced into an arbitrary phrase similar to $\mathbf{c}$. Hence, the primary role of the error mapping is to guide the generation of artificial errors during the data synthesis process described in Chapter 6.

The relations of the error mapping are each categorized as being one of five types: **insertion**, **deletion**, **preservation**, **re-conjugation**, or **substitution**. Each relation is represented as an equation $e_j = \text{OPERATION}(x)$. This shows that given an arbitrary correct phrase $\mathbf{c}'$ matched by the rule, the jth token of a corresponding error phrase $\mathbf{e}'$ containing the rule error can be derived from x, where x is either a constant string or c'_i, the ith token of $\mathbf{c}'$[9]. We define each relation type below:

5.3.1 Insertion

The insertion relation is represented as $e_j = \text{INSERT}(s)$, and shows that that regardless of the values of $\mathbf{c}'$, the jth token of $\mathbf{e}'$ is the string s. An error mapping of a rule containing an insertion is shown in Figure 5-3[10]. The position of the inserted token is shown by the half-shaded boxes in each of the error phrases.

In this example, we have that $e_1 = \text{INSERT}(\text{な})$. This implies that the second token of any generated error phrase must be equivalent to な. This is shown by the generated error 厳しいな先生 ("strict NA teacher"), where な is inserted between the two tokens of the matched correct phrase 厳しい先生 ("strict teacher").

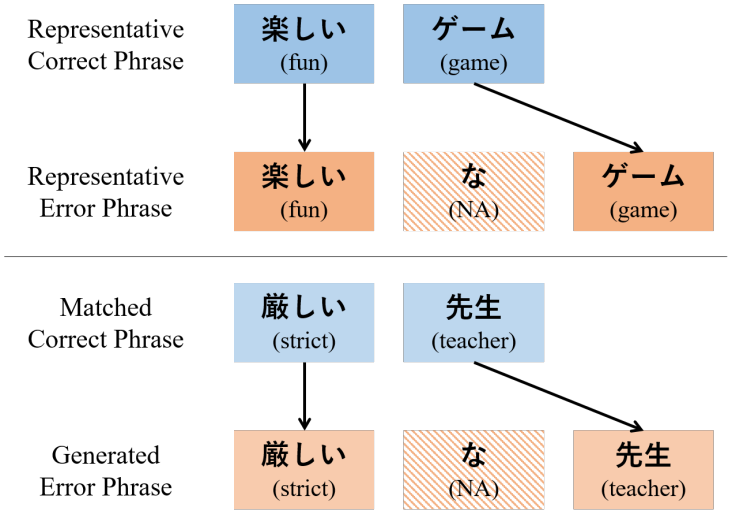

Figure 5-3: Mapping for an example syntactic rule representing the insertion of the token な between an adjective and the noun it modifies.

[9]Note that $\mathbf{c}'$ and $\mathbf{e}'$ are not complete sentences; rather, they are phrases equivalent in length (in tokens) to $\mathbf{c}$ and $\mathbf{e}$, respectively.

[10]This rule is the same as the one shown in Figures 5-1 and 5-2.

5.3.2 Deletion

The deletion relation is the inverse of the insertion relation, and is represented as $* = \text{DELETE}(c_i)$. This relation indicates that the ith token of $\mathbf{c}'$ is ignored (deleted) and does not appear in $\mathbf{e}'$. An example of an error mapping containing a deletion relation is shown in Figure 5-4, with the position of the deleted token denoted by the half-shaded boxes in each of the correct phrases. Here, we have $* = \text{DELETE}(c_1)$, indicating the second token of any matched correct phrase (e.g. な in the example) is ignored when creating the corresponding error phrase.

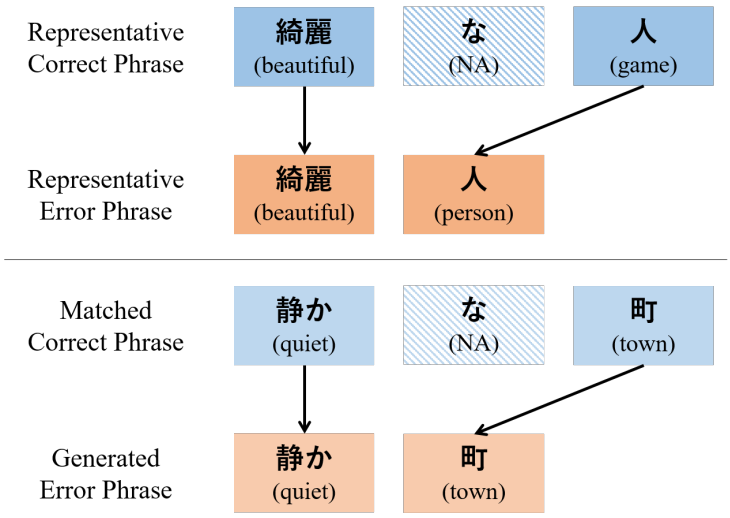

Figure 5-4: Mapping for an example syntactic rule representing the deletion of the token な between a -na adjectives and the noun it modifies.

5.3.3 Preservation

The preservation relation is equivalent to a copy operation, and is represented as $e_j = \text{PRESERVE}(c_i)$. This shows that the jth token of $\mathbf{e}'$ is simply equivalent to the ith token of $\mathbf{c}'$.

Examples of preservation relations are shown in both Figures 5-3 and 5-4 as solid arrows from the correct to error phrases. In Figure 5-3 specifically, we have $e_0 = \text{PRESERVE}(c_0)$ and $e_2 = \text{PRESERVE}(c_1)$, which shows that the first and third tokens of any generated error phrase $\mathbf{e}'$ for that rule are equivalent to the first and

second tokens of the matched correct phrase $\mathbf{c}'$ from which $\mathbf{e}'$ is generated.

5.3.4 Re-conjugation

The re-conjugation relation is represented as $e_j = \text{RECONJUGATE}(c_i)$ and is used when the token e_j of the representative error phrase shares the same lemma as the token c_i in the representative correct phrase, but has a different conjugated form.

During error generation, this means that the jth token of $\mathbf{e}'$ can be derived from the ith token of $\mathbf{c}'$ by searching a dictionary for a token that (i) has the same lemma as c_i and (ii) has the same conjugated form as e_j.

The re-conjugation relation primarily used to represent conjugation errors of verbs and adjectives, as for these errors, the value of the mis-conjugated token in a generated error phrase $\mathbf{e}'$ is dependent on the corresponding token in $\mathbf{c}'$. The re-conjugation relation is necessary to represent this this type of error, as the alternative, a pair of deletion and insertion relations, will always result in the same token being added to all generated errors,

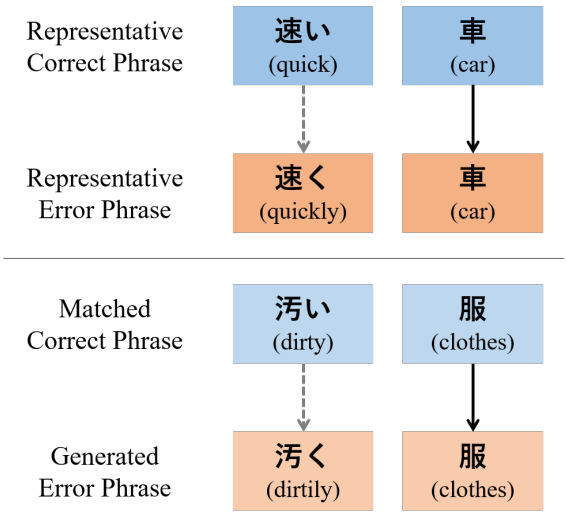

Figure 5-5: Mapping for an example syntactic rule representing the incorrect conjugation of an adjective into the adverbial form when the adjective is followed by a noun.

An example error mapping for a syntactic rule representing a conjugation error is shown in Figure 5-5. The conjugation relation is shown by the gray dashed arrow

from the first token of each correct phrase to the first token of each error phrase.

In this example, we have the re-conjugation relation $e_0 = \text{RECONJUGATE}(c_0)$, which indicates that the first token of any $\mathbf{e}'$ is a mis-conjugation of the first token of $\mathbf{c}'$. In the representative phrases, e_0 is an adverbial form adjective while c_0 is a root form adjective. Thus, for the example generated error, we that the adverbial form 汚く ('dirtily") is derived from 汚い ('dirty"), the root form adjective present in the matched correct phrase.

5.3.5 Substitution

The substitution relation is represented as $e_j = \text{SUBSTITUTE}(c_i)$ and complements the re-conjugation relation. Specifically, it is used when the token e_j of the representative error phrase shares the same conjugated form as the token c_i of the representative correct phrase, but has a different lemma.

The derivation for generated tokens for this relation is therefore similar to that of the re-conjugation relation. Specifically, the jth token of $\mathbf{e}'$ is derived from the ith token of $\mathbf{c}'$ by searching a dictionary for token that (i) has the same lemma as e_j and (ii) has the same conjugated form as c_i.

The substitution relation is used to represent errors in which a verb or adjective is incorrectly replaced by a different verb or adjective with the same conjugation. Again, this is used in place of an insertion/deletion pair as it impossible for a single insertion relation to output different conjugations of the same lemma.

One common grammatical error that can be represented with the substitution relation is the confusion of the verb ある ("to be; to exist"), used for inanimate subjects, with the similar verb いる ("to be; to exist"), which is used for animate subjects. The error mapping of a rule representing this error is shown below in Figure 5-6. In the figure, the substitution relation is indicated by the gray dotted arrow.

In the example, we have that $e_0 = \text{SUBSTITUTE}(c_0)$, indicating that the first token of $\mathbf{e}'$ is the form of ある with the same conjugation as the form of いる in the first token of $\mathbf{c}'$. Note that the rule not only encompasses the root form conjugation,

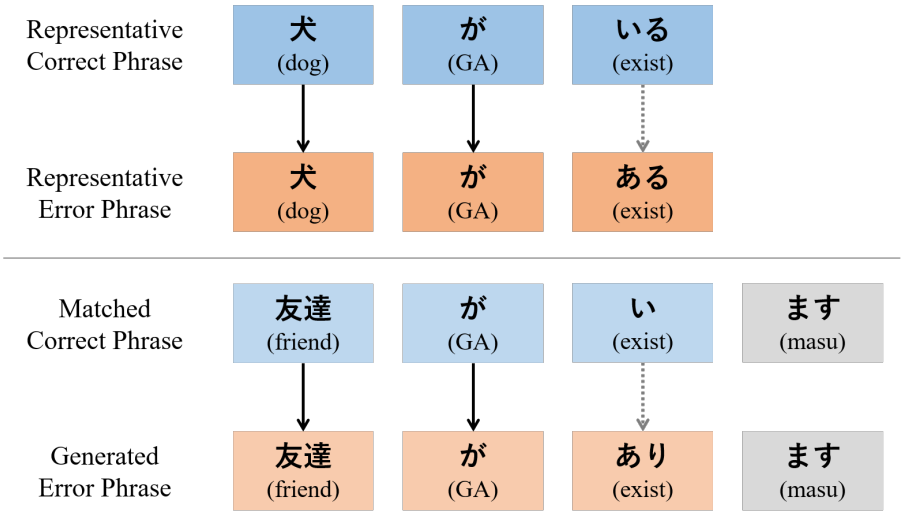

Figure 5-6: Mapping for an example syntactic rule representing the substitution of two "to be" verbs, いる and ある. The tokens written in gray boxes are not part of either the matched correct or generated error phrases; they are shown to provide context for the conjugation of the substituted verbs.

as in the representative phrases, but also the -ます form, as shown by the generated error phrase, and all other verb conjugations.

The error mapping of a syntactic rule can be derived algorithmically from the rule's representative phrases by comparing the the lemmas of each token. This process is explained below.

Let $\epsilon_j = \mathbf{E}_{tag}[j, 4]$ be the lemma of e_j and let $\gamma_i = \mathbf{C}_{tag}[i, 4]$ be the lemma of c_i. Then, for each ϵ_j the set of γ_i is searched to see if any two are equivalent (e.g. $\epsilon_j = \gamma_i$), and relations are added according according to the following logic:

1. If an equivalent lemma exists (e.g. $\exists \epsilon_j = \gamma_i$), and the representative syntactic tags corresponding to e_j and c_i are equivalent, the relation $e_j = \text{PRESERVE}(c_i)$ is added to the error mapping.

2. If an equivalent lemma exists, but the syntactic tags of e_j and c_i differ only in the conjugated form, the relation $e_j = \text{RECONJUGATE}(c_i)$ is added.

3. If no equivalent lemma exists (e.g. $\nexists \epsilon_j = \gamma_i$), but the conjugated forms of e_j

and c_i are equal, the relation $e_j = \text{SUBSTITUTE}(c_i)$ is added.

4. If none of the above conditions are satisfied, e_j is treated as an insertion and add the relation $e_j = \text{INSERT}(e_j)$ instead.

Finally, once each of the e_j are given a relation, for any tokens c_i for which no relations $e_j = \text{OPERATION}(c_i)$ exist, the relation $* = \text{DELETION}(c_i)$ is added.

5.4 Character Syntactic Rules

While the above construction of syntactic rules is sufficient to represent most Japanese grammatical errors, there are several types of common learners' mistakes that cannot be represented. Chief among these are mistakes that modify the spelling of a token, resulting in either (i) an original and modified token that are syntactically unrelated, or (ii) parsing errors within MeCab (e.g. due to the creation of out-of-dictionary tokens).

An example of this is the spelling error in which a small っ is dropped in the middle of a noun[11]. A potential pair of representative phrases for a syntactic rule with this mistake as its rule error is:

$$\mathbf{e} = [\text{いっしょ}] \quad \mathbf{c} = [\text{いしょ}]$$

Here the noun いっしょ ("together") is misspelled as the unrelated いしょ ("will; farewell note"). As the two tokens have different lemmas and no conjugation (as they are nouns), the error mapping as described in Section 5.3 cannot sufficiently model the mistake; two unrelated tokens can only be mapped with an insertion/deletion relation pair.

However, this and many other similar mistakes can instead be modeled as a set of character-level edit operations. Therefore, we propose a natural extension of the syntactic rule, the **character syntactic rule**, that allows for the representation of

[11]The character っ is used to indicate a doubled consonant sound and its accidental deletion is a common mistake among Japanese language learners.

these additional errors. The overall structure of character syntactic rules mirrors that of syntactic rules, retaining the same construction of representative phrases and requisite syntactic tags. However, an additional matrix of **requisite characters** is added on as a further constraint on matched phrases. Furthermore, a new set of character-level relations is defined to create a **character error mapping**, which replaces the error mapping defined in Section 5.3. We overview these two changes below in Sections 5.4.1 and 5.4.2, respectively.

5.4.1 Requisite Characters

The matrix of requisite characters, $\mathbf{R}_{char}$, functions in a similar fashion to the matrix of requisite syntactic tags, $\mathbf{R}_{tag}$. It determines what set of specific characters any correct phrase that matches to a character syntactic rule must have.

To construct $\mathbf{R}_{char}$, we first determine $\mathbf{C}_{char}$, the matrix of characters of $\mathbf{c}$, the representative correct phrase. Let $c_{i,k}$ be the kth character of c_i. Then, we have:

$$\mathbf{C}_{char} = \begin{bmatrix} c_{0,0} & c_{0,1} & \cdots & c_{0,n-1} \\ c_{1,0} & c_{1,1} & \cdots & c_{1,n-1} \\ \cdots & \cdots & \cdots & \cdots \\ c_{l-1,0} & c_{l-1,1} & \cdots & c_{l-1,n-1} \end{bmatrix}$$

where l is the length in tokens of $\mathbf{c}$ and n is the maximum length (in characters) of any of the c_i. For any token c_i with length $m < n$, the entries $c_{i,k}$ for $k > m$ are zero-padded.

$\mathbf{R}_{char}$ is constructed by applying a binary mask, $\mathbf{B}_{char}$, element-wise to the values of $\mathbf{C}_{char}$ as follows:

$$\mathbf{B}_{char} \circ \mathbf{C}_{char} = \mathbf{R}_{char}$$

If $\mathbf{B}_{char}[i,k] = 0$, then $\mathbf{R}_{char}[i,k] = 0$. This means that no constraints are enforced by $c_{i,k}$ on an arbitrary matched phrase $\mathbf{c}'$. Alternatively, if $\mathbf{B}_{char}[i,k] = 1$, then we have that $\mathbf{R}_{char}[i,k] = \mathbf{C}_{char}[i,k] = c_{i,k}$ is **requisite**. In this case, $\mathbf{c}'$ must have $c'_{i,k'} = c_{i,k}$, where $c'_{i,k'}$ is the k'th character of the ith token of $\mathbf{c}'$. Note that k' is not necessarily

equivalent to k; the non-zero values of $\mathbf{R}_{char}[i,:]$ constrain $\mathbf{c}'$ so that the pattern of requisite characters $\mathbf{R}_{char}[i,:] \neq 0$ exists somewhere within $\mathbf{c}'$, but not at any specific index. More formally, the constraint enforced by $\mathbf{R}_{char}$ on $\mathbf{c}'$ is that the following equation holds for each $0 < i < l$:

$$\exists h \quad \text{s.t.} \quad \forall k \in \mathbf{R}_{char}[i,k] \neq 0, c'_{i,k+h} = c'_{i,k'} = c_{i,k}$$

Here h is considered the **match offset** of the matched correct phrase $\mathbf{c}'$ relative to the rule. This offset is used to guide the generation of the corresponding error phrase $\mathbf{e}'$ under the character error mapping defined in Section 5.4.2.

Consider now the aforementioned rule with representative phrases as follows:

$$\mathbf{e} = [\text{いっしょ}] \quad \mathbf{c} = [\text{いしょ}]$$

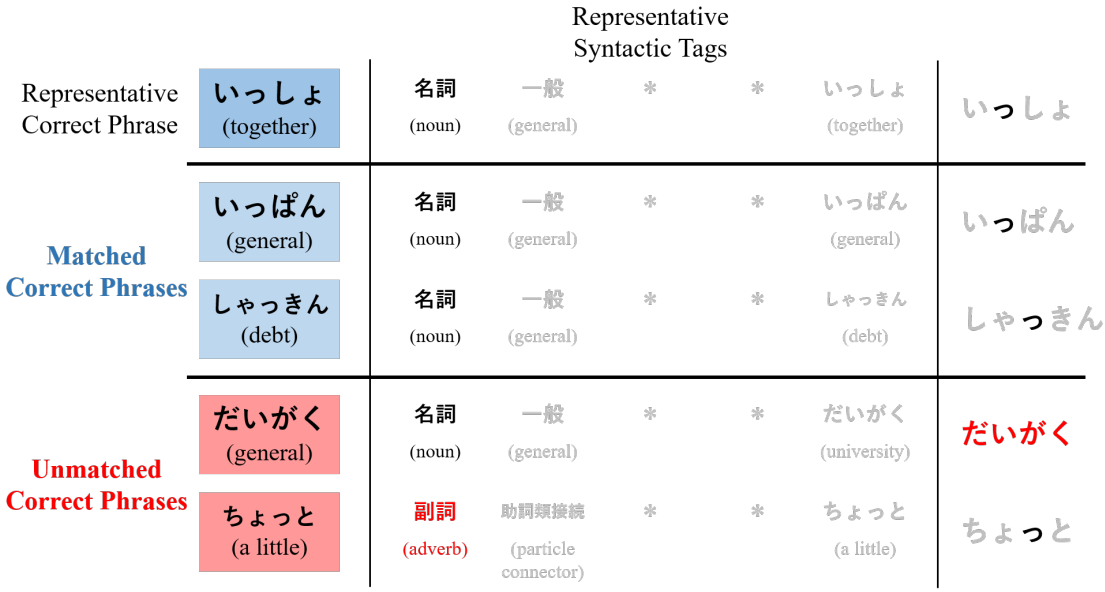

Figure 5-7: Phrase matching for the example character syntactic rule representing the deletion of the character っ within nouns. Four candidate phrases that match to the rule are shown below the first horizontal divider.

An example set of representative syntactic tags and representative characters for this rule is shown in Figure 5-7. In the figure, the requisite (one-masked) and non-

requisite (zero-masked) characters of each phrase are shown in normal and faded grey text, respectively, at the far-right of the figure. Thus, we have:

$$\mathbf{C}_{char} = \begin{bmatrix} \text{い} & \text{っ} & \text{し} & \text{ょ} \end{bmatrix} \quad \mathbf{B}_{char} = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}$$

This results in:

$$\mathbf{R}_{char} = \begin{bmatrix} 0 & \text{っ} & 0 & 0 \end{bmatrix}$$

which indicates that the constraint $\mathbf{R}_{char}$ places on matches $\mathbf{c}'$ to the rule is that っ be one of the characters $c'_{0,k'}$ of the first token of $\mathbf{c}'$. This condition is matched by three of the four candidate phrases of the figure, namely: いっぱん ("general"), しゃっきん ("debt"), and ちょっと ("a little"). Note that the match offset for いっぱん is $h = 0$, while for the other two matches, it is $h = 1$. The last candidate phrase, だいがく ("university") does not match the rule as the character っ is not present in the spelling of its first token.

Additionally, ちょっと ("a little") does not satisfy the constraint imposed by $\mathbf{R}_{tag}$ that c'_0 be a noun; hence, only the two candidate phrases いっぱん ("general") and しゃっきん ("debt") match to the rule.

As more 1 values are added to $\mathbf{B}_{char}$, the rule becomes more restrictive in matching phrases. If more 0 values are present, the rule becomes less specific. This presents the same optimization issue as for the matrix $\mathbf{B}_{tag}$ introduced in Section 5.2. Thus, $\mathbf{B}_{char}$ is determined manually along the same informal criterion used to select the "optimal" $\mathbf{B}_{tag}$.

5.4.2 Character Error Mapping

The character correction error mapping replaces the error mapping described in Section 5.3, and serves the same purpose, defining how the rule error can be introduced into an arbitrary correct phrase matched to the rule. However, instead of deriving the tokens of the representative error phrase $\mathbf{e}$ from those of the representative correct phrase $\mathbf{c}$, the character error mapping instead derives individual characters of $\mathbf{e}$ from

those of $\mathbf{c}$.

The character error mapping is defined using three character-level relations: **insertion**, **deletion**, and **preservation**. These largely mirror the relations of the same names defined in Section 5.3. Specifically, each relation is represented as an equation $e_{j,m} = \text{OPERATION}(x)$[12]. This shows that given an arbitrary correct phrase $\mathbf{c}'$ matched by the rule, and a corresponding error phrase $\mathbf{e}'$ derived from it, the character $e'_{j,m'} = e_{j,m+h}$ is derived from x, where x is either a constant character or some character $c'_{i,k'} = c'_{i,k+h}$ of $\mathbf{c}'$, and h is the match offset of $\mathbf{c}'$.

We define the three relations below:

Insertion

The insertion relation is defined as $e_{j,m} = \text{INSERT}(s)$, where s is a single character. This implies that regardless of what $\mathbf{c}'$ is, for $\mathbf{e}'$, we have $e'_{j,m+h} = s$.

Deletion

The deletion relation is the inverse of the insertion relation, and is represented as $* = \text{DELETE}(c_{i,k})$, where s is a single character. This relation indicates that for a given $\mathbf{c}'$, $c'_{i,k+h}$ is not used to derive any part of the corresponding $\mathbf{e}'$.

Preservation

The preservation relation is represented as the equation $e_{j,m} = \text{PRESERVE}(c_{i,k})$, and shows that that $c'_{i,k+h} = e'_{j,m+h}$ for all $\mathbf{c}'$ and corresponding $\mathbf{e}'$.

The character error mapping is constructed automatically by jointly iterating over the characters of $\mathbf{c}$ and $\mathbf{e}$ in order, and comparing their values. This is done according to the following process, which begins with $i = j = k = m = 0$:

[12]The notation for $e_{j,m}$ is identical to that used for $c_{i,k}$ in the Section 5.4.1 for the construction of $\mathbf{C}_{char}$. It represents the m'th character of the jth token of $\mathbf{e}$. We use similar notation for specific tokens of $\mathbf{c}'$ and $\mathbf{e}'$ in the remainder of this section.

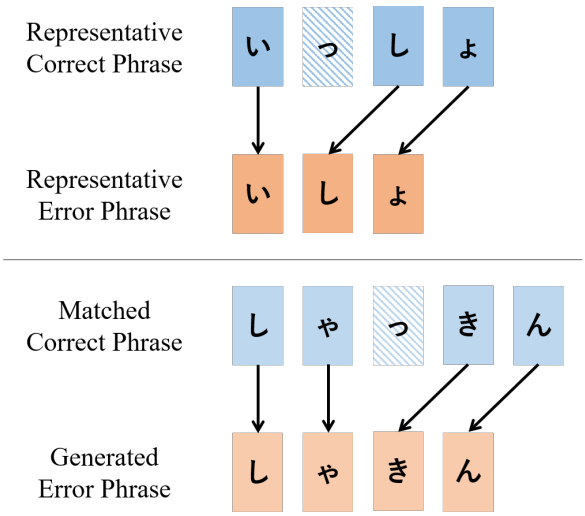

Figure 5-8: Mapping for an example character syntactic rule representing the deletion of the character っ within nouns.

1. If $k = \text{length}(c_i) - 1$, i is incremented by 1 and k is set to 0[13].

2. If $m = \text{length}(e_j) - 1$, j is incremented by 1 and m is set to 0.

3. If $c_{i,k} = e_{j,m}$, the relation $e_{j,m} = \text{PRESERVE}(c_{i,k})$ is added and both k and m are incremented by 1.

4. If $c_{i,k} \neq e_{j,m}$ and $e_{j,m+1} = c_{i,k+p}$ for some $p > 1$, the relation $* = \text{DELETE}(c_{i,k})$ is added and k is incremented by 1.

5. If $c_{i,k} \neq e_{j,m}$ and $e_{j,m+p} = c_{i,k+1}$ for some $p > 1$, the relation $e_{j,m} = \text{INSERT}(s)$ is added and m is incremented by 1.

Additionally, during error generation, the preservation relation is implied for all $e'_{i,m+h}$ that are not explicitly given a relation from the above. Specifically for the cases (i) $m < 0$ or (ii) $m \geq \text{length}(e_j)$[14], the relation $e_{j,m} = \text{PRESERVE}(c_{i,k})$ is assumed to be true, with $k = m$ in the former case and $k = m - (l' - l)$ in the latter case[15].

An example character error mapping for the aforementioned rule involving the deletion of っ is shown in Figure 5-8. In the figure, deleted characters are shown by

[13]The value of length(x), with x a token, is the number of characters within x.
[14]Such cases may occur if $h \neq 0$ or if $\text{length}(c'_i) \neq \text{length}(c_i)$.
[15]As per previous notation, l' is the length in tokens of $\mathbf{e}'$ and l is the length in tokens of $\mathbf{c}'$.

the half-shaded boxes in each of the correct phrases, while preserved characters are indicated with solid arrows. Thus, we have the following relations:

$$
\begin{aligned}
* &= \text{DELETE}(c_{0,1}) \\
e_{0,0} &= \text{PRESERVE}(c_{0,0}) \\
e_{0,1} &= \text{PRESERVE}(c_{0,2}) \\
e_{0,2} &= \text{PRESERVE}(c_{0,3})
\end{aligned}
$$

Note that for the matched phrase shown in the figure, しゃっきん (“debt”), we have that $h = 1$; hence, no relation is given for $e'_{0,0}$, where $m = -1$. Thus, the preservation relation $e_{0,-1} = \text{PRESERVE}(c_{0,-1})$ is inferred and we have that $e'_{0,0} = e'_{0,-1+h} = c'_{0,-1+h} = c'_{0,0}$. This allows us to generate the corresponding error as the word しゃきん (“reward”).

Chapter 6

Parallel Data Synthesis

In this chapter, we overview the process for constructing parallel error-correct sentences from the syntactic and character syntactic rules described in Chapter 5. We describe the process for a single rule in the following sections; however, note that in constructing a full synthetic training corpus, this process is repeated for multiple rules.

6.1 Sentence Matching

The first step of the data synthesis process is the extraction of correct phrases matching to the rule from an arbitrary corpus of correct Japanese. This is accomplished by first parsing the entire corpus through MeCab to construct for each sentence (i) a vector of the sentence's tokens and (ii) a matrix of the sentence's syntactic tags. The former of these is used for the error generation process described in Section 6.2 while the latter is used to check candidate matched phrases to see if they satisfy the constraints enforced by the requisite syntactic tags of the rule. In addition, the characters of each sentence are compiled into another matrix to test candidate matched phrases against constraints enforced by requisite characters if the rule is a character syntactic rule.

We now consider this process for a single sentence s; however, note that in practice, the entire matching process is done in parallel across all sentences of the corpus. Let

the tokens outputted by MeCab for s be compiled into a vector $\mathbf{x}$ as follows:

$$\mathbf{x} = [x_0, x_1, \cdots, x_m]$$

where x_i is the ith token outputted for s and m is the length (in tokens) of s.

Then, syntactic tags outputted by MeCab for s are formed into a matrix $\mathbf{T}$ as follows:

$$\mathbf{T} = [\mathbf{t}_0, \mathbf{t}_1, \cdots, \mathbf{t}_m]^T$$

Here, $\mathbf{t}_i$ is a vector of five of the syntactic tags corresponding to the token x_i. $\mathbf{t}_i$ is constructed in a format identical to that of the $\mathbf{t}_i$ presented in Section 5.2.[1]

Finally, a matrix of the characters of s, represented as $\mathbf{C}$, is constructed as follows:

$$\mathbf{C} = \begin{bmatrix} x_{0,0} & x_{0,1} & \cdots & x_{0,n} \\ x_{1,0} & x_{1,1} & \cdots & x_{1,n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{m,0} & x_{m,1} & \cdots & x_{m,n} \end{bmatrix}$$

Here, $x_{i,k}$ is the kth character of the ith token of s, and n is the maximum length (in characters) of any of the tokens x_i. In a similar fashion to the matrix $\mathbf{C}_{char}$ described in Section 5.4.1, the matrix is padded with zeros[2].

Correct phrases that match to the rule are then extracted by performing a rolling search of stride 1 over s. Consider a candidate matched phrase $\mathbf{c}'$ beginning at the

[1]Specifically, we have that each $t_{i,k}$ of $\mathbf{t}_i$ is the tag outputted for the following features:

- $t_{i,0} \rightarrow$ 品詞 ("part-of-speech")
- $t_{i,1} \rightarrow$ 品詞細分類1 ("part-of-speech sub-category 1")
- $t_{i,2} \rightarrow$ 活用型 ("inflection form")
- $t_{i,3} \rightarrow$ 活用形 ("base form")
- $t_{i,4} \rightarrow$ 原形 ("lemma")

[2]Specifically, for any token x_i of length $n' < n$, each of the entries $x_{i,k}$ of $\mathbf{C}$ with $k > n'$ are assigned a value of 0.

index j of s. In other words, let:

$$\mathbf{c}' = [x_j, x_{j+1}, \cdots, x_{j+l-1}]$$

where l is the length of $\mathbf{c}$, the representative correct phrase of the rule. Then, we have that $\mathbf{c}'$ corresponds to the slice $\mathbf{T}[j : j + l]$ of $\mathbf{T}$, where $\mathbf{T}[j : j + l]$ is as follows:

$$\mathbf{T}[j : j + l] = [\mathbf{t}_j, \mathbf{t}_{j+1} \cdots, \mathbf{t}_{j+l-1}]^T$$

As mentioned in Section 5.2, $\mathbf{c}'$ matches to the rule if its corresponding syntactic tags $\mathbf{C'}_{tag}$ satisfy the condition[3]:

$$\mathbf{C}'_{tag} \circ \mathbf{B}_{tag} = \mathbf{R}_{tag}$$

where $\mathbf{B}_{tag}$ is the binary mask array used to produce the rule's requisite syntactic tag matrix, $\mathbf{R}_{tag}$. Thus, $\mathbf{c}'$ matches to the rule if the above equation is satisfied for $\mathbf{C}'_{tag} = \mathbf{T}[j : j + l]$

If the rule is a character syntactic rule, the additional constraints enforced by the rule's requisite characters[4] are also checked. This is done by comparing the slices $\mathbf{C}'_{char} = \mathbf{C}[j : j + l]$

$$\mathbf{C}[j : j + l] = \begin{bmatrix} x_{j,0} & x_{j,1} & \cdots & x_{j,n} \\ x_{j+1,0} & x_{j+1,1} & \cdots & x_{j+1,n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{j+l-1,0} & x_{j+l-1,1} & \cdots & x_{j+l-1,n} \end{bmatrix}$$

corresponding to $\mathbf{c}'$ to the requisite character matrix, $\mathbf{R}_{char}$, of the rule. $\mathbf{c}'$ still matches to the rule if the following is true for each $0 < i \leq l$:

$$\exists h \quad \text{s.t.} \quad \forall k \in \mathbf{R}_{char}[i, k] \neq 0, x_{j+i,k+h} = c_{i,k}$$

[3]Note that $\circ$ denotes the element-wise or Hadamard product between the two matrices.
[4]See Section 5.4.1.

In other words, for each token x_{j+i} of $\mathbf{c}'$, there must exist some match offset h so that the characters of x_{j+i} align with the requisite characters of the rule.

6.2 Error Generation

Once all matches $\mathbf{c}'$ to the rule within the corpus have been found, a corresponding error phrase $\mathbf{e}'$ is generated for each match. If the rule is a syntactic rule, this is done by applying the relations of the rule's error mapping to $\mathbf{c}'$, in the manner described in Section 5.3. If the rule is a character syntactic rule, the relations of character error mapping, as described in Section 5.4.2, are used instead.

Finally, the new parallel sentence pair is generated. As before, let s be the sentence from which $\mathbf{c}'$ originated, and let

$$\mathbf{c}' = [x_j, x_{j+1}, \cdots, x_{j+l-1}]$$

where j is the index where $\mathbf{c}'$ begins in s and l is again the length of the representative correct phrase $\mathbf{c}$ of the rule. Then the parallel error sentence, s_{err}, is obtained by replacing the tokens of s comprising $\mathbf{c}'$ with the tokens of $\mathbf{e}'$. In other words, if s is given as the tokens

$$\mathbf{x} = [x_0, x_1, \cdots, x_{j-1}, \mathbf{c}', x_{j+l}, \cdots, x_m]$$

then s_{err} is constructed from the tokens

$$\mathbf{x}_{err} = [x_0, x_1, \cdots, x_{j-1}, \mathbf{e}', x_{j+l}, \cdots, x_m]$$

The final synthesized corpus is obtained by aggregating all pairs of s and s_{err} generated across the set of rules.

Chapter 7

Experiment

In this chapter, we describe the experiment we run to test the efficacy of the proposed data synthesis framework. In Section 7.1, we review the overall design and goals of the experiment. In Section 7.2, we introduce the datasets used to train and evaluate our experimental models. In Section 7.3, we present the architecture used to construct each of the trained NMT models. Finally, in Section 7.4, we overview the metrics used to evaluate the trained models' performance on the evaluation datasets.

7.1 Experimental Design

Our primary experiment is intended to evaluate our data synthesis framework with regards to three main goals: (i) improving Japanese GEC system performance through the expansion of existing training corpora, (ii) the successful targeting and correcting of specific Japanese grammatical errors, and (iii) the training of a reasonable Japanese GEC system in the absence of real-world training data. The latter of these two goals correspond to the two gaps in the applicability of current AEG methods as identified in Section 2.2.

To accomplish this, we synthesize a parallel training corpus using our data synthesis framework and use this to expand the existing Japanese Lang-8 corpus[1]. Specif-

[1]See Section 4.3.

ically, we first sample error-correct sentence pairs from the Teacher corpus[2], to craft 362 syntactic and 38 character syntactic rules. Then, using the Yourei corpus[3] as a source of correct Japanese sentences, these 400 rules are used to generate over 150 million parallel sentence pairs using the data synthesis process described in Chapter 6[4]. In the following sections, we refer to this collection of synthesized sentence pairs as the *original synthetic corpus*.

Then, to test the first goal, we train two models: (i) a model trained on an expanded training set containing both Lang-8 and synthesized pairs and (ii) a model trained using just the Lang-8 data alone. Both models are constructed with a standard out-of-the-box NMT architecture representative of recent GEC systems. The two trained models are then evaluated on a set of evaluation corpora that includes the Teacher corpus. Finally, we compare the relative performances of the two models to determine if data synthesized by our framework can improve the corrective ability of Japanese GEC systems.

To test the second goal, we divide the primary evaluation set into subsets based on the individual rules used to construct the synthesized data. Specifically, for each rule, we determine the set of evaluation sentence pairs containing the rule error, and check the relative performances of two models. Ideally, if the synthetic data is effective in targeting specific rules, the model trained with the additional synthetic data should perform better on these sentence pairs.

Finally, to test the third goal, we train an additional model of the same architecture using only the newly synthesized data and test determine its overall performance on the evaluation sets.

[2]See Section 4.2.
[3]See Section 4.1.
[4]Samples of these synthesized sentences are shown for four of the 400 rules in Appendix A.

7.2 Datasets

7.2.1 Training Sets

We compile three different training sets, one for each of the trained models. These are as follows:

- *synthetic training set* - $2,179,569$ sentence pairs obtained via stratified sampling from the original synthetic corpus[5].

- *Lang-8 training set* - $1,609,250$ sentence pairs randomly sampled from the Lang-8 corpus.

- *merged training set* - Dataset of $3,788,819$ sentence pairs formed by merging the above synthetic and Lang-8 training sets.

7.2.2 Development Sets

We compile three development sets to aid in final model selection, as follows:

- *synthetic development set* - $124,576$ sentence pairs obtained via stratified sampling from the original synthetic corpus. Each of the sentence pairs is distinct from any in the synthetic training set.

- *Lang-8 development set* - $89,402$ randomly sampled sentence pairs from the Lang-8 corpus not used in the Lang-8 training set.

- *merged development set* - Dataset of $213,978$ sentence pairs formed by merging the above synthetic and Lang-8 development sets.

[5]The sampling method used preserves the relative frequency of matched correct pairs of each rule in the source (Yourei) corpus. In other words, if Rule A matches x phrases in the Yourei corpus and Rule B matches y phrases, then the sampling method satisfies the equation $x' > y'$, where x' is the number of sentence pairs sampled from the data synthesized using Rule A, and y' the number of sentence pairs sampled from the data synthesized using Rule B.

7.2.3 Evaluation Sets

Finally, we construct three different evaluation sets, as follows:

- *teacher evaluation set* - All $6,375$ sentence pairs of the Teacher corpus. As many of the error sentences are given multiple possible corrections, this dataset contains only $4,375$ unique error sentences.

- *Lang-8 evaluation set* - The remaining $89,402$ sentence pairs from the Lang-8 corpus not used in either the Lang-8 training or development corpora. Accounting for error sentences with multiple given corrections, there are $88,768$ unique error sentences.

- *synthetic evaluation set* - $146,388$ sentence pairs obtained via stratified sampling of the original synthetic corpus. Each sentence pair is distinct from any in the synthetic training or development corpora. Additionally, each error sentence is unique, resulting in $146,388$ unique error sentences.

As mentioned in Section 2.4, we consider the teacher evaluation corpus to be our primary evaluation set as its sentences most closely resemble the data that would be seen in a real-world GEC system. This is due to the stylistic similarity between the Teacher corpus sentences and Japanese learners' writing in the classroom setting.

We reserve portions of the Lang-8 and original synthetic corpora for the remaining two evaluation sets to show the impact that stylistic differences in Japanese can have on the ability of Japanese GEC systems to generalize effectively.

7.3 Models

We use the FConv module of Facebook AI Research's open-source Python library, *fairseq* [23] as the basis for our trained models. This module implements a convolutional sequence-to-sequence (seq2seq) model that we consider a reasonable candidate to represent current Japanese GEC systems.

A convolutional seq2seq model is comprised of two jointly trained convolutional neural networks (CNNs), an encoder network that processes an input sequence into an embedded state, and a decoder network that uses that embedded state to generate the output sequence. A detailed overview of the the convolutional seq2seq architecture is given in the original paper that introduced it [9].

In our selected model architecture, both the encoder and decoder networks are 7-layer CNNs with identical structure. In both networks, the first three layers have 192 convolutional units, the following two layers have 384 units, and the final two layers have 768 units. All of the convolutional units are set with a window size of 3 and a stride of 1. Moreover, a dropout layer with a dropout probability of 0.15 is added after each convolutional layer and the decoder's output. Finally, an embedding size of 192 is used for the encoder input, the encoder output, and the decoder output.

7.3.1 Training

We use MeCab to tokenize each training set and then apply *fairseq*'s built-in pre-processing script to limit the vocabulary size to $50,000$ tokens for both the source (error) and target (correct) data. Tokens outside the vocabulary are replaced with the library's default "<unk>" unknown token.

We then train three models of the above architecture, one on each training set. Each model is trained for 30 epoches using *fairseq*'s built-in training script on a pair of NVIDIA RTX 2080 GPUs. The learning rate is set to the default value of 0.1, and a batch size of 64 is used. To speed up training, we opt to use half-point floating point precision. For each model, we select the best-performing epoch on the corresponding development set[6] to use as the final model during evaluation.

[6]The development set for the model trained on X training set of Section 7.2.1 is the corresponding X development set of Section 7.2.2. For example, the model trained on the synthetic training set is validated using the synthetic development set.

7.4 Metrics

Though the current standard metric used in evaluating GEC models is the MaxMatch scorer [5], no gold-standard edit labels exist for any of our evaluation sets. Thus, we use the GLEU score [20] as our primary evaluation metric, as it provides a reasonable estimate for a model's aggregate corpus across each corpus.

In addition to the GLEU score, for both the synthetic and Teacher evaluation sets, we also report each model's **phrase-level binary accuracy**, which is calculated as follows:

Let e_j be the jth error sentence of an evaluation set, and let $\{c_{j_1}, c_{j_2}, \cdots c_{j_i}\}$ be the set of reference corrections corresponding to e_j. Moreover, let $\mathbf{e}_j$ and $\mathbf{c}_{j_i}$ be the tokenized error and correction phrases demarcated within e_j and c_{j_i}, respectively[7]. Then, the for each e_j, a binary accuracy value p_j, is outputted as follows:

- $p_j = 1$ if the model aligned to $\mathbf{e}$ matches exactly with any of the possible corrections, $\{\mathbf{c}_{j_1}, \mathbf{c}_{j_2}, \cdots \mathbf{c}_{j_i}\}$
- $p_j = 0$ if the aligned model output does not match any $\mathbf{c}_{j_i}$

The overall phrase-level binary accuracy for the evaluation set, $\overline{p}$ is then calculated by averaging the above binary values over the set of unique error sentences. In other words:

$$\overline{p} = \frac{\sum_j p_j}{n}$$

where n is the number of unique error sentences in the evaluation set.

We choose to report this metric as a phrase-level metric is better able to capture localized grammatical errors when compared to a metric calculated on a sentence-level (e.g. GLEU). This is especially relevant to the teacher evaluation set where most sentence pairs contain just a single error.

We extract the error/correction phrase boundaries for the teacher evaluation set from the pre-existing annotations. For the synthetic evaluation corpus, these bounds

[7]Note that $\mathbf{e}_j$ is assumed to contain all the errors within e_j and $\mathbf{c}_{j_i}$ all the corresponding corrections within c_{j_i}. Hence, each sentence e_j or c_{j_i} is expected to contain only one demarcated phrase.

are derived from the bounds of $\mathbf{c}'$ and $\mathbf{e}'$ as defined in Section 6.2. Since no phrase boundaries exist for the Lang-8 corpus, we use entire sentences as phrases instead (e.g. $\mathbf{e}_j = e_j$ and $\mathbf{c}_{j_i} = c_{j_i}$).

Chapter 8

Results and Discussion

In this chapter, we introduce and discuss the results of our experiments. We begin in Section 8.1 with the overall results recorded on each evaluation set. Then, in Section 8.2 we divide the teacher corpus into **in-rule** and **out-of-rule** subsets and re-analyze each model's performance. Finally, in Section 8.3, we look at the models' performance on individual rules.

We report and discuss the results of each trained model under aliases as follows:

- *Synthetic* - Model trained on only the synthetic training set
- *Lang-8* - Model trained on only the Lang-8 training set
- *Merged* - Model trained on the merged training set that combines both Lang-8 and synthetic data

For additional comparison, we also report results for a *NULL* model, which simply outputs the input it receives.

8.1 Overall Results

We calculate the GLEU and phrase-level binary accuracy values for each of the trained models on the three evaluation sets in Table 8.1 below. The values for the two metrics are reported in the columns labeled as "**GLEU**" and "**Acc"**, respectively, and the

evaluation set each metric corresponds to is given by the parentheses in the top row. Note that the phrase-level accuracy is actually equivalent to a sentence-level statistic for the Lang-8 evaluation set as no specific error/correction bounds are given.

	(Teacher)		(Synthetic)		(Lang-8)	
Model	**GLEU**	**Acc**	**GLEU**	**Acc**	**GLEU**	**Acc**
NULL	0.610	0.043	0.904	0.064	0.766	0.299
Synthetic	0.660	0.484	0.970	0.835	0.690	0.202
Lang-8	0.699	0.449	0.775	0.134	0.836	0.487
Merged	**0.726**	**0.559**	**0.977**	**0.843**	**0.838**	**0.493**

Table 8.1: Performance of trained NMT models on the three evaluation sets. The performance of the best model for each metric is shown in bold.

In general, the merged model[1] appears to exhibit the best performance, achieving the highest value for both metrics in each evaluation set. This result suggests that the data synthesized by our framework is effective at improving the performance of GEC systems. Indeed, we can conclude that the merged model is able to correct most if not all of the same errors as the Lang-8 model, as it achieves slightly higher GLEU and phrase-level accuracy on the Lang-8 evaluation set. Moreover, we can infer that the model is able to correct many errors that the Lang-8 model cannot, as evinced by the significant discrepancy in the GLEU and phrase-level accuracies on the teacher evaluation set.

Additionally, on the synthetic evaluation set, while both the merged model and the synthetic model perform very well, with each model reporting over 80% phrase-level accuracy, the Lang-8 model reports quite poor accuracy and actually performs worse than the NULL model when the GLEU score is considered. This is most likely a generalization issue, as the synthetic data that the first two models train on is nearly identical to that of the evaluation set, whereas the sentence pairs of the Lang-8 training set are dissimilar. This generalization issue also explains the poor performance of the synthetic model on the Lang-8 evaluation set. Together, these observations imply that any model trained using only our synthetic data will only perform well when similar data (i.e. data resembling the training data) is inputted.

[1]For the purposes of brevity we refer to each model with the same alias as used in the data tables.

Finally, the extremely high GLEU scores reported for the synthetic and merged models on the synthetic evaluation set showcase how this metric can be misleading when both error and reference correction sentences are nearly identical. This is true for both the teacher and synthetic evaluation sets, due to localization of errors in both sets. This suggests that we should consider the phrase-level metric to be more informative for those two evaluation sets.

8.2 In-rule vs. Out-of-rule Rules

One of the hidden abilities of our data synthesis framework is the ability to essentially "classify" phrase pairs[2]. Specifically, given a correction phrase $\mathbf{c}$ and corresponding error phrase $\mathbf{e}$, let $\{\mathbf{R}_1, \mathbf{R}_2, \cdots, \mathbf{R}_i\}$ be the set of rules for which $\mathbf{c}$ is a match. Furthermore, let $\mathbf{e}_i$ be the error phrase created by applying the error mapping of $\mathbf{R}_i$ to $\mathbf{c}$. Then, if $\mathbf{e} = \mathbf{e}_i$ for some i, we can conclude that the grammatical error of $\mathbf{e}$ is equivalent to the rule error of $\mathbf{R}_i$, and that $\mathbf{R}_i$ thus "represents" the pair of $\mathbf{c}$ and $\mathbf{e}$. In this way, for some sentence pairs, we can determine the exact error type contained in the error sentence.

Extending upon this idea of rule representation, intuitively, if a rule $\mathbf{R}_i$ is used to generate part of a synthetic training set, we can expect that a model trained on that corpus to perform well on error-correct sentence pairs represented by it $\mathbf{R}_i$. Conversely, for sentences pairs not represented by any $\mathbf{R}_i$, we should expect rather poor model performance, since the synthetic corpus does not contain the grammatical errors of the error sentences.

This implies that we can split our evaluation sets into two distinct subsets as follows:

- an **in-rule** subset consisting of the unique error sentences e_j with reference corrections $\{c_{j_1}, c_{j_2}, \cdots c_{j_i}\}$ for which some sentence pair (e_j, c_{j_1}) is represented by one of the 400 rules used in the construction of the synthetic training set.

[2]Note that this is not unlike the way in which the rule-based models originally developed for GEC functioned; however, in this case the classification is done a sentence pair, rather than just the error sentence.

- an **out-of-rule** subset consisting of the remaining unique error sentences that are not part of any error-correct pair represented by some rule.

Within this split, we would expect the merged and synthetic models (e.g. the models trained with some synthetic data) to perform significantly better on the in-rule subset than the out-of-rule subset.

We therefore split our primary evaluation set, the teacher evaluation set, into in-rule and out-of-rule subsets as above. The resulting in-rule subset contains $2,651$ of the $4,375$ unique error sentences; the out-of-rule subset therefore contains $1,724$ unique error sentences. The GLEU and phrase-level binary accuracy values of each trained model on the these two subsets are given below in Table 8.2.

	(Overall)		(In-rule)		(Out-of-rule)	
System	**GLEU**	**Acc**	**GLEU**	**Acc**	**GLEU**	**Acc**
NULL	0.610	0.043	0.618	0.049	0.603	0.059
Synthetic	0.660	0.484	0.810	**0.714**	0.547	0.131
Lang-8	0.699	0.449	0.767	0.547	0.646	0.298
Merged	**0.726**	**0.559**	**0.817**	0.692	**0.654**	**0.355**

Table 8.2: Performance of trained NMT models on the teacher evaluation set. Values for the GLEU and phrase-level binary accuracy metrics are shown over (i) the overall evaluation set, as well as the (ii) in-rule and (iii) out-of-rule subsets.

Confirming the above hypothesis, we find that there is a significant discrepancy in the performance of the merged and synthetic models on the in-rule vs. out-of-rule subsets of the teacher evaluation set, with both models performing far better on the former subset than on the latter. Moreover, the synthetic model actually achieves the highest phrase accuracy on the in-rule subset by a significant margin. These observations both suggest that models trained using our synthesized data learn to correct most of the grammatical errors present in that training data. In other words, we can expect that if we construct synthetic data using rules containing a grammatical error X, a model trained using that synthetic data will correct most instances of X. Consequently, this also implies that the performance of such a model could easily be increased by using a more comprehensive rule set, since our rule-based framework is not constrained to any specific set of rules.

The high performance of the synthetic model on the in-rule subset also demonstrates that even in the absence of natural parallel training corpora, our synthesized data can be used to train a relatively successful GEC model. Such a model would not generalize well to grammatical errors outside the training corpus, as demonstrated by the synthetic model's low accuracy on the out-of-rule subset. However, in an extremely data-scarce environment, this model would still be a significant improvement on any alternative models that could be developed.

Finally, we find that the Lang-8 model also performs better on the in-rule subset on both metrics. This interesting observation suggests that the grammatical errors of the in-rule error sentences are either well-represented within the Lang-8 training set already or relatively easy to correct. Either of these hypotheses are reasonable, given that the seed corpus of the syntactic rules, the Teacher corpus, is itself inspired by common learners' mistakes.

8.3 Individual Rule Results

We also use the idea of rule representation to analyze the performance of the trained models with respect to individual rules. This is important as different rules represent grammatical errors of differing complexity. For example, consider the relative difficulty of the rule errors of **token-specific** vs. **token-independent** rules, which we distinguish as follows:

Let $\mathbf{R}_i$ be a syntactic rule and let $\mathbf{C}_i = \{\mathbf{c}_{i_1}, \mathbf{c}_{i_2}, \cdots \mathbf{c}_{i_j}\}$ be the set of of all theoretical phrases that match to $\mathbf{R}_i$. Note that $\mathbf{c}_{i_j}$ is a phrase that satisfies the syntactic requirements to match to $\mathbf{R}_i$, but does not necessarily have to occur in correct Japanese text. Then, let $\mathbf{E}_i = \{\mathbf{e}_{i_1}, \mathbf{e}_{i_2}, \cdots \mathbf{e}_{i_j}\}$, where $\mathbf{e}_{i_1}$ is the error phrase generated from $\mathbf{c}_{i_1}$. In other words, $\mathbf{E}_i$ is the set of all possible error phrases that could theoretically be generated using $\mathbf{R}_i$. Given this, we consider $\mathbf{R}_i$ to be token-specific if there exists some grammatically correct $\mathbf{e}_{i_1}$ within $\mathbf{E}_i$; otherwise, we consider $\mathbf{R}_i$ to be token-independent.

An example of such a rule is given by the representative phrases as follows:

$$\mathbf{e} = [\text{授業, を, 始まる}] \quad \mathbf{c} = [\text{授業, が、始まる}]$$

In English these are:

$$\mathbf{e} = [\text{class, WO, begin}] \quad \mathbf{c} = [\text{class, GA, begin}]$$

This rule represents a common particle substitution error in which a nominative marker が that follows a noun and precedes a verb, is replaced by an accusative marker を. However, in correct Japanese, transitive verbs such as 始める ("to start") are supposed to be marked by を, meaning that the above substitution is only erroneous for intransitive verbs like 始まる ("to begin"). Conversely, this also means that the ideal correction implied by the rule (the swapping of を back to が) cannot be blindly applied to any error phrase $\mathbf{e}'$ matching the rule, as this would introduce new errors if $\mathbf{e}'$ contains a transitive verb. Thus, the validity of the above rule to a given error phrase $\mathbf{e}'$ depends on the specific tokens of $\mathbf{e}'$[3].

An example of a token-independent rule is the example rule used throughout Chapter 5 with representative phrases as follows:

$$\mathbf{e} = [\text{楽しい, な, ゲーム}] \quad \mathbf{c} = [\text{楽しい, ゲーム}]$$

In English:

$$\mathbf{e} = [\text{fun, NA, game}] \quad \mathbf{c} = [\text{fun, game}]$$

This rule represents the insertion of the copula particle な in between a root-form adjective and the noun it modifies. This insertion is always grammatically incorrect, regardless of which noun-adjective pair is considered, making the validity of the rule to a given error phrase $\mathbf{e}'$ independent of the specific tokens of $\mathbf{e}'$.

[3]Note that while the phrase $\mathbf{c}' = [\text{授業, が, 始める}]$, resulting in $\mathbf{e}' = [\text{授業, を, 始める}]$, $\mathbf{c}'$ is not observed in correct Japanese. Hence, no grammatically correct error sentences will be generated for token-specific rules during data synthesis. This problem therefore does not concern the error generation component of our framework and is only relevant during model inference and analysis.

Rule #	**Representative Phrases** (Error ⇒ Correct)	**Rule Explanation**	**n**	**Acc$_S$**	**Acc$_L$**	**Acc$_M$**
Token-specific Rules						
1	ことが教える ⇒ ことを教える	verb-particle agreement error	51	0.490	0.431	**0.588**
2	バスを乗る ⇒ バスに乗る	verb-particle agreement error	43	0.605	0.698	**0.884**
3	授業を始まる ⇒ 授業が始まる	verb-particle agreement error	43	0.442	0.581	**0.674**
4	図書館に勉強 ⇒ 図書館で勉強	verb-particle agreement error	36	0.333	0.611	**0.667**
5	人がある ⇒ 人がいる	subject-verb agreement error	30	0.700	0.667	**0.867**
Token-independent Rules						
6	吸って ⇒ 吸いて	conjugation error, connecting-form verb	103	**0.835**	0.485	0.787
7	作るまし ⇒ 作りまし	conjugation error, ます-form verb	84	**1.000**	0.655	0.917
8	楽しいでした ⇒ 楽しかった	conjugation error, past-tense adjective	48	**0.979**	0.938	0.896
9	甘いのケーキ ⇒ 甘いケーキ	particle insertion error	44	**0.910**	0.704	0.523
10	好き音楽 ⇒ 好きな音楽	particle deletion error	39	**0.821**	0.333	0.744

Table 8.3: Phrase accuracy of trained NMT models on selected token-specific and token-independent. The accuracies for the synthetic, Lang-8 and merged models are given in the columns "**Acc**$_S$", "**Acc**$_L$", and "**Acc**$_M$", respectively. The values in the column labeled "**n**" denote the number of unique error sentences represented by each rule.

Intuitively, we should expect the grammatical errors of token-specific rules to be more difficult to correct, as we can these rules are usually dependent on token characteristics that are not representable with only syntactic information (e.g. verb transitivity). Such characteristics are highly context-specific and should therefore be more difficult for models to distinguish. To confirm this idea, we determine the subsets of the teacher evaluation set represented by each rule. Then, we calculate the phrase-level accuracy of the three models on the subsets corresponding to 5 well-represented rules of each category (token-specific and token-independent) in Table

8.3; samples of the models' outputs on some of these individual rules are given in Appendix B.

Overall, the results show that the synthetic and merged models generally perform better on token-independent rules than on token-specific rules. Though the dichotomy is not as apparent for the Lang-8 model, in general the data reaffirms the above expectations of the relative difficulties of the two categories.

Importantly, the data also suggests that it is possible to learn most of these token-specific rules from the synthetic data. Indeed, the synthetic model corrects a little over half of all sentence pairs represented by a token-specific rule. Moreover, the merged model improves upon the performance of the Lang-8 model in all five token-specific rules. The latter observation suggests that our synthesized data can be used to improve the performance of GEC systems even on relatively complex errors.

Currently, several complex error types, such as those that involve long-distance relations between tokens, cannot be represented in our frameworks. However, if these are eventually incorporated, this observation implies that significant improvements to model performance on these specific errors could be obtained. This is important, given that many of these currently unrepresentable errors are among the ones that models struggle to learn the most.

Interestingly, the synthetic model outperforms the merged model in all five of the token-independent rules. This is particularly apparent for Rule 9, which represents the common particle insertion error where the possessive marker $\mathcal{O}$ is inserted between an adjective and the noun it modifies. Given the expected lower difficulty of token-independent rules, it is somewhat surprising that the merged model struggles to correct this particular grammatical error especially when the remaining two models, in particular the synthetic model, outperform it. This suggests that some fine tuning of the training scheme is needed to fully realize the potential of the above architecture. Thus, the reported performances of the trained models in this chapter could likely be improved upon significantly.

Chapter 9

Conclusion

The scarcity of existing parallel training corpora is currently the greatest limiting factor on the performance of GEC systems. This data scarcity problem is especially acute in Japanese GEC, where only one parallel corpus of significant size, the Lang-8 corpus [18], is publicly available. To address this challenge, in this thesis, we proposed a framework for synthesizing parallel error-correct sentences for Japanese GEC.

We evaluated the efficacy of our framework by first training two GEC models, each with the same standard NMT architecture, on two different training datasets: a training set derived from the Lang-8 corpus, and a training set created by augmenting the Lang-8 data with addition synthesized sentences. We then evaluated the two models on a novel evaluation corpus to determine if our synthesized data could be used to improve the performance of Japanese GEC systems.

Our main result shows that the addition of our synthesized data does significantly improve the predictive performance of the standard NMT architecture. Moreover, we show that our framework can be successfully applied to two primary use cases that are not readily covered by existing approaches to AEG. Firstly, we show that the proposed approach can be used to target specific grammatical errors by demonstrating that the model trained on the augmented dataset corrects grammatical errors represented in the synthetic data but not in the Lang-8 corpus. Additionally, we show that our framework can be used in the absence of existing parallel training corpora by training a third model using only our synthesized data. We find that such a model is successful

in generalizing to sentences containing grammatical errors represented in the synthetic data.

9.1 Challenges and Limitations

Despite the positive results, our framework has a number of limitations. For instance, a significant amount of manual work required to craft each individual syntactic. As shown in Chapter 5, while some parts of a syntactic rule can be derived algorithmically from seed sentence pairs, there are still several steps (namely the determination of the extent of the representative phrases and the requisite syntactic tags) that must be done by a human. We believe that it may be possible to automate even these processes with a clever optimization formulation; however, this will be a significant challenge, and is beyond the scope of this work.

Moreover, the current framework does not account for sentences with multiple or co-dependent errors. This is problematic, as natural learners' text (e.g. like that of the Lang-8 corpus) is not limited to a single error per sentence, as is true of the synthesized data. We believe that an extension to allow for the generation of error sentences containing multiple, complex errors is entirely feasible, and hope to approach this in future work.

Finally, our proposed framework is heavily dependent on the parsing ability of MeCab, as it relies on the morphological analyzer for all tokenization and syntactic tagging. As a result, when MeCab encounters a parsing error, both the rule construction and error generation components of our framework fail. This is problematic for many potential edge cases, including sentences that are primarily composed of kana[1], for which the token boundaries MeCab outputs are often inconsistent and even nonsensical. Unfortunately, alternatives to MeCab that resolve these deficiencies exist, and our framework can therefore not be adapted to account for these edge cases.

[1]It is not uncommon for Japanese language learners to write without the use of Kanji; hence this edge case would not be uncommon in real world data.

9.2 Future Work

In addition to the proposed solutions to the challenges above, there are many areas in which this work can be expanded upon. For example, the construction of syntactic rules could be converted into a regular expression-like syntax and allow for the representation of complex long-distance errors such as tense-agreement errors. As current GEC models often struggle to correct these types of errors, we believe that an expanded framework could be used to target these errors and improve model performance.

Additionally, the classification ability of our framework could also be used to annotate model outputs. For example, if the pair of a learner's input **c** and corresponding model output **e** are represented by a given syntactic rule, text explaining the rule error could be displayed to help teach the learner exactly why they were mistaken. We imagine that this form of in-depth feedback would be of immense use in any deployed Japanese GEC system.

And finally, while this framework was designed with Japanese GEC in mind, the ideas and concepts that motivate it are not unique to Japanese. In particular, we believe a similar approach could easily be formulated for Korean, as the language is both syntactically similar to Japanese and since a Mecab variant for Korean already exists.

Ultimately, we hope that the data synthesis method proposed in this thesis can enable the development of future models and research in Japanese GEC, as we believe that a mature Japanese GEC system would be of great benefit to Japanese language learners.

Appendix A

Sample Synthesized Data

In this appendix, we provide sample parallel error-correct sentences from the original synthetic corpus[1] created for our experiment. Each of the four tables below represents data synthesized using one of the 400 syntactic or character syntactic rules. The sentences of each table are labeled as $E\{n\}$, $C\{n\}$, $TE\{n\}$, or $TC\{n\}$. The integer value of n indicates that this is the nth sentence pair generated for the rule of the table.

The abbreviations E, C, TE, and TC indicate the following:

- E - Error sentence of the sentence pair
- C - Correct sentence of the sentence pair
- TE - English translation of the error sentence of the sentence pair
- TC - English translation of the correct sentence of the training pair

Additionally, the matched correct phrases $\mathbf{c}'$ and corresponding generated error phrases $\mathbf{e}'$ for each sentence pair are denoted by green and red text, respectively. For tables concerning character syntactic rules, the translations for $\mathbf{e}'$ and $\mathbf{c}'$ are given first as their rōmaji equivalents followed by the phrases' actual English translations immediately afterwards in parentheses. Finally, non-words or words with no English translation are given as "N/A".

[1] See Section 7.1.

Label	Sentence
E2884	レナペ族との関係は弱いものであり、オランダ人がコミュニポーで行った殺人を切掛けにキーフトの戦争が始まった。
C2884	レナペ族との関係は弱いものであり、オランダ人がコミュニポーで行った殺人を切っ掛けにキーフトの戦争が始まった。
TE2884	The relationship with the Lenape people being weak, the murders carried out by the Dutch in Communipaw were the kikake (“N/A”) of Kieft’s war.
TC2884	The relationship with the Lenape people being weak, the murders carried out by the Dutch in Communipaw were the kikkake (“start”) of Kieft’s war.
E2071	頭をできるだけ空ぼにすることですよ。
C2071	頭をできるだけ空っぽにすることですよ。
TE2071	(That’s) to make your head as karapo (“N/A”) as possible
TC2071	(That’s) to make your head as karappo (“empty”) as possible
E1354	そこで憲法は、自由といしょに、この平等ということをきめているのです。
C1354	そこで憲法は、自由といっしょに、この平等ということをきめているのです。
TE1354	That’s where the Constitution defines, isho (“will; farewell note”) with freedom, this thing called equality.
TC1354	That’s where the Constitution defines, issho (“together”) with freedom, this thing called equality.
E9562	海に向かって、どちらの扉もいばいに開放されていた。
C9562	海に向かって、どちらの扉もいっぱいに開放されていたす。
TE9562	Facing the sea, both doors were open in ipai (“N/A”).
TC9562	Facing the sea, both doors were open in ippai (“full”).
E8936	彼は何事が起ったのかとびくりして跳ね起きた。
C8936	彼は何事が起ったのかとびっくりして跳ね起きた。
TE8936	Wondering what had happened, he jumped awake, bikuri (“N/A”).
TC8936	Wondering what had happened, he jumped awake, bikkuri (“surprised”).

Table A.1: Sample data synthesized using a syntactic character rule representing the deletion of the character っ within a noun. This is the same rule as the one displayed in Figure 5-8. っ is used to represent a doubled consonant sound, and is represented as such in the English translations.

Label	Sentence
E59864	かみさんは顔がゆがんで醜いが、率直でいいのところがあるらしい。
C59864	かみさんは顔がゆがんで醜いが、率直でいいところがあるらしい。
TE59864	Kami-san's face may be distorted and ugly, but he's seems to be frank and have good NO points.
TC59864	Kami-san's face may be distorted and ugly, but he's seems to be frank and have good points.
E232	この歳になって、ちょっと厳しいのことを言われたくらいで傷つくような柔なハートを持ってはいないはずだ。
C232	この歳になって、ちょっと厳しいことを言われたくらいで傷つくような柔なハートを持ってはいないはずだ。
TE232	At this age, you shouldn't have so soft a heart that you would be hurt if you were told some tough NO words
TC232	At this age, you shouldn't have so soft a heart that you would be hurt if you were told some tough words
E59663	全力で勝ちに行かなければ、いいの勝負すらできないだろう。
C59663	全力で勝ちに行かなければ、いい勝負すらできないだろう。
TE59663	Unless you try to win with all your strength, it won't be a good NO contest.
TC59663	Unless you try to win with all your strength, it won't be a good contest.
E44238	新しいの団地も点在していた。
C44238	新しい団地も点在していた。
TE44238	New NO apartment complexes were also scattered about.
TC44238	New apartment complexes were also scattered about.
E11240	かういふ家が画の正面の大部分を占めて居つて、その家は低いの石垣の上に建てられて居る。
C11240	かういふ家が画の正面の大部分を占めて居つて、その家は低い石垣の上に建てられて居る。
TE11240	The Kauifu house occupies the majority of the front of the picture, and the house is built on a low NO stone wall
TC11240	The Kauifu house occupies the majority of the front of the picture, and the house is built on a low stone wall

Table A.2: Sample data synthesized using a syntactic rule representing the insertion of the character の between an adjective and the noun it modifies. の is a possession marker and is translated into its romanji equivalent "NO" in the table.

Label	Sentence
E243	向こうから声をかけてきて、その後キャンパスに会うたびに、言葉を交わすようになった。
C243	向こうから声をかけてきて、その後キャンパスで会うたびに、言葉を交わすようになった。
TE243	After she greeted me、we began to exchange words each time we met at campus.
TC243	After she greeted me、we began to exchange words each time we met on campus.
E3801	そうです、しかし、とび上ったり、こわい目付きににらみつけたりしないで下さい。
C3801	そうです、しかし、とび上ったり、こわい目付きでにらみつけたりしないで下さい。
TE3801	That's right. But, please don't jump up or glare at me at that scary expression.
TC3801	That's right. But, please don't jump up or glare at me with that scary expression.
E214	すると夫人はポーランド語に喋って説明してくれた。
C214	すると夫人はポーランド語で喋って説明してくれた。
TE214	Thereupon, my wife spoke at Polish and explained it.
TC214	Thereupon, my wife spoke in Polish and explained it.
E12394	初めはさびしさに泣いたが、吟子はだからといって甘やかすことはなかった。
C12394	初めはさびしさで泣いたが、吟子はだからといって甘やかすことはなかった。
TE12394	At first, I cried at loneliness; nevertheless, Ginko did not pamper me.
TC12394	At first, I cried from loneliness; nevertheless, Ginko did not pamper me.
E8945	あきれたような口調に言い、三太は全身に力を込めた。
C8945	あきれたような口調で言い、三太は全身に力を込めた。
TE8945	Speaking at an astonished tone, Santa poured his strength into his whole body.
TC8945	Speaking in an astonished tone, Santa poured his strength into his whole body.

Table A.3: Sample data synthesized using a syntactic rule representing the substitution of the particle で with the particle に between a noun and a verb. In general, で is used to mark the means or location of an action, while に is used to show the location of an object or indicate an object's motion.

Label	Sentence
E282	写真の花嫁は、いま早百合の目の前にある女らしい。
C282	写真の花嫁は、いま早百合の目の前にいる女らしい。
TE282	Apparently, the bride in the photo is the woman in front of Sayuri.
TC282	Apparently, the bride in the photo is the woman in front of Sayuri.
E6895	テレザとルイジは山の頂に着き、と同時に、二十人ばかりの山賊が目の前にありました。
C6895	テレザとルイジは山の頂に着き、と同時に、二十人ばかりの山賊が目の前にいました。
TE6895	At the same time Theresa and Luigi arrived at the summit of the mountain, there were about twenty bandits right in front of them.
TC6895	At the same time Theresa and Luigi arrived at the summit of the mountain, there were about twenty bandits right in front of them.
E7568	いをとなえる者などあろうはずもない。
C7568	いをとなえる者などいようはずもない。
TE7568	It's impossible for a dissenting voice to exist.
TC7568	It's impossible for a dissenting voice to exist.
E4090	うちの家内は今何処にあるのでしょう。
C4090	うちの家内は今何処にいるのでしょう。
TE4090	I wonder where my wife is now.
TC4090	I wonder where my wife is now.
E7394	もし弱い者がいれば、その人が悪いのではなく、大将が励まさない罪による。
C7394	もし弱い者がいれば、その人が悪いのではなく、大将が励まさない罪による。
TE7394	If weak men exist, it is not their fault, but the fault of their general failing to encourage them.
TC7394	If weak men exist, it is not their fault, but the fault of their general failing to encourage them.

Table A.4: Sample data synthesized using a syntactic rule representing the substitution of the "to be" verb いる, which is used for animate subjects, with the related "to be" verb ある, which is used for inanimate subjects. This is the same rule as the one displayed in Figure 5-6.

Appendix B

Sample Model Results

In this appendix, show sample output for our trained models on the teacher evaluation corpus. Each of the four tables below is data sampled from a subset of the teacher evaluation set corresponding to one of the 400 syntactic or character syntactic rules. The sentences of each table are labeled as $E\{n\}$, $C\{n\}$, $S\{n\}$, $L\{n\}$, $M\{n\}$, or $T\{n\}$. The integer value of n indicates that this is the nth unique error for the rule of the table. The abbreviations E, C, S, L, M, and T indicate the following:

- E - Unique error sentence
- C - Reference correction(s)
- S - Output of model trained on synthetic training corpus
- L - Output of model trained on Lang-8 training corpus
- M - Output of model trained on merged training corpus
- T - English translation of reference correction(s)

The annotated error and correction phrases are denoted with red and green text, respectively. The aligned model output to each error phrase is given in cyan text. For error sentences with multiple reference corrections, each correction and corresponding translation is labeled with number(s) indicating order.

Label	Sentence
E2295	ゲームがしてもいいですか。
C2295	ゲームをしてもいいですか。
S2295	ゲームがしてもいいですか。
L2295	ゲームをしてもいいですか。
M2295	ゲームをしてもいいですか。
T2295	May I play games?
E1739	冬だから、アイスクリームが食べたくじゃありません。
C1739	冬だから、アイスクリームを食べたくありません。(1) 冬だから、アイスクリームが食べたくありません。(2)
S1739	冬だから、アイスクリームを食べたくじゃありません。
L1739	冬だから、アイスクリームは食べたくありません。
M1739	冬だから、アイスクリームが食べたくありません。
T1739	Since it's winter, I don't want to eat ice cream. (1, 2)
E1453	車の窓から富士山が見ます。
C1453	車の窓から富士山を見ます。(1) 車の窓から富士山が見えます。(2)
S1453	車の窓から富士山が見えます。
L1453	車の窓から富士山が見えます。
M1453	車の窓から富士山を見ます。
T1453	I can see Mt. Fuji from the car window. (1) I see Mt. Fuji from the car window. (2)
E3517	電気が消します。
C3517	電気を消します。(1) 電気が消えます。(2) 電気が消してあります。(3)
S3517	電気を消します。
L3517	電気が消します。
M3517	電気が消えます。
T3517	I turned off the lights. (1) The lights went out. (2) The lights are turned off. (3)

Table B.1: Sample model outputs on the subset of the teacher evaluation set represented by the syntactic rule given as Rule 1 in Table 8.3. This rule represents the error in which the accusative marker を following a noun and preceding a verb is replaced by the nominative marker が.

Label	Sentence
E3169	4 時に映画館に会う予定があっても、町田さんはありませんでした。
C3169	4 時に映画館に会う予定があっても、町田さんはいませんでした。(1) 4 時に映画館に会う予定があっても、町田さんはいませんでした。(2)
S3169	4 時に映画館に会う予定があっても、町田さんはいませんでした。
L3169	4 時に映画館に会う予定があっても、町田さんはありませんでした。
M3169	4 時に映画館に会う予定があっても、町田さんはいませんでした。
T3169	Though we had planned to meet at the movie theater at 4 o'clock, Tanaka-san was not there. (1) Though we had planned to meet at the movie theater at 4 o'clock, Tanaka-san did not come. (2)
E27	五人家族があります。
C27	五人家族がいます。(1) 家族が五人います。(2)
S27	五人家族がいます。
L27	五人家族がいます。
M27	五人家族がいます。
T27	I have a five-person family. (1) My family has five people. (2)
E3331	教室に行きましたが、先生がありませんでした。
C3331	教室に行きましたが、先生がいませんでした。
S3331	教室に行きましたが、先生がありませんでした。
L3331	教室に行きましたが、先生がいませんでした。
M3331	教室に行きましたが、先生がいませんでした。
T2831	I went to the classroom, but the teacher was not there.
E1453	田中さんはデパートにありますか。
C1453	田中さんはデパートにいますか。
S1453	田中さんはデパートにあるか。
L1453	田中さんはデパートにありますか。
M1453	田中さんはデパートにありますか。
T1453	Is Tanaka-san at the department store?

Table B.2: Sample model outputs on the subset of the teacher evaluation set represented by the syntactic rule given as Rule 5 in Table 8.3. This rule represents the error in which the "to be" verb いる for animate subjects is replaced by the other "to be verb" ある, which is used for inanimate subjects.

Label	Sentence
E2805	ちょっと急ぎでください。
C2805	ちょっと急いでください。
S2805	ちょっと急いでください。
L2805	ちょっと急いでください。
M2805	ちょっと急いでください。
T2805	Please wait a moment.
E582	スーパーへ行いて、やさいをたくさん買いました。
C582	スーパーへ行って、やさいをたくさん買いました。
S582	スーパーへ行って、やさいをたくさん買いました。
L582	スーパーへ行き、やさいをたくさん買いました。
M582	スーパーへ行って、やさいをたくさん買いました。
T582	I went to the store and bought lots of vegetables.
E1583	この箸を使いてください。
C1583	この箸を使ってください。
S1583	この箸を使ってください。
L1583	この箸を使ってください。
M1583	この箸を使ってください。
T1583	Please use these chopsticks.
E203	韓国でお土産を買いて、日本に帰りました。
C203	韓国でお土産を買って、日本に帰りました。
S203	韓国でお土産を買って、日本に帰りました。
L203	韓国でお土産を買い、日本に帰りました。
M203	韓国でお土産を買って、日本に帰りました。
T203	I bought souvenirs in Korea and returned to Japan.
E1359	家に帰りて、ご飯を食べました。
C1359	家に帰って、ご飯を食べました。
S1359	家に帰って、ご飯を食べました。
L1359	家に帰って、ご飯を食べました。
M1359	家に帰って、ご飯を食べました。
T1359	I returned home and ate food.

Table B.3: Sample model outputs on the subset of the teacher evaluation set represented by the syntactic rule given as Rule 6 in Table 8.3. This rule represents a conjugation error in which the te-connecting form of a verb is replaced by the verb stem.

Label	Sentence
E1571	綺麗写真を撮りました。
C1571	綺麗な写真を撮りました。
S1571	綺麗な写真を撮りました。
L1571	綺麗な写真を撮りました。
M1571	綺麗な写真を撮りました。
T1571	[I] took a pretty picture.
E4269	便利コンビニにおにぎりを買いたいです。
C4269	便利なコンビニでおにぎりを買いたいです。(1) コンビニでおにぎりを買いたいです。(2)
S4269	便利なコンビニにおにぎりを買いたいです。
L4269	便利コンビニでおにぎりを買いたいです。
M4269	便利コンビニでおにぎりを買いたいです。
T4269	[I] want to buy onigiri at the convenient convenience store. (1) [I] want to buy onigiri at the convenience store. (2)
E671	友だちは綺麗海の家があります。
C671	友だちは綺麗な海の家があります。(1) 友だちは綺麗な海の家を持っています。(2)
S671	友だちは綺麗な海の家があります。
L671	友だちは綺麗な人がいます。
M671	友だちは綺麗な海の家があります。
T671	My friend has a pretty beach hut. (1, 2)
E4000	この町には、親切人がたくさんいる。
C4000	この町には、親切な人がたくさんいる。
S4000	この町には、親切な人がたくさんいる。
L4000	この町には、親切な人がたくさんいる。
M4000	この町には、親切な人がたくさんいる。
T4000	In this town, there are many kind people.
E3107	この綺麗のください。
C3107	この綺麗なのをください。
S3107	この綺麗なください。
L3107	この綺麗のください。
M3107	この綺麗にしてください。
T3107	Please give [me] the pretty one.

Table B.4: Sample model outputs on the subset of the teacher evaluation set represented by the syntactic rule given as Rule 10 in Table 8.3. This rule represents the deletion error in which a copula particle な that exists between a 形容動詞 (literally "adjective verb", also known as a ～な adjective) and the noun it modifies is removed.

Bibliography

[1] Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy, August 2019. Association for Computational Linguistics.

[2] Yo Joong Choe, Jiyeon Ham, Kyubyong Park, and Yeoil Yoon. A neural grammatical error correction system built on better pre-training and sequential transfer learning. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 213–227, Florence, Italy, August 2019. Association for Computational Linguistics.

[3] Shamil Chollampatt and Hwee Tou Ng. Connecting the dots: Towards human-level grammatical error correction. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 327–333, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[4] Shamil Chollampatt and Hwee Tou Ng. A multilayer convolutional encoder-decoder neural network for grammatical error correction. *CoRR*, abs/1801.08831, 2018.

[5] Daniel Dahlmeier and Hwee Tou Ng. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572, Montréal, Canada, June 2012. Association for Computational Linguistics.

[6] Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[7] Mariano Felice, Zheng Yuan, Øistein E Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. Grammatical error correction using hybrid systems and type filtering. Association for Computational Linguistics, 2014.

[8] Tao Ge, Furu Wei, and Ming Zhou. Reaching human-level performance in automatic grammatical error correction: An empirical study. *arXiv preprint arXiv:1807.01270*, 2018.

[9] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122, 2017.

[10] Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, 2019.

[11] Na-Rae Han, Martin Chodorow, and Claudia Leacock. Detecting errors in english article usage with a maximum entropy classifier trained on a large, diverse corpus. In *LREC*, 2004.

[12] Kenji Imamura, Kuniko Saito, Kugatsu Sadamitsu, and Hitoshi Nishikawa. Particle error correction from small error data for japanese learners. *Information and Media Technologies*, 9(4):834–856, 2014.

[13] Marcin Junczys-Dowmunt and Roman Grundkiewicz. The amu system in the conll-2014 shared task: Grammatical error correction by data-intensive and feature-rich statistical machine translation. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 25–33, 2014.

[14] Marcin Junczys-Dowmunt and Roman Grundkiewicz. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. *CoRR*, abs/1605.06353, 2016.

[15] Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. Marian: Cost-effective high-quality neural machine translation in C++. *CoRR*, abs/1805.12096, 2018.

[16] Karen Kukich. Techniques for automatically correcting words in text. *Acm Computing Surveys (CSUR)*, 24(4):377–439, 1992.

[17] Jared Lichtarge, Christopher Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. Corpora generation for grammatical error correction. *CoRR*, abs/1904.05780, 2019.

[18] Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 147–155, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing.

[19] Daniel Naber et al. *A rule-based style and grammar checker*. Citeseer, 2003.

[20] Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593, Beijing, China, July 2015. Association for Computational Linguistics.

[21] Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[22] Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[23] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

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

[25] Marek Rei, Mariano Felice, Zheng Yuan, and Ted Briscoe. Artificial error generation with machine translation and syntactic patterns. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 287–292, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[26] Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, and Dan Roth. The university of Illinois system in the CoNLL-2013 shared task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 13–19, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[27] Alla Rozovskaya and Dan Roth. Generating confusion sets for context-sensitive error correction. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 961–970. Association for Computational Linguistics, 2010.

[28] Alla Rozovskaya and Dan Roth. Training paradigms for correcting errors in grammar and usage. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics*, pages 154–162. Association for Computational Linguistics, 2010.

[29] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *CoRR*, abs/1511.06709, 2015.

[30] Joel Tetreault and Martin Chodorow. The ups and downs of preposition error detection in esl writing. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 865–872, 2008.

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

[32] Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. Noising and denoising natural language: Diverse backtranslation for grammar correction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 619–628, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[33] Zheng Yuan, Felix Stahlberg, Marek Rei, Bill Byrne, and Helen Yannakoudakis. Neural and FST-based approaches to grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 228–239, Florence, Italy, August 2019. Association for Computational Linguistics.

[34] 藤本恭子, 水本智也, 井上一成, and 奥村紀之. 日本語作文の自動誤り訂正における統計的機械翻訳とニューラル機械翻訳の性能評価. In 言語処理学会第*25*回年次大会 発表論文集, pages 414–417. Association for Natural Language Processing, 2010.