Identifying negative language transfer in learner writing: using syntactic information to model structural differences

by

Leticia Farias Wanderley

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Second language learners transfer rules from their native languages when trying to communicate in the new language. When the transferred rules do not match the second language grammar, this reuse results in errors. Although this phenomenon is well known and documented by linguists and language teachers, few computational methods have been applied to detect it in learner writing. Without the automatic detection of language transfer, it is harder to provide feedback that makes learners aware of the phenomenon. In this thesis, I introduce a new method to identify when learner errors are related to the negative language transfer phenomenon. Along with the method description, this thesis contains the results of applying it to a dataset of errors made by Chinese native speakers who were learning English. These results show that my method achieves high precision scores in detecting negative language transfer errors in the writing of Chinese native speakers. These results can be applied in informing error feedback that explains the negative language transfer causes of an error. Providing this type of feedback should help learners reflect on the differences between language rules and refrain from reusing native language rules in their English writing.

Preface

The results from this thesis were originally published as:

Leticia Farias Wanderley and Carrie Demmans Epp. Identifying negative language transfer in learner errors using POS information. In Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications, pages 64–74, Online, April 2021. Association for Computational Linguistics.

Some of the ideas discussed in the future directions section originally appeared as part of the following publication:

Leticia Farias Wanderley and Carrie Demmans Epp. Identifying negative language transfer in writing to increase English as a Second Language learners' metalinguistic awareness. In Companion Proceedings 10th International Conference on Learning Analytics & Knowledge (LAK20), pages 722–725, March 2020.

All negative language transfer annotations were done by Nicole Zhao. The description of the annotation procedure in Appendix A was originally published as:

Leticia Farias Wanderley, Nicole Zhao, and Carrie Demmans Epp. Negative language transfer in learner English: A new dataset. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3129–3142, Online, June 2021. Association for Computational Linguistics.

Acknowledgements

Above all, this thesis would not have been possible without Carrie. Her support, guidance, and supervision made everything better. I will be forever grateful for her patience and help. All of my gratitude also goes to Nicole, whose work in annotating the negative language transfer dataset enabled the fulfilment of this thesis. Many times, Nicole's cheerfulness, positivity, and trust were all I needed to keep on working. I am thankful for my committee, who took the time to read this thesis with such care and provided feedback that made this work all the better. Finally, I would like to acknowledge the members of EdTeKLA for their never-ending encouragement and support.

Contents

List of Tables

List of Figures

Glossary

F1-score

Harmonic mean of precision and recall

FCE

First Certificate in English, an upper-intermediate level English language examination offered by Cambridge Assessment English

GEC

Grammatical error correction, natural language processing task in which the goal is to correct grammatical mistakes

L1

First language, an individual's mother tongue

L2

Second language, a foreign language different from the individual's mother tongue

Language modelling

Natural language processing technique that creates a probability distribution over sequences of tokens from a dataset

Negative language transfer

Second language acquisition phenomenon in which learners incorrectly reuse rules from their native languages when communicating in the second language

Part-of-speech (POS)

Grammatical word category

Precision

Ratio of relevant entries retrieved to total retrieved entries

Recall

Ratio of relevant entries retrieved to total relevant entries

RNN

Recurrent neural network, neural network architecture in which each output unit value is computed based on current input and previous output values

Shallow syntactic language model

Language model trained with part of speech tag sequences extracted from data in a particular language

UD

Universal Dependencies, a grammar annotation framework that is shared across several languages

Chapter 1

Introduction

Oftentimes knowing how to communicate in English can mean better opportunities, be it studying abroad, attending graduate school, getting a better job, or being promoted at work. Natural language processing research has developed several systems that help individuals learn languages and communicate better. The output from grammatical error correction systems, for example, can help learners prevent errors and improve their understanding of the language they are learning (Monaikul and Di Eugenio, 2020). These systems have been used to correct learner writing and to inform error feedback. They are meant to provide direct corrective feedback to the learners' incorrect utterances (Nadejde and Tetreault, 2019).

Direct corrective feedback highlights the learners' errors and suggests a corrective action to address them (Bacquet, 2019). Education and language learning researchers have found that having access to this type of feedback is helpful to learners, as it not only supports the error's correction but also prevents that mistake from being made again (Sheen, 2007). However, research in this area has found that direct corrective feedback is not the only type of feedback that supports learning. Feedback that encourages learners to think about error causes has been shown to improve writing accuracy. This feedback type, called metalinguistic feedback, provides learners with information that helps them understand error causes and increase their language knowledge (Karim and Nassaji, 2020). Through metalinguistic error feedback, learners are presented with the language rules and an explanation of how their errors

violated these patterns.

One metalinguistic dimension that often causes problems for language learners is the occurrence of language transfer. Language transfer is a second language acquisition phenomenon in which learners reuse patterns from their native languages when communicating in a second language[1] (Lado, 1957). This rule reuse, characterized as language transfer, can be harmless. However, it frequently happens when there is a divergence between native and second language rules. When this divergence occurs, language learners create utterances that are invalid in the second language and, thus, make a mistake. This type of mistake is called negative language transfer, and it is one of the most common causes of learner errors in second language acquisition (Bardovi-Harlig and Sprouse, 2017). Negative language transfer is a well known phenomenon in second language acquisition research. It also has been widely reported by language teachers in teaching guides, such as the one written by Swan and Smith (2001). Although Monaikul and Di Eugenio (2020) discussed providing contrastive feedback for missing preposition errors that are related to the negative language transfer phenomenon, I am not aware of any previous work on detecting this phenomenon for a broader range of errors.

In this thesis project, I set out to investigate whether it is possible to detect when Chinese native speakers who are learning English make errors that are related to negative language transfer. More specifically, I examined erroneous syntactic structures that were more similar to the learners' first language (i.e., Chinese) than to English and whether computational models that represent language structures could be used to correctly identify them. To demonstrate my method's feasibility, I analysed the performance of traditional and artificial neural network language structure representations in detecting negative language transfer[2]. The results demonstrate that artificial neural network methods outperform traditional ones in this task, with the best performing model yielding an F1-score of 0.51 in detecting negative language transfer on

[1]By second language, I mean any additional language beyond the native tongue.

[2]The data and code used in this analysis are available at `https://github.com/EdTeKLA/LanguageTransfer`

English essays written by Chinese native speakers.

To ground my work, I discuss the limitations and potential applications of my results. I hope the negative language transfer detection results can be used to inform error feedback with more than just direct corrective feedback. Information about the negative language transfer dimension of errors can enhance error feedback with metalinguistic cues. These cues would highlight how the rules in the first language and English diverge, which could support learners in developing a more accurate understanding of English. By contrasting their native languages and English, language learners will be able to reflect on each languages' rules and develop a better representation of both.

The choice to use errors made by Chinese native speakers to evaluate my method comes from Chinese being one of the most common first languages in the learner dataset analysed. Beyond that, the differences in language representation and grammar between English and Chinese are reflected in the number of structural errors found in learner writing. Finally, within my research group I had the opportunity to work with a negative language transfer annotator, who enhanced error annotated essays with information about the negative language transfer phenomenon.

The work presented in this thesis is the first step in the pursuit of providing negative language transfer informed feedback to English language learners. It investigates whether it is possible to detect structural negative language transfer by representing errors as sequences of word categories. This work examines the use of word category sequences to represent learner errors as well as first and second language syntax. Moreover, it models first and second language patterns and compares the word category sequences extracted from learner errors to those patterns. The models trained to represent the languages are tasked with detecting traces of the learners' first language in their writing. This detection output could be used to enhance learner error feedback in the future.

This thesis is organized as follows. Chapter 2 introduces background concepts and research with the goal of helping the reader understand the methods applied. Chapter 3 discusses research in natural language processing and sec-

ond language acquisition that inspired and grounded this thesis. In chapter 4, I describe the datasets used to validate the method proposed, along with the data preprocessing techniques applied to those datasets. Chapters 5 and 6 describe, respectively, the baseline and artificial neural network methodologies used in the negative language transfer detection task. In chapter 7, I provide an in-depth discussion of the results obtained and discuss the limitations, implications, and future research directions that stem from this work. Finally, chapter 8 concludes this manuscript with a summary of the work performed and results achieved in this research.

Chapter 2

Background

In this chapter, I discuss the background research that will help the reader understand this thesis' methodology and motivation. First, I introduce the linguistic theories that explain the language transfer phenomenon. Then, I describe part-of-speech tagging and language modelling, the natural language processing techniques applied in the computational representation of language structures.

2.1 Language transfer

Second language acquisition research has identified several strategies used by learners to acquire a language. Repetition, memorization, and translation are a few examples of learning strategies used by second language learners (O'Malley and Chamot, 1990). Among these learning strategies, there is the language transfer phenomenon. This phenomenon is characterized by learners reusing patterns from their first languages (L1s) when communicating in a second one (L2) (Lado, 1957).

Sometimes learners will transfer language rules intentionally, when they know the languages share a grammatical pattern or when they are not familiar with a particular structure in the L2. In other situations, learners are not aware they are transferring structures from their L1. That is, the language transfer phenomenon occurs unintentionally, due to learners resorting to known L1 structures while communicating in the L2. Although learners are not always aware they are transferring L1 rules into L2 utterances, this phenomenon has

been consistently reported by second language acquisition research (Lipka, 2020), as well as language teachers (Swan and Smith, 2001).

A second language acquisition theory that helps explain the language transfer phenomenon is the Interlanguage theory. It posits that second language learners maintain a dynamic psychological representation of the L2 that evolves as they acquire more knowledge of that new language (Selinker, 1972). The more primitive these representations are the more learners borrow from their L1s. According to Interlanguage theory, the contrast between L1 and L2 rules helps learners develop their L2 understanding and update their interlanguages towards the L2 grammar. Using this conceptualization, the language transfer phenomenon could be defined as learners applying developing areas of their interlanguage to create L2 utterances. These incomplete areas, that still mirror L1 rules and patterns, could be responsible for what is transferred (Bardovi-Harlig and Sprouse, 2017).

Several methods have been applied to investigate the effects of a developing interlanguage in learner writing. One of them is visualizing learner errors and their features. Visualization tools enable the comparison of different learner dimensions, such as L1s and proficiency levels. For example, the H-matrix tool described in Shimabukuro et al. (2019) uses matrices and tree diagrams to display learner error frequencies according to the learners' L1s. This tool groups learner mistakes by error category and displays their frequencies according to their occurrence in essays written by learners with different L1s. It allows users to analyse which error categories are more or less common depending on the learners' L1s. The circular heatmap and ranked visualizations described in Shimabukuro and Collins (2019) can help learners prevent acceptability errors in English. That is, it may help learners to avoid applying certain collocations in English that, although grammatically correct, sound or look odd to native English speakers. This visualization highlights words that are frequently used in the learners' L1, Portuguese, where the translations are rarely used in English (Shimabukuro and Collins, 2019).

To better research and act upon language transfer, the phenomenon is subdivided into two different categories. The first one, named positive language

transfer, refers to situations in which the L1 and L2 share a language pattern. Hence, when learners transfer that L1 pattern, the resulting L2 utterance is valid. However, when L1 and L2 rules diverge, reusing L1 patterns results in an L2 error. This second category of language transfer is called negative language transfer. The occurrence of negative language transfer in learners' utterances results in errors due to the transferred L1 rule not matching the L2 rules. These errors arise from learners not being familiar with the differences between languages. To illustrate the negative language transfer phenomenon, I turn to an exemplar sentence written by a Chinese native speaker:

"The idea of International Art Festival was great."

In this sentence, the noun phrase "International Art Festival" should be preceded by a determiner, i.e., "the" or "an". However, the language learner did not use any determiners before that noun phrase. This determiner omission error may have occurred due to the fact that there are no determiner equivalents in Chinese (Robertson, 2000). It is possible that when writing this sentence, the language learner applied Chinese rules in their English writing. The difference in Chinese and English determiner usage rules means that the transferred pattern resulted in an error.

As this example illustrates, negative language transfer happens when learners incorrectly apply structures from the L1 in the L2. Some of these incorrectly applied structures can be represented by sequences of word categories that are uncommon or invalid in the L2. To support my negative language transfer detection methodology, language structures are represented using sequences of part-of-speech tags. These sequences are used to differentiate between syntactic patterns found in the L1 and L2. The natural language processing tasks used to obtain these syntactic language representations are described in the following section.

2.2 Part-of-speech tagging

Part-of-speech (POS) tagging is a natural language processing task in which computational systems assign word categories to tokens in a text (Jurafsky

and Martin, 2009). These word categories, or part-of-speech tags, define the syntactic function that tokens perform within the text. POS tagging systems are trained to predict a word's part-of-speech based on the word itself and its neighbouring context. The set of parts-of-speech used to tag a piece of text may vary depending on the text's language and on the level of granularity desired.

2.2.1 Part-of-speech tagsets

Different sets of POS tags serve different purposes. They mainly vary according to which language characteristics are represented by the tags. Furthermore, distinct languages may need different numbers and types of tags to comprehensively represent their word categories. For example, unlike English, Chinese does not have determiners, which makes the determiner POS tag unnecessary in a Chinese POS tagset.

A very popular tagset for English is the Penn Treebank tagset (Marcus et al., 1993). This tagset contains 48 tags that represent parts-of-speech (36 tags) and other writing symbols (12 tags), such as punctuation marks. Each of the 36 POS tags in the Penn Treebank dataset holds information about distinct syntactic characteristics, such as number, tense, and inflection, along with the main word categorization (e.g., verb, noun, adverb). However, the syntactic characteristics described by the tags are only reflective of properties found in the English language. That is, the Penn Treebank tagset specializes in representing English syntactic structures. Although this specialization supports more detailed tagging, it does not fit my thesis' purpose as my goal is to directly compare language structures. My methodology requires a shared tagset, a tagset that can be used to tag texts written in different languages with the same part-of-speech tags.

The Universal Dependencies (UD) project is a cross-linguistic effort to create and maintain a shared treebank annotation scheme (Nivre et al., 2016). This treebank describes syntactic features that are shared among several languages and defines an annotation scheme to support cross-lingual syntactic annotation. There are 17 different POS tags in the UD tagset, and they repre-

sent common word categories across languages. The UD project was created with the objective of supporting comparative analysis among languages (Nivre et al., 2016). It enables syntactic comparison across languages, as different languages can be annotated using a shared scheme. The UD tagset fits my methodology as it provides a POS tagging scheme that is common to English and Chinese. This means that it is possible to POS tag texts in English and Chinese using the same tags and directly compare these languages' structures.

A sequence of POS tags can be used to represent a language structure, as the ordered POS tags extracted from a sentence reflect a language pattern. Hence, POS tag sequences extracted from a large set of sentences in a given language can serve as a proxy to that language's grammatical structures. To enable the representation of language structures and their application in negative language transfer detection, I used POS tag sequences to train shallow syntactic language models. The following chapter discusses the classes of language modelling techniques used in my thesis.

2.3 Language modelling

Language models are computational systems that assign probabilities to sequences of tokens (Jurafsky and Martin, 2009). These models are trained to recognize token sequences that belong to a language, and upon seeing a test sequence, compute the likelihood of that sequence belonging to the language they represent. Hence, language modelling is a task in which training data is used to create a model that represents its language and is able to identify other sequences that likely belong to that language.

Recent advances in artificial neural network architectures have achieved high performance in natural language processing tasks, such as language modelling. The state of the art methodology for language modelling is the transformer approach (Alikaniotis and Raheja, 2019). These novel models forgo the use of recurrent architectures and apply self-attention to train artificial neural network units and learn latent language patterns (Vaswani et al., 2017). Transformer models allow for the processing of an entire sequence at one sin-

gle time step, as opposed to recurrent approaches, which process one token at a time and tend to lose important information from distant points in the sequence (Vaswani et al., 2017). Transformers are able to model languages through a series of self-attention layers that process the input sequence in parallel. The ability to parallelize the computation and to take in more information from previous tokens, makes the transformer architecture a powerful language modelling approach (Jurafsky and Martin, 2020).

Although transformer models have achieved impressive results in natural language processing tasks, this architecture was not explored in my thesis. Most of the power in such models comes from the existence of pre-trained models. These models are pre-trained on large amounts of data and can be fine-tuned to perform a specific task (Devlin et al., 2019; Radford et al., 2018; Peters et al., 2018). The task proposed in this thesis focuses on modelling the structures of languages using part-of-speech sequences extracted from textual data. I am not aware of any pre-trained transformer model that was trained using this specific syntactic representation. Furthermore, my objective is to explore the task's feasibility, as well as its applicability in educational settings. The cost of training and maintaining a large-scale transformer model may become prohibitive in such environments.

The following subsections introduce the language modelling techniques used in my thesis. With them, I hope to provide the background knowledge to support the understanding of the methodology described in chapters 5 and 6.

2.3.1 N-gram models

N-gram models are a statistical natural language processing approach to language modelling. These models work by deriving a token sequence distribution from the training data and observing how well a test sequence fits that distribution (Jurafsky and Martin, 2009). More specifically, n-gram models compute the probability of a token based on the tokens that precede it. That is, the likelihood of seeing a token at position i depends on the tokens at positions 0 to i - 1.

N-gram models apply the Markov assumption to condition a token's likelihood on closer previous tokens instead of on all its preceding tokens. For example, in a bigram model (n = 2) the probability of a bigram ($w_{i-1}w_i$) is conditioned on the unigram (w_{i-1}) probability. In this manner, the probability of a sentence can be computed by multiplying the probability of each bigram in the sentence (equation 2.1).

$$P(S) \approx \prod_{i=1}^{n} P(w_i|w_{i-1}) \tag{2.1}$$

The length of the sequences analysed by an n-gram model is one of the model's parameters. For example, n-gram models that analyse the distribution of one-word sequences are called unigram models, while models that compute the distribution of two-word sequences are called bigram models. The simplest way to compute an n-gram probability is to compute its relative frequency. This value is calculated based on the number of times the n-gram in question occurs in the training data and the number of times its respective (n - 1)-gram occurs. See equation 2.2 for the bigram probability computation formula.

$$P(w_i|w_{i-1}) = \frac{C(w_{i-1}w_i)}{C(w_{i-1})} \tag{2.2}$$

Although straightforward, this frequency computation may lead to a few complications depending on the datasets used. There could be cases in which a token is present in the test data, but not in the training data. In such cases, a problem would arise due to the denominator in equation 2.2 being zero, as there are no occurrences of the token in the training data. The result of this computation would be an undefined number that would invalidate the entire sentence's probability. N-gram language models solve this problem by defining a vocabulary during training. This vocabulary is the set of tokens from the training data that occur at a high frequency. It contains the most frequent training data words. Training data words that occur at a lower frequency than the one defined in the vocabulary creation are replaced by an unknown token. Before testing, the test data tokens that are not present in this pre-defined vocabulary are replaced by the unknown token. This procedure prevents unseen

tokens from driving a sentence's probability to zero.

Another issue that arises in n-gram model computations is the occurrence of unseen n-grams. Unseen n-grams are token sequences that do not occur in the training data even though the tokens that compose it belong to the model's vocabulary. Unlike unseen tokens, unseen n-grams only drive the numerator on equation 2.2 to zero. To cope with this issue, it is common to apply smoothing techniques to the n-gram distribution. These techniques distribute a small portion of the probability weight among unseen but possible n-grams. The redistribution of probability weight prevents the occurrence of zero probabilities when test sequences contain unseen n-grams.

Note that, applying smoothing techniques to n-gram models that represent the POS tag sequences' distribution in a language implies assigning small probability weights to POS tag sequences that may be invalid. That is, by smoothing the POS tag n-grams distribution, POS tag sequences that did not occur in the training data are assigned a probability greater than zero. The reason why those POS tag sequences did not occur in the training data could be attributed to them representing ungrammatical structures or to those structures being grammatical but infrequent in the language. Although the application of these techniques creates a somewhat artificial distribution, it only assigns a small probability to unseen n-grams, maintaining the shape of the distribution and preventing the invalidation of entire test sequences.

2.3.2 Recurrent neural networks

Sequential data is characterized by data in which the order of the input tokens is meaningful. When learning from sequential data, information from previous tokens is often highly relevant for the current token's computation. Recurrent neural networks (RNNs) are artificial neural network architectures that maintain a representation of previous outputs when computing the current output value (Jurafsky and Martin, 2020). This recurrent architectural feature makes RNNs suitable for use with sequential data, as the network is able to take previous information into account while it processes current input values (Dyer et al., 2016).

Natural language data is inherently sequential. To understand the meaning of each word in a sentence, it is necessary to have information about the words that precede it. For this reason, RNN models can be successfully applied in language modelling.

RNN-based language models work by processing sentences word-by-word and keeping a representation of the previously seen words at each time step (Mikolov et al., 2010). It is possible to model languages using a simple recurrent neural network architecture, also known as an Elman network (Elman, 1990). Figure 2.1 illustrates a simple recurrent neural network architecture.

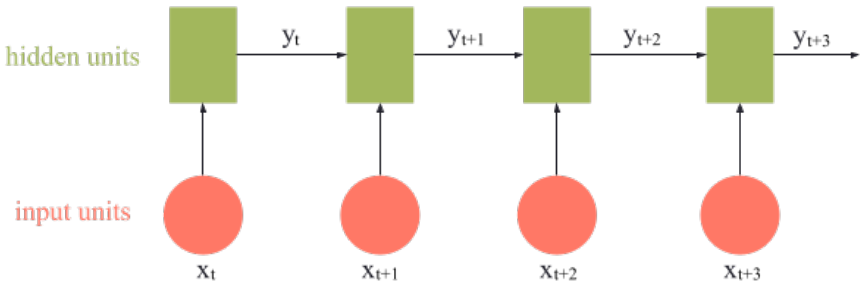

Figure 2.1: Unrolled simple recurrent neural network structure

Simple recurrent neural networks are composed of an input layer x, a hidden unit layer h, and an output layer y. Each element in each of the layers can be indexed by time step (e.g., y_t means the output layer at time t). Each word in the input sequence is processed in one time step t. The input layer unit at time t is the vector w_t, which is the vectorized representation of the word at position t (equation 2.3).

$$x_t = w_t \tag{2.3}$$

$$h_t = g(Wx_t + Uh_{t-1}) \tag{2.4}$$

$$y_t = f(Vh_t) \tag{2.5}$$

The weight matrices W and U in equation 2.4 are two of the three model parameters learnt during training. The current input vector and previous

hidden layer vector are multiplied by W and U, respectively. The results of these operations are added, and the activation function g in equation 2.4 determines the hidden layer activation value at the current time step (Jurafsky and Martin, 2020).

Finally, the weight matrix V is the last parameter learnt during training. It multiplies the final hidden layer vector to get to the network's output. In RNN-based language models, the function f in equation 2.5 is usually a softmax. The softmax function is used to normalize the scores from the $\mathrm{Vh_t}$ vector into a probability distribution over the training vocabulary (Jurafsky and Martin, 2020). During training, the loss is computed after each training iteration and used to update the model parameters (W, U, and V) via backpropagation.

The resulting vector $\mathrm{y_t}$ contains one probability value for each token in the training data vocabulary. To obtain the probability of the next token in a sequence, one needs to look at its respective entry in the output vector. The probability of a sentence, i.e., a sequence of words, can be computed as the product of the probability of each word in the sequence at its respective time step (see equation 2.6).

$$P(S) = \prod_{i=0}^{n} y_i^{wi} \tag{2.6}$$

The recurrent nature of RNN language models allows the model to maintain a representation of all the previously seen sequence tokens. Unlike n-gram language models, in which n determines the number of previous tokens that inform the computation, RNNs can take the entire previous sequence into account to compute the output vector. This property elucidates the superior performance of RNNs in the language modelling task.

Chapter 3

Related work

In this chapter, I discuss research areas that use non-native English learner data and language transfer information to improve computational and educational tasks. These tasks range from the automatic detection of authors' first languages to the provisioning of corrective language feedback for language learners. Related research shows that native language informed data is beneficial in both computational linguistics and second language acquisition settings and that this thesis' results can be integrated into other tasks to improve their performance and explainability.

3.1 Native language identification

Native language identification is a natural language processing task in which the objective is to predict the native language (L1) of a text's author. In the task's most common format, native language identification models take English text written by a non-native English speaker as input and output their prediction about the speaker's first language (Tetreault et al., 2013; Malmasi et al., 2017). During training, native language identification models learn to identify writing patterns that distinguish speakers of different first languages when writing in English.

A latent pattern that can be applied in native language identification is the use of cognates in the target language. Cognates are words that have similar forms and meanings in the L1 and L2. Rabinovich et al. (2018) showed that non-native English speakers tend to overuse English words that have cognates

in their L1s. Their experiments show that it is possible to reconstruct language family trees from the frequency with which certain cognates are used by non-native English speakers (Rabinovich et al., 2018).

Another writing pattern that can aid native language identification models in their task is the error type distribution in the author's writing. This is possible because some of the errors made by language learners are a result of transfer from their first languages, as shown by second language acquisition research (Bardovi-Harlig and Sprouse, 2017; Lipka, 2020). Flanagan et al. (2015) showed that using error distributions extracted from learner data outperformed a native language identification model trained on unbiased word vectors from the same dataset. Their results show that English error patterns can be used to identify the learners' first languages (Flanagan et al., 2015).

Some systems apply second language acquisition hypotheses to the native language identification task. One of those theories is the contrastive analysis hypothesis. It posits that challenging areas for language learners vary according to the linguistic distance between the L1 and L2 (Lado, 1957). This hypothesis proposes that the more distinct the two languages are the more difficult it is for learners to acquire the L2. It also posits that the more challenging structures for language learners to acquire are the ones in which the L1 and L2 diverge.

Wong and Dras (2009) examined the application of the contrastive analysis hypothesis by using information about three syntactic error types as features in native language identification models. The resulting models show that applying the contrastive analysis hypothesis between L1 and L2 supports native language identification tasks. The contrastive analysis hypothesis also inspired experiments that function in the opposite direction. Berzak et al. (2015) used the typological differences between the learners' L1s and English to predict error patterns in learner data. They applied contrastive analysis to discover challenging areas in English as a second language learning that are L1-specific. That is, they identified English patterns that are difficult to acquire and vary according to the learners' L1.

3.2 Grammatical error correction

Another natural language processing task that benefits from information about differences between L1 and L2 is grammatical error correction (GEC). In this task, the goal is to find and correct grammatical errors in learner data (Ng et al., 2013, 2014; Bryant et al., 2019). GEC systems learn to detect and correct errors from error-annotated learner data. State-of-the-art GEC systems model the task as a translation problem in which the erroneous data is the source language, and its corrected version the target language (Bryant et al., 2019).

Apart from large amounts of error-annotated data, GEC systems can take advantage of other types of information, such as the learners' native languages and proficiency levels in the target language. Rozovskaya and Roth (2011) showed that information about the learner's native language was useful to the GEC task by using distribution priors extracted from L1-specific errors to improve a Naïve Bayes GEC model. In their work, they improved preposition error correction by suggesting preposition replacements that frequently occurred in learner data from other learners with the same L1 (Rozovskaya and Roth, 2011).

L1-specific data can also improve the performance of artificial neural network GEC systems. Chollampatt et al. (2016) used L1-specific data to fine-tune a neural network joint model and then integrated it into a statistical machine translation GEC system. Their results showed that this L1-specific adaptation improved the GEC system's performance in correcting errors made by Spanish, Chinese, and Russian native speakers (Chollampatt et al., 2016).

More recently, Nadejde and Tetreault (2019) adapted general neural machine translation GEC systems to the learner's L1 and proficiency level in English. Their experiments analysed five proficiency levels and twelve different L1s. The results showed that fine-tuning the GEC systems to both properties improved the model's performance compared to the baseline. The best results were achieved when adapting to L1 and proficiency level at the same time (Nadejde and Tetreault, 2019). These results corroborate second language acquisition hypotheses, namely interlanguage and contrastive analy-

sis, that posit that learner error patterns change according to the learner's L1 and familiarity with the target language (Bardovi-Harlig and Sprouse, 2017).

Although experiments such as the ones described above show that information about the learner's L1 is beneficial for GEC performance, GEC research has not specifically focused on negative language transfer. Monaikul and Di Eugenio (2020) proposed providing negative language transfer informed feedback along with their preposition error correction results. As preposition errors are one of the most common errors in second language learning and are often related to negative language transfer, their work envisioned providing contrastive feedback for those error types (Monaikul and Di Eugenio, 2020). Similarly, I hope to use my models' results to enhance error feedback with information that contrasts the L1 and L2. Unlike the work described by Monaikul and Di Eugenio (2020), mine aims to detect structural negative language transfer by determining if learner errors are related to divergences between L1 and L2 rules.

3.3 Error feedback

Error feedback is an important area of research in second language acquisition. Although some researchers posit that no feedback should be given to language learners, most of them defend that some type of error feedback should be provided (Bacquet, 2019). The output from GEC systems can be used to provide direct corrective feedback to learners, as their output contains a suggested correction to the learner's errors. Direct corrective feedback such as this has been shown to be effective in helping learners acquire language knowledge (Liaqat et al., 2020). There are, however, other feedback types that support language learning, such as recasts, elicitation, and metalinguistic cues (Lyster and Ranta, 1997). In their written form, recasts are similar to direct corrective feedback, as they involve the instructors rewriting the learner's phrase without the error. Elicitation involves highlighting the learner's errors and encouraging them to correct it. Metalinguistic cues also highlight the learner's errors, but they do so by providing information about possible error causes (Lyster and

Ranta, 1997).

3.3.1 Metalinguistic feedback

Metalinguistic feedback invites learners to look back at their utterances and understand why they are incorrect (Lyster and Ranta, 1997). Its goal is to support learners in examining language as an object, analysing and internalizing its rules. In written language tasks, this goal is achieved through annotating learners' written work with observations on their language usage. For example, when an agreement error occurs, adding comments on why the words involved do not agree with one another.

Second language acquisition research has examined how this feedback type impacts language learning (Shintani and Ellis, 2013; Karim and Nassaji, 2020). Experimental results show that providing some type of error feedback improves writing performance when compared to not providing feedback. They also indicated that metalinguistic information improved learners' short-term writing accuracy (Karim and Nassaji, 2020). Additionally, language learners reported that the metalinguistic cues helped them become more aware of the target language rules (Shintani and Ellis, 2013).

Second language acquisition research examining the application of metalinguistic feedback to negative language transfer related errors has suggested that contrasting L1 and L2 rules in the error feedback supports the acquisition of accurate language structures (Tomasello and Herron, 1989; Kupferberg, 1999). Han (2001) discusses the application of fine-tuned error correction feedback. By their definition, fine-tuning means highlighting the causal factors of an error when providing feedback, bringing the learner's attention to these factors. The study's results show that acknowledging learners' perspectives about the feedback they have received when fine-tuning error feedback improved the learners' language awareness (Han, 2001).

To further understand the learners' perspective, Watts (2019) surveyed English language learners' awareness of the negative language transfer phenomenon. The study's participants reported that information about the phenomenon helped them prevent transfer related errors and improved their En-

glish writing accuracy. With my work, I hope to enable the automation of metalinguistic feedback that contrasts L1 and L2 patterns, as it has been shown that this contrast can help second language acquisition and metalinguistic awareness.

Chapter 4

Data

In this chapter, I provide an overview of the training and test datasets used in my thesis. Following the datasets overview, I describe the preprocessing procedures applied to extract structural information from parallel text in different languages and from learner data. These procedures were necessary as my goal was to compare language structures as opposed to word sequence distributions.

At this point, it is important to highlight the meaning of "language structure" in this manuscript. I modelled a language's structure as the distribution of part-of-speech tag sequences that are used in that language. Using these sequences to train language models resulted in a "shallow syntactic language model". That is, language models that represent language through part-of-speech tags, which are simple, yet pertinent, linguistic features.

4.1 Training data

Two datasets were used to train the models that represent language structures. As my thesis' goal is to compare erroneous written structures to the learners' L1 (Chinese) and English syntaxes, I needed to obtain textual data in both English and Chinese. The selected datasets contain parallel English and Chinese textual data aligned at the sentence level (Tiedemann, 2012). Parallel datasets were chosen to ensure that the Chinese and English sentences used to train the models were equivalent. These datasets are equivalent with regards to number of sample sentences and context.

Dataset	Number of sentences
Global Voices	138 582
WMT19	11 960
Combined	150 542

Table 4.1: Training datasets sizes

One of my data sources is the Global Voices corpus (Prokopidis et al., 2016). This dataset contains news stories published on the Global Voices website[1]. The news stories from this website discuss global news and events. They are written and translated by a team of collaborators and volunteers. The other data source is the news test set from the ACL 2019 Fourth Conference on Machine Translation (WMT19)[2]. This dataset contains parallel news stories selected from online sources. The selected news stories were translated specifically for the machine translation shared task (Barrault et al., 2019).

These datasets were chosen as training data because news stories usually contain simple and grammatical language structures. Language learners are encouraged to use patterns such as those in their writing. As a preprocessing step, sentences that did not have a parallel version in the other language were excluded from the training dataset. Table 4.1 presents the number of parallel sentences in each dataset, as well as the total number of training sequences.

4.2 Test data

To understand whether models that represent language structures were able detect negative language transfer on learner data, it was necessary to create a dataset that contained information about errors related to the language transfer phenomenon.

English learner data is readily available online due to well-known natural language processing tasks, such as grammatical error correction and native language identification. The datasets used in these tasks usually contain text written by non-native English speakers and, especially in the GEC corpora, these datasets are often annotated with learner error information. Error anno-

[1] `https://globalvoices.org/`

[2] `http://www.statmt.org/wmt19/translation-task.html`

tated datasets usually highlight and correct learner errors. In these datasets, errors and corrections are shared along with the learner text.

For my thesis, the learner dataset selected was the First Certificate in English (FCE) from the Cambridge Learner Corpus (Yannakoudakis et al., 2011). This dataset contains 1244 essays written by English as a second language learners while taking the FCE exam, an intermediate-level English test. The FCE dataset not only contains information about learner errors and their corresponding corrections, but also information about the learners' scores, age range, and native language. The learners whose essays are part of the FCE dataset have in total 16 different L1s. Each error in the dataset is also annotated with an error type, following the error coding described by Nicholls (2003). In the FCE datasets there are 66 essays written by Chinese native speakers. These essays have 30916 words in total and on average 468 words per essay ($SD = 101$).

Each essay in the FCE dataset is an Extensible Markup Language (XML) file, the errors and corrections are highlighted using XML tags. This mark-up enables the addition of extra features to a text file by encapsulating text with tags. Consequently, to correctly extract the erroneous utterances and POS tag them, preprocessing steps were necessary. First, the incorrect words and phrases were extracted from the enclosing XML formatted character sequences. This procedure aimed to make sure that the POS taggers did not get confused with the non-alphanumeric characters, such as ">", "<", and "=", that are habitually used in the XML format. After this extraction, plain text versions, i.e., text without XML features, of the learner essays were created. Table 4.2 provides examples of the FCE dataset's error mark-up and the error's respective plain text version.

In addition to that, the positional indices of the learner errors were calculated and stored for each error in the dataset. This procedure was envisioned to facilitate the localization of errors in the plain text versions of the learner essays. These indices were also useful to enable the precise extraction of the incorrect utterances and their neighboring words.

XML	Plain text
The idea of <NS type="MD"> <c>an</c></NS> International Art Festival was great.	The idea of International Art Festival was great.
I <NS type="TV"> had been<c>went</c> </NS> to your annual international art festival.	I had been to your annual international art festival.

Table 4.2: Examples of learner text before and after error extraction from XML format

4.2.1 Negative language transfer dataset

The L1 information available in the FCE dataset allowed the manual identification of negative language transfer related errors. As the essays could be grouped by their authors' L1s, it was possible to compare erroneous patterns from the essays with rules and structures from their authors' L1s. Then, if the grammatical structure of an error was incorrect in English, but resembled the learner's L1, that error would be flagged as negative language transfer related.

The errors from the FCE dataset were annotated with information about their relation with the language transfer phenomenon. This annotation enabled the evaluation of the shallow syntactic language models' performance in detecting negative language transfer. Each error in the negative language transfer annotated dataset contains a feature that indicates if the error is related to negative language transfer or not. The manual annotation process that allowed my models' evaluation was performed as part of this research. More information about the resulting annotated dataset can be found in Appendix A.

There are 3584 errors made by Chinese native speakers in the FCE dataset. Out of those, 1891 (52.76%) are tagged as negative language transfer and 1389 (38.76%) are tagged as non-transfer errors. The remaining 304 (8.48%) errors were left unlabelled for one of two reasons, they were spelling errors or the negative language transfer annotator was uncertain about its correct classification in the original dataset.

Most of the errors flagged as not related to negative language transfer were

Structural error type	Number of errors
Negative language transfer errors	1457
Not negative language transfer errors	914

Table 4.3: Distribution of structural negative language transfer errors in the test dataset

caused by learners applying English rules where they were not necessary. This phenomenon, called overcorrection, is characterized by learners attempting to conform to L2 grammatical rules by overusing them. Another common non-transfer error cause is connected to the learners' vocabularies. It is related to words being used out of context. As learners had yet to acquire the appropriate vocabulary, they would use words that did not fit their surroundings.

As my thesis' focus is to detect structural negative language transfer, I filtered the errors on the test set to only contain structural errors. I borrow some of the definition of structural errors from Berzak et al. (2015), filtering out spelling and word replacement errors from the test dataset. The resulting dataset contains 2371 structural error instances. Among those, 61.45% are related to negative language transfer and 38.55% are not. Table 4.3 presents the distribution of structural errors between these categories.

4.2.2 Test sequences

Each word unit in a sentence usually has some type of connection to other words in its surroundings. Hence, some learner errors in the FCE dataset are incorrect because of their relation to other words. Even though these other words may not be part of the incorrect structure, they contain relevant information about the learner error. For example, in the error "This remind me of what I experienced.", the incorrect verb inflection "remind" is only incorrect because its preceding word, "This", is a third person demonstrative pronoun.

Errors also vary with regard to length. Some errors, such as the one in the previous example, only contain one incorrect word, while others are composed of a set of words. An incorrect word order error, for example, always contains more than one word incorrectly ordered. With that in mind, I posit that it

Incorrect sentence	**Error type**	**Padded error span**	**Error + unigram span**	**Error + bigram span**
Moreover, the stars and artists were *only from* six countries.	Wrong word order	were *only from* six AUX ADV ADP NUM	*only from* six ADV ADP NUM	*only from* six countries ADV ADP NUM NOUN
This are only my immature views.	Pronoun agreement	*This* are DET AUX	*This* are DET AUX	*This* are only DET AUX ADV
This *remind* me of what I experienced.	Verb agreement	This *remind* me DET VERB PRON	*remind* me VERB PRON	*remind* me of VERB PRON ADP
I wanted somebody to share my *feeling.*	Wrong noun form	my *feeling.* PRON NOUN PUNCT	*feeling.* NOUN PUNCT	*feeling.* NOUN PUNCT

Table 4.4: Learner errors and test spans examples

is not enough just to look at the incorrect word sequence. Its surrounding context is also meaningful for the detection of negative language transfer.

To analyse how an error's context impacts the detection of negative language transfer, I decided to test three different POS tagged error spans. They are:

- The **padded error** span, which encompasses the POS tag extracted from the word that precedes the error, followed by the POS tags extracted from the error, followed by the POS tag extracted from the word that follows the error;
- The **error + unigram** span, which consists of the POS tags that belong to the error followed by the POS tag extracted from the subsequent word; and
- The **error + bigram** span, which is composed of the POS tagged error followed by the POS tags from its two subsequent words.

Note that the POS tags that belong to the incorrect words are always part of the test sequences. The different spans only define how much of the previous and subsequent context is added to the test POS tag sequences. Also note that context POS tags are only included in the test sequences if the word they are meant to represent occurs. That is, if an error occurs at the beginning of a sentence, its respective padded error sequence will not contain the leftmost context tag. In the same way, if an error occurs at the end of a sentence, its respective error + unigram and error + bigram sequences will not contain subsequent words' POS tags. Table 4.4 illustrates the different spans used in my thesis.

Although the definition of error spans helps the models to focus on learner misconceptions and their contexts, there is an error type that might not be well-represented by these spans' general definition. Omission errors are errors characterized by learners refraining from using certain words where they are necessary. For example, in the sentence "I hope she knows that [it] was her fault and won't do it again.", the word "it" was omitted by the learner although it was required. If represented by the error + unigram or error + bigram spans following their general definition, the context from missing, or omission, errors would not be fully captured. As an example, the error + unigram representation of the error described above would only contain the POS tag extracted from the word "was", which does not convey a lot about the error's context.

To better characterize missing errors, an adjustment was made to the error + unigram and error + bigram spans. When representing this type of error, the error + unigram span consisted of the POS tags extracted from the two words that followed the error, and the error + bigram span contained the POS tags extracted from three words after the error position. This adjustment allowed the model to better analyse missing errors' contexts. The missing error spans' adjustment also helped keep the amount of information represented by each error span consistent throughout the evaluation. As it can be seen in Table 4.5, the test sequences' lengths are fairly homogeneous within each span.

The manually-annotated errors from the FCE dataset allow for a focused

Span	Min	Q1	Median	Q3	Max
Error (no padding)	0	1	1	1	12
Padded error	1	3	3	3	14
Error + unigram	1	2	2	3	13
Error + bigram	1	3	3	4	14

Table 4.5: Length statistics of part-of-speech tagged test sequences

analysis of the learner errors. Using the error positions and types to extract the POS tags from the errors and their contexts created a condition that is hard to replicate. That is, grammatical error correction systems have achieved impressive results in detecting and correcting learner errors, however, their outputs are not always correct. When detecting negative language transfer on these systems' outputs, it will be necessary to design around possible system misclassifications.

4.3 Data preprocessing

In this project, language models that assign probabilities to part-of-speech tag sequences were used to represent a language's structure. These models were created to enable the comparison of a POS tag sequence's likelihood in both the L1 and L2 (Chinese and English). As the same POS tag sequence needed to be compared with L1 and L2 structures, both languages' structures were modelled using the same POS tagset, the Universal Dependencies tagset. This tagset was chosen because it contains a set of POS tags that are common and consistent among languages, including English and Chinese.

The POS tag sequences used to train the shallow syntactic language models were extracted from parallel data in English and in the learners' L1, Chinese. After collecting parallel data from different sources, each sentence in the datasets was part-of-speech tagged. The Chinese sentences were tagged by a POS tagger trained on Chinese data, and the equivalent sentences in English were tagged by an English POS tagger. Figure 4.1 illustrates the preprocessing procedure.

The POS taggers used to tag the training and test data come from the

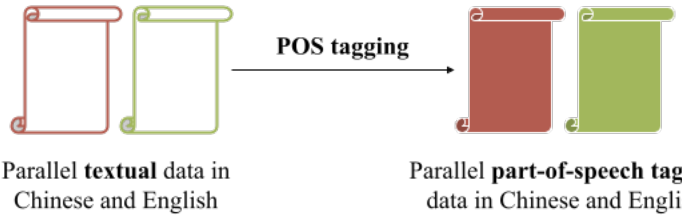

Figure 4.1: Illustration of the data preprocessing step

Python library spaCy version 2.3[3]. SpaCy is a natural language processing library equipped with pre-trained models in several different languages. I used spaCy's Chinese and English POS taggers to tag the parallel sentences written in Chinese and English, respectively. The English POS tagger was also used to tag the FCE test sentences, as they were written in English.

The POS tag sequences obtained through the process described above were used to train and test the language models used in negative language transfer detection. These shallow syntactic language models learnt to represent language structures and were applied in identifying Chinese patterns in the learners' English writing. The methodology used to train n-gram and RNN shallow syntactic language models and their results in the negative language transfer detection task are described in the two following chapters.

[3] `https://spacy.io/usage/v2-3`

Chapter 5

N-gram baseline approach

In this chapter, I describe the methodology and results from the n-gram negative language transfer detection approach. This approach uses n-gram models to represent Chinese and English language structures and detect negative language transfer in learner errors. It is the baseline approach to the task.

5.1 Methods

N-gram language models are probabilistic models that represent the distribution of token sequences in a language (Jurafsky and Martin, 2009). With n-gram language models, this baseline approach modelled the distribution of POS tag sequences extracted from textual data in English and Chinese.

Prior to training the n-gram models, the POS tag sequences in the training set were split into two sets according to the language they represent. This process resulted in two distinct training datasets, one for English and one for Chinese. Each training set was used to train an n-gram shallow syntactic language model to represent that language. Hence, two n-gram models were trained for the negative language transfer detection task. One of the models represented the structure of the learners' L1 (Chinese) and the second modelled English language structures. Each of these models was trained on its corresponding set of POS tag sequences. That is, the English model was trained on POS tag sequences extracted from English data, while the Chinese model was trained on the POS tag sequences extracted from the equivalent parallel data in Chinese. This procedure was devised to generate models with

analogous training sets, i.e., the training data had the same structural context and the same number of training sequences.

The n-gram models used in negative language transfer detection were trained using the Python interface for KenLM[1]. KenLM is an n-gram language model implementation that combines an advanced smoothing technique (modified Kneser-Ney smoothing) with optimized querying data structures (Heafield, 2011; Heafield et al., 2013).

The negative language transfer detection process consisted of computing the probability of POS tag sequences extracted from learner errors using both English and Chinese n-gram shallow syntactic language models. Each model outputted the likelihood of the POS tag sequence belonging to the language structure they represented. These likelihoods were compared to determine the error's relation to negative language transfer. If the likelihood assigned by the Chinese model was greater than the one assigned by the English model, the error was classified as negative language transfer (Figure 5.1).

Figure 5.1: N-gram baseline approach testing procedure

For each error in the test dataset, each POS tag sequence span (padded error, error + unigram, and error + bigram spans) was processed by both shallow syntactic language models and the resulting probabilities were compared. When a POS tag sequence extracted from an error was assigned a greater probability by the Chinese model, in comparison to the probability assigned by the English model, it was interpreted that the erroneous POS tag sequence

[1] `https://kheafield.com/code/kenlm/`

Learner error	**Chinese model output**	**English model output**	**Negative language transfer classification**
This *remind* me of what I experienced.	**-5.05**	-5.06	True
They flew out of the window and hit *directly* the headmaster's head.	-7.88	**-7.23**	False

Table 5.1: Negative language transfer classification results based on Chinese and English shallow syntactic language models' outputs

was more similar to Chinese language structures than to English ones. Hence, the learner error was classified as negative language transfer.

Table 5.1 contains examples of the baseline negative language transfer classification process and output. The negative language transfer classification output, derived from the models' probabilistic results, was compared to the negative language transfer annotation that accompanied the learner errors. The baseline approach's performance in detecting negative language transfer was calculated from this comparison's results.

5.2 Hyperparameter tuning

To select the parameter setting that best enabled the representation of language structures, the length of the POS tag sequences analysed by the n-gram models was tuned. The best performing n-gram length setting was used to train the models applied in detecting negative language transfer. Hence, the hyperparameter tuning step happened before the training and testing steps described in the previous section.

The n-gram length refers to the length of the POS tag sequences analysed by the model. Five different n-gram lengths were analysed, from 2 to 6. Before tuning, each monolingual training dataset was split into training and evaluation sets. The dataset that contained sequences extracted from English sentences was used to train English n-gram models and the one containing sequences extracted from Chinese sentences was used to train Chinese n-gram models.

The best parameter setting was selected according to the average source language prediction accuracy computed in a 5-fold cross-validation process with the training data. This process consisted of each monolingual training dataset being randomly split into 5 distinct folds. Then, on each iteration, models were trained using 4 of those folds and evaluated on the remaining one. This procedure was repeated for each n-gram length analysed. The best performing n-gram length was the one the achieved the highest average accuracy over all 5 iterations. Table 5.2 presents the average number of POS tag sequences on the splits used in hyperparameter tuning.

	Training split	**Evaluation split**
Chinese	120 433	30 109
English	120 433	30 109

Table 5.2: Number of sequences in the training and evaluation splits used for hyperparameter tuning

Each monolingual training split was used to train one English and one Chinese shallow syntactic language model for each hyperparameter combination. The models' performance was determined using both evaluation splits. The evaluation process consisted of computing the probabilities for each POS tag sequence in the evaluation split using the English and Chinese models trained with the same hyperparameter combination. Then, the models' outputs were compared and the sequence was labelled according to the model that assigned the highest likelihood, e.g., if the English shallow syntactic language model's output was greater than its Chinese counterpart, the sequence was classified as English. The labels assigned in this step were then compared to the sequences' source languages. The models' accuracy was computed based on these comparisons. The hyperparameter's performance was defined as the models' accuracy on the evaluation split.

The evaluation process described in the previous paragraph was repeated for each n-gram length analysed. The resulting evaluation accuracies were recorded and compared. Finally, the POS tag sequence length that yielded the best accuracy on the evaluation set was selected to train the models used in negative language transfer detection. The models that achieved the highest

Span	Precision	Recall	F1-score
Padded error	**0.68**	0.32	0.43
Error + unigram	0.64	**0.34**	**0.45**
Error + bigram	0.66	0.27	0.38

Table 5.3: N-gram baseline negative language transfer detection results

accuracy on the evaluation split had n = 5. That is, they analysed one sequence of five POS tags at a time. The best accuracy achieved in the tuning process was 96.97%. This result refers to the accuracy yielded by the n-gram models in predicting the source language of the POS tag sequences in the evaluation split. The evaluation split accuracy results for all n-gram lengths analysed can be found in Appendix B.

5.3 Results

The n-gram models trained to predict the source language of a POS tag sequence were then used to assign the negative language transfer label of learner errors. Table 5.3 contains the precision, recall, and F1-score results attained by the n-gram baseline in the negative language transfer detection task. The n-gram baseline achieved the highest recall and F1-score results at detecting negative language transfer related learner errors when the error + unigram span was used. The error + unigram span includes the POS tags extracted from the error along with the POS tag extracted from the word that comes immediately after the error. For example, the error + unigram sequence retrieved from a verb agreement error in the sentence "This remind me of what I experienced." contains the POS tags extracted from the words "remind me".

The error + unigram span, however, yielded the lowest precision scores in the n-gram baseline approach. The best performing span with regards to precision was the padded error span. This span encompasses the POS tags extracted from the words that surround the error along with the POS tags extracted from the error. These results indicate that both previous and following contexts influence the accurate detection of negative language transfer in this approach.

When compared to the error + unigram span, it is possible to perceive that the extra POS tag that is included by the error + bigram span made negative language transfer detection more precise. However, the error + bigram span yielded the lowest recall and F1-score results with the n-gram baseline. This added POS tag may have misled the models into classifying negative language transfer related errors as not transfer related, as the recall value for this error span was the lowest one among the error spans analysed. These results indicate that including information from words that are further away from the error may not be beneficial for negative language transfer detection in all cases.

For all error spans analysed, the precision scores were always higher than the recall scores. The low recall results show that the n-gram baseline performed poorly at identifying most of the negative language transfer related errors. However, their precision results indicate that this approach made more accurate predictions regarding those errors. That is, most of the errors classified as negative language transfer are indeed related to the negative language transfer phenomenon.

Overall, the low recall scores yielded by the n-gram baseline may indicate that the n-gram baseline methodology does not fully support the negative language transfer detection task. This methodology uses two independent shallow syntactic language models to detect negative language transfer. Each n-gram shallow syntactic language model was trained to represent the structures from one of the languages, and negative language transfer detection relied on a comparison between two independent results.

By design, one n-gram model is only able to represent one language. These models process the training data and derive a sequence distribution from it. This feature of n-gram models made it necessary to train one model to represent English structures and another one to represent Chinese structures. Each of the shallow syntactic language models trained does not recognize other languages' structures. Hence, for the n-gram baseline, a model's output is solely based on how similar the error structure is to the language structures it represents. The independence between the models' sequence distributions may have caused negative language transfer related errors to be misclassified as not

transfer related.

5.4 Error analysis by grammatical error type

The error coding used by the FCE annotators defines five general error categories. These categories are "missing", "replacement", "unnecessary", "wrong form used", and "wrongly derived" (Nicholls, 2003). These error categories are augmented with information about which part-of-speech is missing, unnecessary, used in the wrong form, wrongly derived, or needs replacing. For example, when an error is annotated with the "unnecessary adverb" code, it means that the learner used an adverb where it was not needed, e.g., the word "directly" in the sentence "They flew out of the window and hit directly the headmaster's head." was annotated as an unnecessary adverb error. The learner wrote "hit directly the headmaster's head" when they should have written "hit the headmaster's head".

By grouping the n-gram baseline negative language transfer predictions using the error coding described in Nicholls (2003), it is possible to notice some patterns in the results. The most common errors made by Chinese native speakers in the FCE dataset are punctuation replacement, wrong verb tense, and missing determiner errors (with 336, 267, and 209 instances in the dataset, respectively). Table 5.4 presents the precision, recall, and F1-score results achieved by the n-gram models when detecting negative language transfer for these error subtypes.

Note that the best scores for punctuation replacement and wrong verb tense errors are lower than the ones for missing determiner errors. This difference can be attributed to the POS tagset used to represent the language structures. The Universal Dependencies tagset uses the tag "PUNCT" to represent all punctuation marks. This means that the POS tag sequences extracted from punctuation replacement errors do not have additional information about the type of punctuation mark that was used incorrectly, e.g., it is not possible to distinguish a period from a comma using the error's POS tag sequence. There are differences between the usage of certain punctuation marks in English

Error type	Span	Precision	Recall	F1-score
Replace punctuation (n = 336)	Padded error	0.65	0.30	0.41
	Error + unigram	**0.69**	**0.43**	**0.53**
	Error + bigram	0.62	0.23	0.34
Wrong verb tense (n = 267)	Padded error	0.70	0.21	0.32
	Error + unigram	**0.78**	**0.36**	**0.49**
	Error + bigram	0.73	0.22	0.33
Missing determiner (n = 209)	Padded error	**1.00**	**0.43**	**0.60**
	Error + unigram	0.97	0.33	0.50
	Error + bigram	**1.00**	0.35	0.52

Table 5.4: N-gram models' results on most common error types (an extended version of this table, containing all error types, is available in Appendix D)

and Chinese (Liu, 2011) that may cause negative language transfer, but that difference is not represented by the n-gram models on account of a too general POS tag.

A similar issue occurs to the representation of verb tenses. The Universal Dependencies tagset uses two POS tags to represent verbs, "VERB" and "AUX". Although the POS tags used differentiate between auxiliary and main verbs, they do not provide any information about verb tenses. Hence, information about different verb tense usage patterns in each language is not modelled by the n-gram shallow syntactic language models and cannot be reliably used to detect negative language transfer in wrong verb tense errors.

The detection of negative language transfer in missing determiner errors, is not hindered by the POS tags used. These errors' structure is marked by the absence of a determiner before a noun phrase, and it is related to negative language transfer as there are no determiner equivalents in Chinese (Robertson, 2000). All the error spans used achieved high precision scores when detecting transfer in missing determiner errors. However, the padded error span yielded the highest recall score, meaning that the models were able to detect more negative language transfer related missing determiner errors when analysing this span.

The padded error span may have been the error span that best represented this error type because it contains the POS tags that surround the error. The English model did not recognize the error structure because in English

structures it is more common for a determiner to be between those POS tags. The Chinese model, on the other hand, assigns it a higher probability to the sequence as determiners are not used between any POS tags in Chinese.

Chapter 6

Recurrent neural network approach

One of the disadvantages of the n-gram baseline approach is that the model that represents the Chinese structures and the one that represents English structures are independent of one another. Even though they were trained on parallel data, the English n-gram shallow syntactic language model is never exposed to Chinese structures and vice versa. The probability value outputted by one of the models only represents the likelihood of a POS tag sequence belonging to the language structures represented by that model. It does not signify the probability of the sequence not belonging to the language represented by the parallel n-gram model.

To address this shortcoming, a recurrent neural network (RNN) architecture was selected to represent language structures. This language modelling architecture enables one single model to differentiate between two languages. This chapter reports on the usage of a recurrent neural network approach to negative language transfer detection. It describes the methodology applied and discusses the results achieved by a recurrent neural network shallow syntactic language model in the negative language detection task.

6.1 Methods

RNN-based language models are able to maintain a representation of preceding tokens when computing the current state of a sequence (Mikolov et al., 2010).

By doing this, these models can make inferences based on previous tokens as well as on the current one. This RNN architecture feature enables the modelling of sequential data, such as natural language.

To perform the negative language detection task, an RNN model was introduced to POS tag sequences extracted from both English and Chinese sentences during training. Throughout this process, it learnt to predict to which language a given POS tag sequence belongs. The RNN shallow syntactic language model analysed POS tag sequences extracted from sentences in both languages and created an internal representation that was able to differentiate between language structures.

The RNN's input units are vectorized Universal Dependencies POS tags. Each POS tag from the UD tagset is represented by a one-hot-encoding vector. As there are 17 tags in the UD tagset, each input vector is 17 units long, and each of those 17 vector positions is equivalent to one of the UD tags. Each of the training and test data POS tag sequences was transformed into an ordered list of POS tag vectors. The training POS tag vector lists are processed by the RNN during training, while the ones that represent the test data are only used during testing. The RNN's output is a language label, i.e., English or Chinese. This output represents the language that the input POS tag sequence most resembles, according to the RNN syntactic model.

The recurrent neural network implementation from the Python library PyTorch[1] was used to create the negative language transfer detection model. The model was trained for ten epochs with Adam optimization (Kingma and Ba, 2015). The RNN model's hyperparameter combination (16 hidden units, learning rate = 0.0001, mini batch size = 1, and negative log likelihood as the loss function) was selected from a set of options in the hyperparameter space. The hyperparameter tuning process is described in section 6.2.

Throughout the training process, the RNN model learnt how to make a prediction based on the POS tag vector lists. It received these sequence representations as input and made a language prediction, i.e., English or Chinese. This prediction reflected the language structures that the input sequence was

[1] `https://pytorch.org/`

more related to, according to the RNN shallow syntactic language model internal computation. The RNN weights were adjusted via backpropagation. The backpropagation error vector was computed based on the difference between the model's output and the true language value for the input POS tag sequence.

After being trained with the training dataset and the best hyperparameter combination, the RNN shallow syntactic language model was used to determine the relation between errors made by English as a second language learners and the language transfer phenomenon. When the RNN shallow syntactic language model outputted that a given POS tag sequence extracted from a learner error was more similar to Chinese structures than to English structures, the error was deemed related to negative language transfer and classified as such. Figure 6.1 illustrates the RNN testing procedure.

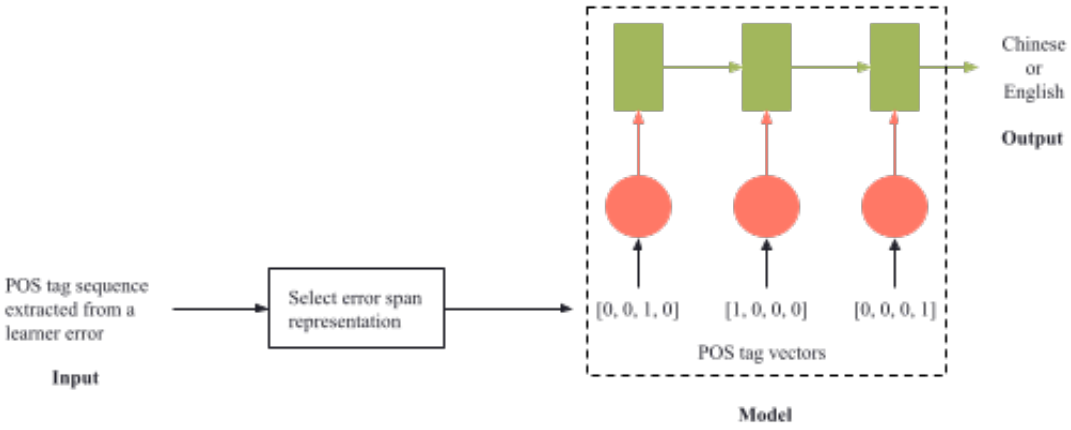

Figure 6.1: RNN approach testing procedure

Each structural error made by Chinese native speakers in the FCE dataset was analysed by the RNN model using the three distinct error spans. During testing, the RNN assigned a language label to the POS tag sequence extracted from the error. When that language label was "Chinese", the error was classified as negative language transfer. The RNN model's output was interpreted as an indication that that error's structure was more similar to Chinese than to English structures. The negative language transfer label associated with the error was compared to the negative language transfer annotation that accom-

panied that error. The model's performance in detecting negative language transfer was computed based on how often the model's outputs matched the negative language transfer annotations.

6.2 Hyperparameter tuning

Akin to the n-gram models, parameter tuning was performed to select the best hyperparameter combination for the recurrent neural network model. In this process, the training dataset was divided into training and evaluation splits. The training split contained 80% of the training dataset, while the evaluation split was composed of the remaining 20% of the training dataset. Table 5.2 in section 5.2 contains the number of POS tag sequences in each of these splits.

In the RNN tuning procedure, the training and evaluation splits contain POS tag sequences extracted from both English and Chinese sentences. Each RNN model was trained with a different parameter combination and learnt to predict the source language of POS tag sequences from the training split. In the n-gram model tuning, the training data was split according to the source language of its sentences and each of the two resulting monolingual splits was used to train a distinct n-gram model. In the RNN tuning procedure, however, the monolingual training splits were merged into a single dataset, and the sequences in this dataset were the RNN model's input. The model's output was a prediction of the language represented by the input POS tag sequences.

The hyperparameters analysed during the tuning procedure were the number of hidden units, learning rate, mini batch size, and loss function. Five learning rate values were analysed, 0.01, 0.001, 0.0001, 0.00001, and 0.000001. One of seven hidden layer sizes was used to test each hyperparameter combination. The hidden layers were composed of 8, 16, 32, 64, 128, 256, or 512 units. The networks computed 1, 2, 4, 8, 16, or 32 samples per mini batch. Finally, two loss functions were analysed, negative log likelihood[2] and binary cross entropy with logits[3].

[2] `https://pytorch.org/docs/stable/generated/torch.nn.NLLLoss.html`
[3] `https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.`

Each combination of these parameters and the training split were used to train one RNN shallow syntactic language model for 10 epochs. The resulting RNN models were tested on the evaluation split. Each resulting model's accuracy was computed as the percentage of correctly identified samples in the evaluation split.

The model that achieved the best accuracy on the evaluation set had 16 hidden units, a learning rate of 0.0001, mini batch size = 1, and negative log likelihood as its loss function. Although the learning rate value selected may seem small, it is used in parallel with mini batch size = 1. This combination should be acceptable because the network weights are updated after each training sample is processed, and in this setting a small learning rate helps to prevent the network from overshooting local minima, i.e., making weight updates that are too large to converge over time (Buduma and Locascio, 2017). The best performing model achieved 95.16% accuracy on the evaluation split, and it was used to train the RNN shallow syntactic language model applied in negative language transfer detection. The evaluation split accuracy results for all hyperparameter combinations analysed can be found in Appendix C. The evaluation results yielded during tuning were consistent across multiple runs with random initialization.

Note that the evaluation accuracy achieved in the n-gram models hyperparameter tuning is greater than the best evaluation accuracy attained by the RNN during tuning. These results can be explained by the type of data being modelled. POS tag sequences extracted from grammatical and high-quality (i.e., manually generated and translated) data follow a well-defined and consistent structure. The regular structure of these sequences is well-represented by n-gram models. Hence, the superior performance observed in the n-gram models' tuning. Each monolingual n-gram model is capable of inferring whether a POS tag sequence belongs to the language structure represented. However, it is not able to make inferences about that sequence belonging to the language modelled by the other n-gram model. In this aspect, an RNN model fits the negative language transfer detection task better. It is able to distinguish be-

`html`

Span	Precision	Recall	F1-score
Padded error	0.69	0.34	0.46
Error + unigram	0.67	**0.41**	**0.51**
Error + bigram	**0.70**	0.35	0.46

Table 6.1: RNN approach negative language transfer detection results

tween English and Chinese and classify POS tag sequences as belonging to a language instead of the other.

6.3 Results

Table 6.1 contains the precision, recall, and F1-score results attained by the RNN shallow syntactic language model in detecting negative language transfer. The error span that yielded the best recall and F1-score results was the error + unigram span. This span encompasses the POS tags extracted from the error followed by the POS tag extracted from the word that comes immediately after the error. For example, the error + unigram sequence retrieved from the pronoun agreement error in the sentence "This are only my immature views." contains the POS tags extracted from the words "This are".

A parallel can be drawn between the RNN and the n-gram baseline analysing errors represented by the error + unigram span, as this error span also yielded the lowest precision scores among the RNN results. With the RNN approach, the padded error span and error + bigram span representation attained higher precision scores, 0.69 and 0.70 respectively. Compared to the error + unigram span, the padded error span encompasses one extra POS tag, extracted from the word that precedes the error. The error + bigram span also contains one extra POS tag, extracted from the word that follows the words represented by the error + unigram span. These results may indicate that to make more accurate predictions regarding negative language transfer, the RNN model benefits from more context extracted from the errors' surroundings.

The results yielded by the RNN approach where learner errors were represented by the padded error and error + bigram spans were analogous. The F1-scores achieved by the RNN analysing these error spans were the same,

but the error + bigram span performed slightly better in precision and recall. Both spans yielded considerably lower recall scores when compared to the error + unigram span results. Lower recall scores indicate that the padded error and error + bigram spans drove the RNN to classify more negative language transfer related errors as not related to transfer.

Furthermore, the recall results using the padded error span are lower than the ones from the error + unigram and error + bigram spans. This may be an indication that, in the RNN approach, the POS tag extracted from the word that precedes the error may not benefit negative language transfer detection as much as the words that follow the error. That is, the error's preceding context is not as important as its successive context for detecting negative language transfer related errors.

Another feature that may hinder the RNN shallow syntactic language model's performance in detecting language transfer is the level of detail offered by the POS tagset used. The RNN model is able to distinguish between English and Chinese structures more accurately than the n-gram baseline. However, there are some error types for which the current POS tag representation is not detailed enough to support negative language transfer detection.

For example, replacement errors are a category in which learners use an incorrect token from the correct category. This means that the POS tag sequence extracted from these errors is the same as the POS tag sequence that would be extracted from correct versions of the learners' writing. The POS tag sequence extracted from replacement errors is valid in English. The learner error lies on the choice of token, not on its category. Representing language structures using POS tags does not convey enough information for replacement errors to be correctly classified as related to negative language transfer or not.

As my language structure representation and POS tagset choices affect both n-gram baseline and RNN approaches, the limitations of the chosen methodology are considered in greater depth in the discussion chapter. The following section examines the RNN performance on the most common errors made by Chinese native speakers when writing in English in the FCE dataset.

Error type	Span	Precision	Recall	F1-score
Replace punctuation (n = 336)	Padded error	**0.71**	0.39	0.51
	Error + unigram	0.70	**0.47**	**0.56**
	Error + bigram	**0.71**	0.38	0.50
Wrong verb tense (n = 267)	Padded error	0.78	0.39	0.52
	Error + unigram	0.77	**0.41**	**0.54**
	Error + bigram	**0.79**	0.29	0.42
Missing determiner (n = 209)	Padded error	**1.00**	0.26	0.41
	Error + unigram	0.98	**0.45**	**0.61**
	Error + bigram	0.98	0.40	0.57

Table 6.2: RNN model's results on most common error types (an extended version of this table, containing all error types, is available in Appendix E)

6.4 Error analysis by grammatical error type

Table 6.2 presents the RNN shallow syntactic language model's precision, recall, and F1-score results for three of the most common Chinese native speaker error types. As discussed in section 5.4, punctuation replacement and wrong tense of verb errors are harder to represent in this methodology due to the POS tagset used. This tagset represents punctuation marks and verbs using POS tags that are too general and do not convey information about the type of punctuation or verb tense used. Nonetheless, the error spans seem to be reasonably precise when applied to represent negative language transfer in instances from these error types. This may occur because the RNN model is able to detect usage patterns that happen in parallel with punctuation and verb tense errors and that may be indicative of negative language transfer.

In contrast, missing determiner errors seem to be better represented by Universal Dependencies POS tag sequences. The RNN shallow syntactic language model is able to differentiate between Chinese and English usage patterns that involve missing words, such as in the POS tag sequences from missing determiner errors. This ability is corroborated by the higher scores achieved in detecting negative language transfer in "missing" error instances.

When the RNN model analysed missing determiner errors using the padded error span, it was very precise. All of the missing determiner errors that the model classified as negative language transfer related were in fact related to

language transfer. However, this error span was not so successful in capturing all of the negative language transfer related missing determiner errors. Its recall score was 0.26.

The error + unigram span achieved high precision and the highest recall scores when classifying missing determiner errors. It seems to be more suitable for representing this error type. In the RNN shallow syntactic language model approach, the POS tag extracted from the word that precedes the error was not as helpful in detecting negative language transfer as the one extracted from the word that follows the error. The model performed better in detecting negative language transfer when only the POS tag that followed the error was added to the POS tag sequence analysed.

One possible explanation for this result is that it is common in English for determiners to be at the beginning of sentences. The error + unigram POS tag sequences began with a POS tag that should be preceded by a determiner in English but was not, then the model correctly assigned the sequence a Chinese, i.e., negative language transfer, label. The padded error span was not so successful in that regard because there are more mid-sentence contexts in English in which determiners are not necessary.

The hypothesis that the RNN expects determiners to be at the beginning of POS tag sequences in English, as opposed to Chinese, is supported by the error + bigram span scores in detecting negative language transfer related missing determiner errors. This span is analogous to the error + unigram span as it begins with the POS tag that should be preceded by a determiner in English. Its recall and F1-score results indicate that the RNN was able to detect more missing determiner errors that are related to negative language transfer with the error + bigram span than with the padded span.

Altogether, the RNN approach achieved higher scores than the n-gram baseline in most of the settings analysed. In the following chapter, I will discuss the similarities and differences between these two negative language transfer detection approaches.

Chapter 7

Discussion

This chapter compares and discusses the results obtained using the n-gram baseline and RNN approaches described in chapters 5 and 6. Beyond that, it discusses the approaches' limitations, the implications of the methods used, and possible future research directions.

7.1 Performance analysis across models

The error span being equal, the RNN approach achieved higher scores than the n-gram baseline in all settings. The only setting in which the n-gram models outperformed any of the RNN results was when the n-gram models classified negative language transfer errors using the padded error span. The precision achieved by that combination was higher than the one achieved by the RNN when using the error + unigram span. Overall, the n-gram models yielded lower recall scores regardless of the error span analysed. This resulted in the n-gram baseline's F1-scores also being lower than the RNN F1-scores.

The results yielded by each approach indicate that RNN models are more suited to identify negative language transfer in learner errors. Compared to n-gram models, RNN can take more previous context into account when analysing a POS tag sequence. This feature can be useful to detect negative language transfer in errors which are longer than the n-gram models' length, i.e., are composed of 6 POS tags or more. Apart from this, the RNN model used in this task was trained to differentiate between Chinese and English structures. The knowledge about both language structures might have given

	RNN correct	**RNN incorrect**
N-gram correct	896	234
N-gram incorrect	270	965

Table 7.1: Padded error span contingency table

Error span	**Test statistics**	**Additional errors**
Padded error	$\chi^2 = 2.43$, $p = .11$, $OR = 0.86$	36
Error + unigram	$\chi^2 =$ **4.33,** $p =$ **.03,** $OR =$ **0.78**	38
Error + bigram	$\chi^2 =$ **33.25,** $p <$ **.001,** $OR =$ **0.56**	121

Table 7.2: McNemar's test results for each error span representation (OR stands for odds ratio, this measure represents the ratio between RNN and n-gram misclassifications)

the RNN approach the upper hand as it could directly delineate the distinction between negative language transfer related errors and non-negative language transfer related errors.

To analyse whether the incorrect classifications made by the language modelling approaches are statistically different, the approaches' misclassifications were compared using McNemar's test. This statistical test examines the amount of test samples that were misclassified by each approach to comment on whether the approaches err similarly (Dietterich, 1998). Under McNemar's test, the null hypotheses is that both the n-gram models and the RNN misclassify a similar amount of test samples.

The test statistic for McNemar's test is computed from a contingency table. This table represents the distribution of correct and incorrect classifications for each classifier as well as how the classification results overlap. Table 7.1 presents the contingency table for these models when learner errors were represented by the padded error span. The value 896 in the first table cell is the number of learner errors that were correctly assigned a negative language transfer label by both of the RNN and n-gram models. Similarly, the value 965 is the number of learner errors that received, from both approaches, a negative language transfer label that differed from their gold standard annotation. The values 234 and 270 represent learner errors that were correctly classified by one of the approaches but not by the other.

	RNN correct	RNN incorrect
N-gram correct	1038	139
N-gram incorrect	177	1011

Table 7.3: Error + unigram span contingency table

	RNN correct	RNN incorrect
N-gram correct	907	156
N-gram incorrect	277	1025

Table 7.4: Error + bigram span contingency table

McNemar's test evaluates how different the errors made by the approaches are. That is, whether the values 234 and 270 are statistically different in this context. As show in Table 7.2, the test statistics computed from the padded error span contingency table were not significant, which means that the misclassifications made by two approaches were not measurably different when analysing errors represented by the padded error span.

The null hypotheses could not be rejected based on the results achieved when errors were represented by the padded error span, but the statistical analysis of results obtained when learner errors were represented using the error + unigram and error + bigram spans yielded significant results. Table 7.2 contains the test statistics, p-values, and number of additional misclassifications for each of the error spans analysed. The RNN model made 38 fewer misclassifications than the n-gram models when learner errors were represented by the error + unigram span, and this difference was statistically significant. The null hypothesis was rejected with $p = .03$. Table 7.3 is the contingency table for the models' results when analysing learner errors represented by the error + unigram span.

Representing learner errors with the error + bigram span also yielded different amounts of misclassifications from the two approaches, with the RNN model making fewer misclassifications again. When errors were represented by the error + bigram span, the n-gram models made 121 additional misclassifications when compared to the RNN model. By applying McNemar's test to the models' misclassifications, the null hypothesis was rejected with a small p-value ($p < .001$), which means that there is a significant difference in the

number of errors made by each approach. Table 7.4 contains the distribution of model mistakes between n-gram and RNN models when learner errors were represented by the error + bigram span.

7.2 Error analysis across models

By examining the different approaches' results on the most common error types, it is possible to identify a similar pattern, in which the RNN approach frequently yields higher scores. As previously discussed, the two most common error types in the test dataset are replacement and verb tense errors. These error types are not comprehensively represented by the methodology used. Representing language structures with POS tag sequences means that information that is important to detect negative language transfer in instances of these error types is not modelled. Therefore, to provide a fair comparison between the approaches, their performance on instances of the third and fourth most common error types made by Chinese native speakers are compared and contrasted.

Both the third and fourth most common error types in the test dataset are omission, or "missing", errors. Missing determiner is the third most common error type in the dataset, with 209 instances, and missing punctuation is the fourth most frequent error type with 152 instances. Determiner omission errors are highly correlated to negative language transfer as Chinese does not have determiner equivalents, which causes Chinese native speakers to omit determiners (Robertson, 2000; Han et al., 2006). Similarly, the usage of punctuation marks in Chinese diverges from English rules (Liu, 2011). In Chinese, commas are commonly used as sentence boundaries, in parallel to how periods are used in English (Xue and Yang, 2011). Chinese speakers who are learning English may refrain from using commas mid-sentence as they are used to commas marking the end of complete thoughts.

The scores achieved by analysing missing determiner errors with different error spans are consistent with the overall RNN approach results. The RNN's highest F1-score came from analysing the errors using the error + un-

igram span. Unlike the overall trend, the n-gram baseline yielded the highest F1-score when it analysed missing determiner errors using the padded error span. The highest F1-scores achieved by each best performing approach and span combination are similar (Figure 7.1). The baseline and RNN approaches achieved F1-scores of 0.6 and 0.61 when detecting negative language transfer in missing determiner errors using the padded error and error + unigram spans, respectively.

As previously discussed, the padded error span was more successful in representing missing determiner errors for the n-gram models. This may happen because this span encapsulates the POS tags that surround the position where the determiner should be, and the n-gram models can make accurate inferences from these sequences. Determiner omission is a pattern in Chinese native speakers' English writing because determiners are not used in Chinese (Robertson, 2000). The POS tag sequences extracted from missing determiner errors, which do not contain determiner POS tags, are more similar to Chinese structures and are related to negative language transfer.

The English n-gram model assigned the missing determiner error sequences a low probability, because it is more common in English to have determiners between the sequence's POS tags. The Chinese model did not expect determiners to be present in any POS tag sequence, consequently, it assigned the missing determiner error sequences a high probability. As the probability assigned by the Chinese n-gram model is higher than the one assigned by the English model, most missing determiner errors were correctly classified as negative language transfer.

The RNN model achieved the highest F1-score when it analysed missing determiner errors using the error + unigram span. This error span was able to represent missing determiner errors as negative language transfer as it lacks a determiner POS tag at the beginning of the sequence. The RNN model has learnt to process English POS tag sequences that begin with a determiner tag followed by the POS tags extracted from a noun phrase. The POS tag sequences extracted from missing determiner errors using the error + unigram span begin with the tags extracted from the noun phrase. This represents a

disconnect with what the RNN model expects from an English sequence, which prompts the model to classify the sequence as Chinese, i.e, negative language transfer related.

The RNN model is not able to correctly classify most missing determiner errors when analysing them with the padded error span. The RNN achieved the lowest F1-score among all model/span combinations when it analysed determiner omission errors represented by the padded error span. Although not expected at first, these results can be explained by the RNN model learning that certain patterns extracted from missing determiner errors are valid in English. There are certain English grammar rules that allow noun phrases to be introduced without a determiner in the middle of a sentence. For example, determiners are omitted after some prepositional phrases, such as in "He went to bed", and before lists of nouns, such as in "We talked to father and grandfather". The RNN model may have learnt that these patterns are valid in English and identified them as English in POS tag sequences extracted from missing determiner errors represented by the padded error span.

As Figure 7.1 shows, the error span that yielded the highest F1-scores in analysing missing punctuation errors was the same for both approaches. The n-gram baseline and RNN approaches achieved the highest F1-scores when analysing missing punctuation errors using the padded error span.

The padded error span encompasses the words that surround the position where a punctuation mark should be. It is composed of the POS tag extracted from the word that precedes the error followed by the POS tag extracted from the word that succeeds the error. When detecting negative language transfer in missing punctuation errors, the padded error span achieved higher scores than the error + unigram and error + bigram spans in both approaches. This indicates that missing punctuation errors' preceding contexts are useful for negative language transfer detection.

One possible explanation for these results is that the structural pattern represented by the padded error span helped the n-gram and RNN models to connect POS tag sequences to Chinese punctuation usage. Another explanation is that the lack of a punctuation mark between the sequences' POS tags

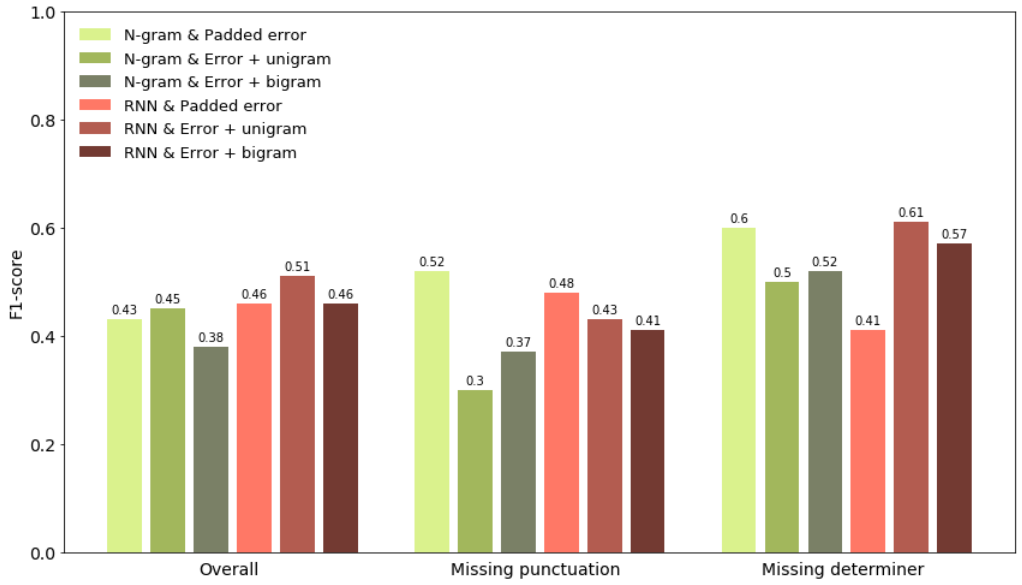

Figure 7.1: F1-scores for shallow syntactic language models and error spans combinations

diverges from English grammar rules and causes the models to classify the sequences as belonging to Chinese language structures.

The n-gram models analysing missing punctuation errors represented by the padded error span outperformed all of the RNN results. They achieved an F1-score of 0.52, while the RNN F1-score results were 0.48, 0.43, and 0.41. These results are an indication that the n-gram models were more successful in associating missing punctuation errors to negative language transfer effects. The n-gram models analysing errors represented by the padded error span yielded the highest precision and recall scores in detecting when a POS tag sequence was more similar to Chinese language structures than to English ones. That is, this setup was able to more accurately and broadly detect when the POS tag sequence required a punctuation mark in English, but did not in Chinese.

The results discussed in this chapter suggest that to achieve more precise and comprehensive negative language detection performance it is necessary to analyse distinct contexts depending on the error type and detection approach. For example, when analysing missing determiner errors with an RNN shallow syntactic language model, the model performance increased by representing

the errors with the error + unigram span. However, if n-gram models were used, the best representation for these errors was the padded error span. Similarly, punctuation omission errors benefitted from being represented by the padded error span regardless of the detection approach. These observations indicate that different errors need to be represented using different spans to accurately represent the errors' negative language transfer characteristics. Errors from each error type should be represented by an appropriate span before being analysed by negative language detection models in an automatic grammatical error feedback system.

7.3 Limitations

The methodology used to detect negative language transfer in Chinese learners' writing introduced a few limitations that hindered model performance. Both approaches were limited in their ability to represent language structures using the Universal Dependencies tagset, as this tagset can be too general to capture some language patterns.

As previously mentioned, the Universal Dependencies POS tagset was created as a multilingual annotation scheme. Its POS tags represent word categories that are common among a variety of languages. This commonality allowed my negative language transfer detection models to be trained and tested using the same language structure representation for two distinct languages. However, to provide a multilingual annotation scheme, the Universal Dependencies POS tagset sacrificed the level of detail that each POS tag is able to represent.

The POS tags in the Universal Dependencies tagset represent general word categories, e.g., nouns, verbs, and adjectives, but these POS tags do not contain information about grammatical features such as tense, number, gender, or form. Although the Universal Dependencies scheme provides morphological annotations that cover some of these features, they are not common across languages and, hence, not represented by the main tagset.

The lack of details encountered in some of the POS tag representations

of learner errors hindered the models' negative language transfer detection performance. For example, the accurate classification of most errors involving verb usage are harmed by the POS tagset used. The Universal Dependencies tagset uses two POS tags to represent verbs, and these tags do not contain information about features, such as person, form, and tense. Being able to analyse these verb features would help the model distinguish among verb usage patterns that are more common in English than in Chinese, and vice versa.

Although a more detailed POS tagset, such as the Penn Treebank (Marcus et al., 1993), could improve the representation of learner English structures, the methodology applied required that both language structures were represented using the same POS tags. At the moment, the Universal Dependencies tagset is the only tagset I am aware of that provides a shared annotation scheme between English and Chinese.

To overcome the limitation posed by Universal Dependencies POS tagset's level of detail, it would be valuable to create a POS annotation scheme specific to English and Chinese word categories. This process would involve developing an inventory of POS tags that are applied similarly in both languages, e.g., both languages distinguish between nouns and proper nouns (Xue et al., 2005). The resulting shared POS tagset would then be applied to POS tag the training and test data used in the task. A POS tagset based on shared word categories between English and the learners' L1 would allow POS language structures to be represented in as much detail as possible, and perhaps, improve the models' negative language transfer performance.

A more detailed POS tagset would help identify negative language transfer in some learner structures. However, representing language structures with POS tags does not suffice to identify all types of language transfer related errors. For example, model performance in detecting negative language transfer related replacement errors shows that these error types cannot be well represented by POS tag sequences. The methodology reported in this thesis is only suited for structural errors, errors in which the learner has incorrectly arranged the words in their writing. However, there are errors which are more related to semantic incongruences. For example, the word "number" in the sentence

"It is because of the increasing number of the population" was annotated as a noun replacement error. The annotator suggested that it should be replaced by the word "size". The POS tag sequence extracted from the incorrect sentence would not be different from the one extracted from the corrected version of the sentence. Semantic negative language transfer such as the one shown in this error requires more information than POS tag sequences to be correctly identified.

A negative language transfer detection solution for semantic errors would require a new methodology, one that does not solely consider structural patterns in the errors. One possible direction is to use the negative language transfer annotations developed as part of this thesis work to fine-tune grammatical error detection models. After the adaptation, these models would be able to recognize semantic patterns that are related to negative language transfer. For example, whenever the word "number" is used to refer to an uncountable noun, such as the word "population" in the example above, the models would classify the error as negative language transfer related.

Apart from the POS tag language representation, the n-gram baseline was also limited by n-gram models not being able to directly distinguish between two languages. That is, the n-gram baseline's negative language transfer detection was derived from the outputs from two independent models. The model trained on POS tag sequences extracted from English sentences was only able to output how similar a test POS tag sequence was to English structures. It could not infer its similarity to Chinese structures. In the same way, the model trained on Chinese POS tag sequences was only able to compare the test POS tag sequences to Chinese structures.

The n-gram models' independence may have limited their negative language transfer detection power as each model's output only indicated the similarity shared between the POS tag sequence and the language represented by the model. It is possible that the comparison between model outputs worked well when there was a clear distinction between the structure from the learner errors and English language structures. For example, when the structure used by the learner is very common in Chinese but rare or invalid in English. In

those situations, the n-gram model that represents Chinese structures would assign the POS tag sequence a high probability value, while the one that represents English structures would assign it a low probability value. As a result, the learner error would be correctly identified as negative language transfer. Sequences such as these, in which erroneous structures were assigned high Chinese probabilities and low English ones, occurred in 34% of negative language transfer related learner errors.

The FCE test-takers have acquired some level of proficiency in English. Hence, they may not fully rely on Chinese patterns when writing in English, but rather inconsistently try to apply both languages' patterns, using both language structures at the same time or erratically switching between the two. This interlingual process could make the learner error structures ambiguous to the independent n-gram shallow syntactic language models, and cause them not to be able to correctly identify many negative language transfer related errors.

In relation to that, it is important to note that as the FCE exam assesses an upper-intermediate level of English proficiency, FCE test-takers commonly demonstrate high proficiency in English. This proficiency factor may influence what type of negative language transfer occurs in the test dataset, and hence, the models' performance in negative language transfer detection. As language learner errors and negative language transfer occurrences vary according to learner proficiency level (Bardovi-Harlig and Sprouse, 2017), it would be valuable to investigate negative language transfer detection performance on learner datasets that represent other learner populations, e.g., language learners who have only recently started to learn English.

7.4 Implications

My methods perpetuate biases favouring what is viewed as grammatical English. The datasets used to train the shallow syntactic language models are filled with a formal and grammatical English variant, and the language structures extracted from it are used to evaluate learner errors. As discussed by

Dixon-Román et al. (2020), the use of these datasets is, in a certain way, imposing that same writing style and its associated structures on learners. The errors in the test dataset were annotated following a procedure that penalizes divergence from English grammatical rules, regardless of whether the learner sentences' were able to convey their intended meaning or not. That is, even if a piece of text is completely understandable in the way that it was written, it will be annotated as erroneous where it diverges from English grammar rules. The nature of the datasets and methods used in this work conserves these biases towards a "standard" grammar usage.

In many cases, individuals learn a new language due to social expectations. It is expected that to join graduate school or get a promotion one would need to be proficient in a language other than their mother tongue. English proficiency is often recognized via certification exams, such as the International English Language Testing System (IELTS), Test of English as a Foreign Language (TOEFL), or even the FCE exam. Exams like these tend to prioritize what can be defined as a standard English grammar, the application of formal and grammatical variants of the English language. Although in everyday life, both native and non-native speakers are accustomed to a more informal language usage. The fact that certification exams require certain types of language structures prevents learners from deviating from this standard and compels learners to follow these rules. For language learners, it is necessary to abide by these rules so that certification is obtained. Such requirements suppress the usage of other English variants, perhaps more familiar to the learners, and standardize English teaching to non-native speakers. Unfortunately, the power dynamic in language learning environments is hardly in favour of learners, and this could be viewed as an attempt to homogenize non-native English teaching and learning.

However damaging this homogenization may be, it is still the system in place. Add to it the advances in automatic writing assessment, and it is possible that the language requirements applied currently will become more strict, since it is considerably more difficult for a computational system to understand language nuances that occur in learner writing. Given that the

assessment of learner writing becomes more deterministic and precise, my work is an attempt to provide an error cause explanation to learners. I aim to help learners understand how using elements of their native language influences their English writing and support their acquisition of a better understanding of both languages. This information can help learners improve their writing accuracy, according to a standard English grammar and ultimately support them in high-stakes situations, such as taking a proficiency exam.

7.5 Future directions

The results reported in the previous chapters show that it is possible to detect some negative language transfer related structural errors in learner data. Overall, the negative language transfer detection models achieved good precision results, with the RNN model performing better than the n-gram models in most cases. To understand whether and how these models' negative language transfer detection outputs could aid language learning, the next step in this research project is to analyse language learner writing performance in English after receiving negative language transfer informed error feedback.

To instrument this analysis, the negative language transfer detection model output would need to be integrated into a pre-existing writing assistant application that provides error feedback to language learners. Feedback for learner errors also needs to be developed from the error cause annotation described in Appendix A and integrated into this application. As a result, along with feedback on how to correct the errors, application users would be presented with feedback detailing why their errors were possibly related to differences between their first language (i.e., Chinese) and English.

The analysis of changes in learner writing performance could be accomplished through a user study in which two groups of Chinese native speakers who are learning English work on essay writing tasks and receive error feedback with and without metalinguistic cues. In this study, the learners in the control group would work on essay writing tasks using a writing assistant that provides direct corrective feedback. The learners in the treatment group would work

in the same environment, but the direct corrective feedback provided by the writing assistant would be enhanced with metalinguistic feedback regarding negative language transfer. The high precision scores achieved by the detection models mean that using their output to inform error feedback represents a lower risk of potentially confusing learners. That is, as the number of false positives in the model's outputs is low, there is a smaller chance that learners will mistakenly get negative language transfer feedback when their errors are not connected to the phenomenon.

The learners' writing performance would be recorded before and after completing the tasks. Their performance could be measured with metalinguistic awareness assessments such as error correction tasks, in which learners are tasked with correcting negative language transfer errors in ungrammatical sentences; cloze tests, in which learners need to select the most appropriate word or phrase to fill the gap in a sentence from a set negative language transfer and correct alternatives; and selection tasks, in which learners need to select the grammatical version of a sentence from a list of options (Chireac et al., 2019).

These metalinguistic awareness assessments would happen soon after the experiment, i.e., after the participants used the writing assistant to work on essays, and at least once again after more time is passed to analyse if improvements from receiving metalinguistic feedback, if any, are lasting. The results from these tests would then be compared and contrasted within and between groups to draw conclusions regarding the effectiveness of negative language transfer feedback on English learners' writing.

Chapter 8

Conclusion

Native and non-native speakers make different errors when writing in English. The errors made by non-native speakers are often related to patterns from their first languages that get transferred to their L2, i.e., English, writing. This transfer phenomenon is called language transfer, and it is a well-studied phenomenon in second language acquisition research. The occurrence of language transfer between L1 and L2 is also investigated in natural language processing research, as it can aid native language identification and grammatical error correction systems to perform their tasks.

Although the strategy of transferring language patterns across languages can positively impact learners' communication, this strategy can cause errors when L1 and L2 patterns are divergent. When the language transfer strategy results in mistakes, it is classified as negative language transfer. Most language learners are not aware of it when they transfer structures incorrectly, as they are accustomed to those structures in their L1s.

To attempt to address the issue of learners' lack of awareness regarding negative language transfer, this thesis presents a method to identify when learner errors are related to their L1 structures. The method presented involves extracting part-of-speech tags from parallel, high-quality text in English and in the learners' L1. These POS tag sequences are used to train language models to differentiate between language structure representations of each language. After the language models are trained, they are used to analyse part-of-speech tag sequences extracted from learner errors. If the language models deem that

the error structure is more similar to the learner's L1, the error is classified as negative language transfer related.

Two language modelling techniques were used to represent the language structures in my thesis. They were n-gram language modelling and RNN-based language modelling. Each learner error was presented to the language models along with varying amounts of context to investigate how much of their surrounding context is necessary to better represent negative language transfer.

This methodology was validated by applying it to errors made by Chinese native speakers whose essays are part of the First Certificate in English dataset. These errors were annotated with information about their relation to the negative language transfer phenomenon. The RNN-based language model outperformed the n-gram language models in all settings analysed. The results also show that representing the errors with the POS tags extracted from the error along with the POS tag extracted from the word that follows the error achieved the highest recall and F1-score results. The highest precision scores were achieved by spans that incorporated more of the error's surrounding context in the POS tag sequence representations.

The negative language transfer detection methodology applied was limited by the type of language structure representation chosen, POS tag sequences. A POS tagset that can be applied to two languages needs to abstract some syntactic and morphological features as they might not be common across languages. This factor causes some POS tag sequences to be too coarse to capture the incorrectness of the learner structure. Furthermore, some learner errors cannot be comprehensively represented by POS tag sequences as it is the semantics of the text in the error that are incorrect.

Nonetheless, the high precision scores achieved by the models in detecting negative language transfer in structural learner errors indicate that it would be safe to perform a user study with language learners. That is, as the models output few false positives, the participants in a user study would not be faced with negative language transfer feedback for errors that are not related to negative language transfer, helping them create an accurate concept of the

phenomenon. The models' output would need to be integrated into a writing assistant in order for metalinguistic feedback about negative language transfer to be provided to language learners. Through a user study, it would be possible to analyse the impact of negative language transfer feedback in language learner writing and, hopefully, find evidence that this feedback type improves their understanding of English.

References

Dimitris Alikaniotis and Vipul Raheja. The unreasonable effectiveness of transformer language models in grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 127–133, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4412. URL https://www.aclweb.org/anthology/W19-4412.

Gaston Bacquet. Is corrective feedback in the ESL classroom really effective? A background study. *International Journal of Applied Linguistics and English Literature*, 8:147–154, 12 2019. doi: 10.7575/aiac.ijalel.v.8n.3p.147.

Kathleen Bardovi-Harlig and Rex A. Sprouse. *Negative Versus Positive Transfer*, pages 1–6. 2017. ISBN 9781118784235. doi: https://doi.org/10.1002/9781118784235.eelt0084. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118784235.eelt0084.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5301. URL https://www.aclweb.org/anthology/W19-5301.

Yevgeni Berzak, Roi Reichart, and Boris Katz. Contrastive analysis with

predictive power: Typology driven estimation of grammatical error distributions in ESL. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 94–102, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.18653/v1/K15-1010. URL https://www.aclweb.org/anthology/K15-1010.

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. doi: 10.18653/v1/W19-4406. URL https://www.aclweb.org/anthology/W19-4406.

Nikhil Buduma and Nicholas Locascio. *Fundamentals of deep learning: Designing next-generation machine intelligence algorithms*. " O'Reilly Media, Inc.", 2017.

Silvia-Maria Chireac, Norbert Francis, and John McClure. Awareness of form and pattern in literacy assessment: Classroom applications for first and second language. *Reading*, 19(1):20–34, 2019.

Shamil Chollampatt, Duc Tam Hoang, and Hwee Tou Ng. Adapting grammatical error correction based on the native language of writers with neural network joint models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1901–1911, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1195. URL https://www.aclweb.org/anthology/D16-1195.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Thomas G Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998.

Ezekiel Dixon-Román, T. Philip Nichols, and Ama Nyame-Mensah. The racializing forces of/in ai educational technologies. *Learning, Media and Technology*, 45(3):236–250, 2020. doi: 10.1080/17439884.2020.1667825. URL https://doi.org/10.1080/17439884.2020.1667825.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1024. URL https://www.aclweb.org/anthology/N16-1024.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990. ISSN 0364-0213. doi: https://doi.org/10.1016/0364-0213(90)90002-E. URL https://www.sciencedirect.com/science/article/pii/036402139090002E.

Brendan Flanagan, Chengjiu Yin, Takahiko Suzuki, and Sachio Hirokawa. Prediction of learner native language by writing error pattern. In Panayiotis Zaphiris and Andri Ioannou, editors, *Learning and Collaboration Technologies*, pages 87–96, Cham, 2015. Springer International Publishing. ISBN 978-3-319-20609-7.

Nan Rae Han, Martin Chodorow, and Claudia Leacock. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12, 2006. ISSN 13513249. doi: 10.1017/S1351324906004190.

Zhao Hong Han. Fine-tuning corrective feedback. *Foreign Language Annals*, 34, 2001. ISSN 0015718X. doi: 10.1111/j.1944-9720.2001.tb02105.x.

Kenneth Heafield. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, July 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W11-2123.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P13-2121.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., USA, 2009. ISBN 0131873210.

Daniel Jurafsky and James H. Martin. Speech and language processing (3rd ed. draft), 2020.

Khaled Karim and Hossein Nassaji. The revision and transfer effects of direct and indirect comprehensive corrective feedback on ESL students' writing. *Language Teaching Research*, 24(4):519–539, 2020. doi: 10.1177/1362168818802469. URL https://doi.org/10.1177/1362168818802469.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Irit Kupferberg. The cognitive turn of contrastive analysis: Empirical evidence. *Language Awareness*, 8, 1999. ISSN 17477506. doi: 10.1080/09658419908667130.

Robert Lado. *Linguistics Across Cultures: Applied Linguistics for Language Teachers*. University of Michigan Press, 1957. URL https://books.google.ca/books?id=ZzYGAQAAIAAJ.

Amna Liaqat, Cosmin Munteanu, and Carrie Demmans Epp. Collaborating with Mature English Language Learners to Combine Peer and Automated Feedback: a User-Centered Approach to Designing Writing Support. *International Journal of Artificial Intelligence in Education*, pages 1–42, 2020.

Orly Lipka. Syntactic awareness skills in english among children who speak slavic or chinese languages as a first language and english as a second language. *International Journal of Bilingualism*, 24(2):115–128, 2020. doi: 10.1177/1367006918812186. URL `https://doi.org/10.1177/1367006918812186`.

Xing Liu. The not–so–humble "Chinese comma": Improving English CFL students' understanding of multi–clause sentences. In *Proceedings of the 9th New York International Conference on Teaching Chinese*, 2011.

Roy Lyster and Leila Ranta. Corrective feedback and learner uptake: Negotiation of form in communicative classrooms. *Studies in Second Language Acquisition*, 19(1):37–66, 1997. doi: 10.1017/S0272263197001034.

Shervin Malmasi, Keelan Evanini, Aoife Cahill, Joel Tetreault, Robert Pugh, Christopher Hamill, Diane Napolitano, and Yao Qian. A report on the 2017 native language identification shared task. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 62–75, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-5007. URL `https://www.aclweb.org/anthology/W17-5007`.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. URL `https://www.aclweb.org/anthology/J93-2004`.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings*

of the 11th Annual Conference of the International Speech Communication Association, INTERSPEECH 2010, volume 2, pages 1045–1048, 01 2010.

Natawut Monaikul and Barbara Di Eugenio. Detecting preposition errors to target interlingual errors in second language writing. In *FLAIRS Conference*, pages 290–293, 2020.

Maria Nadejde and Joel Tetreault. Personalizing grammatical error correction: Adaptation to proficiency level and L1. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 27–33, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5504. URL `https://www.aclweb.org/anthology/D19-5504`.

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. URL `https://www.aclweb.org/anthology/W13-3601`.

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. doi: 10.3115/v1/W14-1701. URL `https://www.aclweb.org/anthology/W14-1701`.

Diane Nicholls. The Cambridge Learner Corpus: Error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics 2003 conference*, volume 16, pages 572–581, 2003.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo

Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://www.aclweb.org/anthology/L16-1262.

J. Michael O'Malley and Anna Uhl Chamot. *Strategies used by second language learners*, page 114–150. Cambridge Applied Linguistics. Cambridge University Press, 1990. doi: 10.1017/CBO9781139524490.007.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://www.aclweb.org/anthology/N18-1202.

Prokopis Prokopidis, Vassilis Papavassiliou, and Stelios Piperidis. Parallel global voices: a collection of multilingual corpora with citizen media stories. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 900–905, 2016.

Ella Rabinovich, Yulia Tsvetkov, and Shuly Wintner. Native language cognate effects on second language lexical choice. *Transactions of the Association for Computational Linguistics*, 6:329–342, 2018. doi: 10.1162/tacl_a_00024. URL https://www.aclweb.org/anthology/Q18-1024.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

Daniel Robertson. Variability in the use of the English article system by Chinese learners of English. *Second Language Research*, 16(2):135–172,

2000. doi: 10.1191/026765800672262975. URL https://doi.org/10.1191/026765800672262975.

Alla Rozovskaya and Dan Roth. Algorithm selection and model adaptation for ESL correction tasks. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 924–933, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P11-1093.

Larry Selinker. Interlanguage. 10(1-4):209–232, 1972. doi: doi:10.1515/iral.1972.10.1-4.209. URL https://doi.org/10.1515/iral.1972.10.1-4.209.

Younghee Sheen. The Effect of Focused Written Corrective Feedback and Language Aptitude on ESL Learners' Acquisition of Articles. *TESOL Quarterly*, 41(2):255–283, 2007. doi: https://doi.org/10.1002/j.1545-7249.2007.tb00059.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1545-7249.2007.tb00059.x.

Mariana Shimabukuro and Christopher Collins. Cross-linguistic word frequency visualization for pt and en. In *Proceedings of the IEEE Conference on Information Visualization (Poster)*, 2019.

Mariana Shimabukuro, Jessica Zipf, Mennatallah El-Assady, and Christopher Collins. H-matrix: Hierarchical matrix for visual analysis of cross-linguistic features in large learner corpora. In *Proceedings of the IEEE Conference on Information Visualization (short papers)*, 2019.

Natsuko Shintani and Rod Ellis. The comparative effect of direct written corrective feedback and metalinguistic explanation on learners' explicit and implicit knowledge of the English indefinite article. *Journal of Second Language Writing*, 22(3):286–306, 2013. ISSN 1060-3743. doi: https://doi.org/10.1016/j.jslw.2013.03.011. URL https://www.sciencedirect.com/science/article/pii/S1060374313000271.

Michael Swan and Bernard Smith. *Learner English: A Teacher's Guide to Interference and Other Problems*. Cambridge Handbooks for Language Teachers. Cambridge University Press, 2 edition, 2001. doi: 10.1017/CBO9780511667121.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 48–57, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W13-1706.

Jörg Tiedemann. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/463_Paper.pdf.

Michael Tomasello and Carol Herron. Feedback for Language Transfer Errors The Garden Path Technique. *Studies in Second Language Acquisition*, 11, 1989. ISSN 14701545. doi: 10.1017/S0272263100008408.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

David Alistair Watts. RAISING AWARENESS OF LANGUAGE TRANSFER WITH ACADEMIC ENGLISH WRITING STUDENTS AT A JAPANESE UNIVERSITY. *Frontier of Foreign Language Education*, 2:191–200, 2019. doi: https://doi.org/10.18910/71891.

Sze-Meng Jojo Wong and Mark Dras. Contrastive analysis and native language

identification. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, pages 53–61, Sydney, Australia, December 2009. URL https://www.aclweb.org/anthology/U09-1008.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238, 2005. doi: 10.1017/S135132490400364X.

Nianwen Xue and Yaqin Yang. Chinese sentence segmentation as comma classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 631–635, 2011.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P11-1019.

Appendix A

Negative language transfer dataset

The errors made by Chinese native speakers whose essays are part of the FCE dataset were manually annotated with information about their relation to negative language transfer. Each learner error in that subset of essays received a binary flag representing whether it was related to transfer or not. Along with this flag and regardless of the error's relation to negative language transfer, each error was annotated with a possible error cause.

The annotation process lasted 4 months and the annotations were performed by an English and Mandarin Chinese native speaker who teaches Chinese as a second language. The annotator is able to read and write in both languages and has higher proficiency in English writing. She speaks several dialects of Mandarin Chinese originating from South-East China and has taken linguistics courses on syntax.

In total, 3584 errors made by Chinese native speakers, whose essays are part of the FCE dataset, were annotated. Out of those, 1891 (52.76%) errors were tagged as negative language transfer related and 1389 (38.76%) were tagged as non-negative language transfer. The remaining 304 errors did not receive any tags as they were either spelling errors (292 errors), or they were omitted (12 errors) due to being annotated as errors because of language variety divergences or because the correction proposed by the FCE annotators was not enough to correct the mistake.

Among the labelled errors, a distinction can be made between errors that

received correction suggestions from the FCE annotator and errors which did not. Most errors in the FCE data are accompanied by suggested corrections. However, when the annotators were unsure about how to correct a learner error, they opted for not suggesting any edits. Out of the 3280 errors that received negative language transfer labels, 207 errors are not accompanied by annotator corrections.

During the annotation procedure the annotator had access to all the errors made by Chinese native speakers in FCE essays. For each error data point that needed annotation, the annotator could analyse the XML text corresponding to the error, a version of the learner's sentence in which the error was present, a version of the sentence in which the error was replaced by the FCE annotator's correction, and the error type according to the error coding described in Nicholls (2003). Using the error coding, it was possible to group learner errors according to their type. This procedure proved itself valuable as it allowed the annotator to focus on a few specific structures at a time.

Similarly, the XML error text, original sentence, and corrected version of the sentence in which the error happened were also useful during the annotation process. Through them the annotator was able to detect when the errors were semantic as opposed to structural. For example, the word "never" does not seem to be incorrect in the sentence "I have never been to in my life". However, it is tagged as such in the FCE dataset. Looking at the error's surroundings "It was the worst show and theatre I have never been to in my life", it is possible to see that the word "never" is incorrect and should be replaced by "ever".

During the annotation process, ambiguous cases such as the one illustrated above were discussed among the research team in weekly meetings. The annotator would bring to the meeting errors that caused her to be confused about the negative language transfer label they should receive. These errors were ambiguous with regard to their relation to the negative language transfer phenomenon. The errors discussed in these meetings fell under three major categories, English variety errors, error structures that do not have parallels in Chinese, and semantic errors. Table A.1 presents a few examples of these

ambiguous error categories.

	Incorrect utterance	**Correct utterance**	**Ambiguity**	**Number of cases**
British vs American English varieties	We *all would* like to go there.	We *would all* like to go there.	The incorrect version of the sentence is more commonly used in the American variety of English. It is not incorrect in that variety.	18
Chinese does not have an equivalent structure	I'm standing on your *left hand* side.	I'm standing on your *left-hand* side.	Hyphens do not have a parallel structure in Chinese.	17
Semantic errors tagged as structural errors	You *could* find a restaurant.	You *can* find a restaurant.	Although the verb "could" is in the past tense, some learners may choose to use it to indicate respect.	10

Table A.1: Ambiguous errors from the FCE dataset

All labelled errors in this dataset were also annotated with a possible cause for their occurrence. That is, along with the negative language transfer binary label, each error is accompanied by a short sentence that describes a possible reason for that error's occurrence. When the error in question is related to negative language transfer, its respective error cause annotation highlights the differences between English and Chinese. In the future, these error causes will be useful to inform the development of negative language transfer feedback for learners.

Appendix B

N-gram model hyperparameter tuning results

N-gram length	Evaluation accuracy
5	96.97%
4	96.95%
6	96.90%
3	96.34%
2	95.81%

Table B.1: N-gram model hyperparameter tuning results

Appendix C

RNN model hyperparameter tuning results

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	16	0.0001	1	95.16%
BCEwithLL	16	0.001	4	95.13%
NLLoss	16	0.0001	8	95.12%
BCEwithLL	8	0.001	8	95.12%
NLLoss	64	0.0001	4	95.11%
NLLoss	64	0.0001	2	95.11%
NLLoss	16	0.0001	4	95.11%
NLLoss	16	0.0001	2	95.1%
NLLoss	32	0.0001	4	95.1%
NLLoss	128	0.0001	16	95.1%
NLLoss	512	0.0001	4	95.1%
NLLoss	8	0.001	8	95.1%
NLLoss	32	0.001	8	95.09%
NLLoss	32	0.0001	8	95.09%
BCEwithLL	256	0.0001	2	95.09%
BCEwithLL	32	0.001	16	95.08%
BCEwithLL	16	0.001	16	95.08%
NLLoss	16	0.001	16	95.08%
NLLoss	64	0.0001	1	95.08%
NLLoss	8	0.001	16	95.08%
NLLoss	8	0.0001	8	95.08%
BCEwithLL	16	0.001	2	95.07%
BCEwithLL	64	0.0001	2	95.07%
NLLoss	32	0.0001	32	95.07%
BCEwithLL	64	0.001	4	95.07%
NLLoss	64	0.0001	8	95.07%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	32	0.0001	16	95.07%
NLLoss	8	0.0001	2	95.07%
NLLoss	8	0.0001	1	95.07%
BCEwithLL	16	0.0001	4	95.07%
NLLoss	64	0.001	16	95.07%
NLLoss	128	0.0001	4	95.07%
BCEwithLL	8	0.0001	1	95.06%
BCEwithLL	8	0.001	32	95.06%
BCEwithLL	16	0.001	1	95.06%
NLLoss	16	0.0001	16	95.06%
BCEwithLL	8	0.0001	4	95.06%
BCEwithLL	256	0.0001	16	95.05%
BCEwithLL	64	0.0001	1	95.05%
NLLoss	128	0.0001	2	95.05%
NLLoss	128	0.0001	32	95.05%
NLLoss	8	0.0001	4	95.05%
BCEwithLL	32	0.0001	1	95.05%
BCEwithLL	512	0.0001	16	95.05%
BCEwithLL	64	0.001	32	95.04%
NLLoss	256	1e-05	8	95.04%
NLLoss	512	0.0001	8	95.04%
BCEwithLL	32	0.0001	2	95.04%
BCEwithLL	16	0.0001	1	95.04%
BCEwithLL	32	0.001	32	95.04%
NLLoss	32	0.001	16	95.04%
BCEwithLL	64	0.0001	4	95.04%
NLLoss	128	1e-05	8	95.04%
BCEwithLL	32	0.0001	16	95.04%
NLLoss	512	0.0001	32	95.04%
NLLoss	8	0.001	32	95.04%
NLLoss	128	0.0001	1	95.04%
NLLoss	256	0.0001	1	95.04%
BCEwithLL	16	0.001	8	95.03%
BCEwithLL	16	0.0001	8	95.03%
NLLoss	128	0.0001	8	95.03%
NLLoss	8	0.001	1	95.03%
NLLoss	16	0.001	32	95.03%
BCEwithLL	128	0.0001	8	95.03%
NLLoss	64	1e-05	1	95.03%
NLLoss	32	0.0001	2	95.03%
BCEwithLL	256	0.0001	4	95.03%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	128	1e-05	1	95.03%
NLLoss	8	0.001	2	95.03%
NLLoss	16	0.0001	32	95.03%
NLLoss	32	1e-05	1	95.03%
NLLoss	8	0.0001	16	95.03%
BCEwithLL	128	0.0001	1	95.02%
NLLoss	64	1e-05	2	95.02%
BCEwithLL	8	0.0001	8	95.02%
BCEwithLL	256	1e-05	2	95.02%
BCEwithLL	32	0.0001	4	95.02%
BCEwithLL	64	0.0001	16	95.02%
NLLoss	32	1e-05	2	95.02%
NLLoss	512	0.0001	2	95.02%
BCEwithLL	128	0.0001	16	95.02%
BCEwithLL	128	1e-05	2	95.02%
BCEwithLL	512	1e-05	1	95.02%
BCEwithLL	8	0.0001	2	95.01%
BCEwithLL	8	0.001	4	95.01%
NLLoss	16	0.001	8	95.01%
BCEwithLL	128	1e-05	4	95.01%
NLLoss	16	0.001	4	95.01%
BCEwithLL	32	0.001	4	95.01%
BCEwithLL	16	0.0001	32	95.01%
NLLoss	32	0.001	4	95.01%
BCEwithLL	32	1e-05	1	95.01%
BCEwithLL	512	0.0001	32	95.01%
BCEwithLL	64	0.0001	32	95.01%
BCEwithLL	16	0.001	32	95.0%
BCEwithLL	128	0.0001	2	95.0%
NLLoss	128	1e-05	2	95.0%
BCEwithLL	256	1e-05	8	95.0%
BCEwithLL	32	0.001	1	95.0%
NLLoss	128	1e-05	16	95.0%
NLLoss	256	0.0001	16	95.0%
NLLoss	256	0.0001	32	95.0%
NLLoss	256	0.0001	2	95.0%
NLLoss	8	1e-05	1	95.0%
BCEwithLL	64	1e-05	4	94.99%
NLLoss	256	0.0001	4	94.99%
BCEwithLL	512	1e-05	2	94.99%
BCEwithLL	8	0.001	16	94.99%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	8	0.001	4	94.99%
BCEwithLL	32	0.0001	8	94.99%
BCEwithLL	32	0.0001	32	94.99%
BCEwithLL	128	0.0001	4	94.99%
BCEwithLL	64	1e-05	1	94.99%
NLLoss	512	1e-05	1	94.99%
BCEwithLL	128	1e-05	1	94.99%
BCEwithLL	16	0.0001	16	94.98%
NLLoss	16	1e-05	1	94.98%
NLLoss	256	1e-05	2	94.98%
BCEwithLL	8	0.001	1	94.98%
NLLoss	32	0.001	2	94.98%
NLLoss	16	1e-05	2	94.98%
NLLoss	8	0.0001	32	94.98%
BCEwithLL	16	1e-05	1	94.98%
NLLoss	256	0.0001	8	94.98%
NLLoss	512	1e-05	2	94.98%
NLLoss	32	0.001	32	94.98%
BCEwithLL	16	0.0001	2	94.98%
BCEwithLL	128	1e-05	8	94.98%
BCEwithLL	256	1e-05	16	94.98%
NLLoss	256	1e-05	1	94.97%
NLLoss	512	1e-05	16	94.97%
BCEwithLL	64	1e-05	2	94.97%
NLLoss	128	1e-05	4	94.97%
BCEwithLL	512	1e-05	8	94.97%
BCEwithLL	128	1e-05	16	94.97%
NLLoss	256	1e-05	32	94.97%
NLLoss	512	1e-05	32	94.97%
BCEwithLL	128	0.0001	32	94.97%
NLLoss	512	1e-05	8	94.97%
NLLoss	64	1e-05	8	94.96%
NLLoss	512	1e-06	1	94.96%
NLLoss	32	0.0001	1	94.96%
BCEwithLL	64	0.0001	8	94.96%
NLLoss	256	1e-05	16	94.95%
BCEwithLL	512	0.0001	1	94.95%
BCEwithLL	8	0.0001	32	94.95%
BCEwithLL	8	0.0001	16	94.95%
NLLoss	16	0.001	2	94.95%
NLLoss	64	1e-05	4	94.95%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
BCEwithLL	8	0.001	2	94.95%
BCEwithLL	64	1e-05	8	94.95%
BCEwithLL	256	1e-05	1	94.95%
BCEwithLL	64	0.001	8	94.95%
NLLoss	64	0.0001	16	94.95%
BCEwithLL	256	1e-05	4	94.95%
NLLoss	256	1e-05	4	94.95%
NLLoss	8	0.01	32	94.95%
NLLoss	512	0.0001	1	94.94%
BCEwithLL	32	1e-05	2	94.94%
BCEwithLL	512	1e-05	32	94.94%
BCEwithLL	16	1e-05	2	94.94%
NLLoss	32	1e-05	4	94.94%
BCEwithLL	512	0.0001	2	94.94%
NLLoss	512	0.0001	16	94.94%
NLLoss	64	0.001	32	94.93%
BCEwithLL	8	1e-05	1	94.93%
NLLoss	512	1e-05	4	94.93%
NLLoss	64	0.001	2	94.93%
NLLoss	256	1e-06	1	94.93%
NLLoss	128	1e-05	32	94.92%
BCEwithLL	256	0.0001	32	94.92%
BCEwithLL	512	1e-05	4	94.92%
BCEwithLL	32	1e-05	4	94.91%
BCEwithLL	32	0.001	2	94.91%
NLLoss	64	0.0001	32	94.91%
BCEwithLL	256	1e-05	32	94.9%
NLLoss	32	1e-05	8	94.9%
BCEwithLL	512	1e-05	16	94.9%
BCEwithLL	256	0.0001	8	94.9%
NLLoss	8	1e-05	2	94.9%
NLLoss	64	1e-05	16	94.89%
BCEwithLL	256	0.0001	1	94.89%
NLLoss	512	1e-06	2	94.89%
BCEwithLL	32	1e-05	8	94.88%
NLLoss	16	0.001	1	94.88%
BCEwithLL	32	0.001	8	94.87%
BCEwithLL	8	0.01	32	94.86%
NLLoss	32	1e-05	16	94.86%
BCEwithLL	512	0.0001	4	94.85%
NLLoss	16	1e-05	8	94.85%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	16	1e-05	4	94.84%
BCEwithLL	128	1e-05	32	94.82%
BCEwithLL	64	1e-05	16	94.81%
NLLoss	512	1e-06	4	94.78%
BCEwithLL	256	1e-06	1	94.77%
BCEwithLL	512	1e-06	2	94.77%
NLLoss	64	1e-05	32	94.75%
NLLoss	128	1e-06	1	94.75%
BCEwithLL	16	1e-05	4	94.74%
BCEwithLL	16	1e-05	8	94.72%
BCEwithLL	8	0.01	16	94.7%
BCEwithLL	8	1e-05	2	94.69%
NLLoss	8	0.01	16	94.69%
NLLoss	8	1e-05	4	94.69%
BCEwithLL	512	1e-06	1	94.65%
NLLoss	16	1e-05	16	94.65%
NLLoss	32	1e-05	32	94.65%
NLLoss	256	1e-06	2	94.62%
BCEwithLL	128	1e-06	1	94.59%
NLLoss	512	1e-06	8	94.57%
BCEwithLL	8	1e-05	4	94.57%
BCEwithLL	32	1e-05	16	94.57%
BCEwithLL	64	1e-05	32	94.56%
NLLoss	16	1e-05	32	94.56%
NLLoss	128	1e-06	2	94.56%
NLLoss	8	1e-05	8	94.56%
NLLoss	64	1e-06	1	94.55%
BCEwithLL	256	1e-06	2	94.55%
BCEwithLL	16	1e-05	16	94.54%
NLLoss	256	1e-06	4	94.54%
BCEwithLL	512	1e-06	4	94.5%
NLLoss	32	1e-06	1	94.5%
BCEwithLL	512	0.0001	8	94.5%
BCEwithLL	32	1e-05	32	94.49%
NLLoss	32	0.001	1	94.48%
BCEwithLL	64	1e-06	1	94.44%
NLLoss	64	1e-06	2	94.43%
BCEwithLL	128	1e-06	2	94.42%
NLLoss	128	1e-06	4	94.4%
NLLoss	256	1e-06	8	94.36%
BCEwithLL	256	1e-06	4	94.34%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	512	1e-06	16	94.33%
BCEwithLL	8	1e-05	8	94.31%
BCEwithLL	32	1e-06	1	94.3%
NLLoss	64	1e-06	4	94.29%
BCEwithLL	512	1e-06	8	94.28%
BCEwithLL	64	1e-06	2	94.26%
NLLoss	32	1e-06	2	94.25%
BCEwithLL	16	1e-05	32	94.25%
NLLoss	16	1e-06	1	94.24%
NLLoss	8	1e-05	16	94.23%
NLLoss	128	1e-06	8	94.12%
NLLoss	256	1e-06	16	94.11%
BCEwithLL	128	1e-06	4	94.09%
NLLoss	8	1e-05	32	94.07%
BCEwithLL	8	1e-05	16	94.04%
BCEwithLL	256	1e-06	8	94.03%
NLLoss	8	1e-06	1	94.0%
NLLoss	512	1e-06	32	93.99%
NLLoss	8	0.01	4	93.98%
BCEwithLL	16	1e-06	1	93.98%
NLLoss	64	1e-06	8	93.97%
BCEwithLL	512	1e-06	16	93.97%
BCEwithLL	32	1e-06	2	93.93%
NLLoss	16	1e-06	2	93.93%
NLLoss	32	1e-06	4	93.91%
BCEwithLL	64	1e-06	4	93.85%
NLLoss	128	1e-06	16	93.84%
BCEwithLL	128	1e-06	8	93.75%
BCEwithLL	16	1e-06	2	93.64%
BCEwithLL	8	1e-06	1	93.62%
NLLoss	256	1e-06	32	93.6%
BCEwithLL	8	1e-05	32	93.51%
BCEwithLL	512	1e-06	32	93.5%
BCEwithLL	32	1e-06	4	93.33%
BCEwithLL	256	1e-06	16	93.33%
NLLoss	8	1e-06	2	93.29%
NLLoss	32	1e-06	8	93.22%
NLLoss	16	1e-06	4	93.22%
NLLoss	64	1e-06	16	93.12%
BCEwithLL	64	1e-06	8	92.99%
NLLoss	128	1e-06	32	92.96%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	8	1e-06	4	92.9%
BCEwithLL	8	1e-06	2	92.8%
BCEwithLL	128	1e-06	16	92.78%
NLLoss	64	0.001	8	92.51%
BCEwithLL	16	1e-06	4	92.42%
BCEwithLL	64	1e-06	16	92.3%
BCEwithLL	32	1e-06	8	92.29%
NLLoss	32	1e-06	16	92.29%
NLLoss	64	1e-06	32	92.15%
BCEwithLL	256	1e-06	32	92.11%
BCEwithLL	128	1e-06	32	91.84%
BCEwithLL	16	1e-06	8	91.78%
NLLoss	16	1e-06	8	91.62%
NLLoss	16	1e-06	16	91.4%
BCEwithLL	8	1e-06	4	91.36%
NLLoss	32	1e-06	32	91.18%
BCEwithLL	64	1e-06	32	91.07%
NLLoss	8	1e-06	8	89.84%
BCEwithLL	32	1e-06	16	89.55%
BCEwithLL	32	1e-06	32	89.38%
BCEwithLL	8	1e-06	8	89.28%
BCEwithLL	16	1e-06	16	88.99%
NLLoss	8	1e-06	16	82.77%
BCEwithLL	16	1e-06	32	81.71%
NLLoss	16	1e-06	32	81.64%
BCEwithLL	8	1e-06	16	79.66%
BCEwithLL	8	1e-06	32	67.29%
NLLoss	8	1e-06	32	63.75%
NLLoss	64	0.001	4	57.74%
BCEwithLL	8	0.01	8	51.76%
BCEwithLL	16	0.01	32	50.0%
BCEwithLL	64	0.01	32	50.0%
BCEwithLL	32	0.01	8	50.0%
BCEwithLL	32	0.01	32	50.0%
BCEwithLL	32	0.01	16	50.0%
BCEwithLL	32	0.01	4	50.0%
BCEwithLL	16	0.01	8	50.0%
BCEwithLL	16	0.01	4	50.0%
BCEwithLL	64	0.01	16	50.0%
BCEwithLL	64	0.01	8	50.0%
BCEwithLL	32	0.01	2	50.0%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
BCEwithLL	16	0.01	2	50.0%
BCEwithLL	64	0.01	4	50.0%
BCEwithLL	64	0.01	2	50.0%
NLLoss	32	0.01	8	50.0%
BCEwithLL	64	0.001	16	50.0%
NLLoss	32	0.01	32	50.0%
NLLoss	16	0.01	16	50.0%
NLLoss	16	0.01	8	50.0%
BCEwithLL	128	0.01	16	50.0%
BCEwithLL	128	0.01	32	50.0%
BCEwithLL	128	0.01	8	50.0%
NLLoss	32	0.01	16	50.0%
NLLoss	32	0.01	4	50.0%
NLLoss	16	0.01	32	50.0%
NLLoss	64	0.01	32	50.0%
BCEwithLL	128	0.01	4	50.0%
NLLoss	32	0.01	2	50.0%
BCEwithLL	16	0.01	1	50.0%
BCEwithLL	128	0.01	2	50.0%
NLLoss	16	0.01	4	50.0%
NLLoss	64	0.01	8	50.0%
NLLoss	16	0.01	2	50.0%
BCEwithLL	32	0.01	1	50.0%
NLLoss	64	0.01	16	50.0%
BCEwithLL	64	0.01	1	50.0%
BCEwithLL	64	0.001	2	50.0%
BCEwithLL	128	0.001	32	50.0%
BCEwithLL	16	0.01	16	50.0%
NLLoss	64	0.01	4	50.0%
BCEwithLL	128	0.001	16	50.0%
NLLoss	128	0.01	32	50.0%
BCEwithLL	128	0.001	4	50.0%
NLLoss	64	0.01	2	50.0%
BCEwithLL	128	0.001	8	50.0%
BCEwithLL	128	0.001	2	50.0%
NLLoss	32	0.01	1	50.0%
NLLoss	128	0.01	8	50.0%
NLLoss	128	0.01	16	50.0%
NLLoss	16	0.01	1	50.0%
BCEwithLL	128	0.01	1	50.0%
BCEwithLL	256	0.01	32	50.0%

Loss function	Number of hidden units	Learning rate	Mini batch size	Evaluation accuracy
NLLoss	128	0.01	4	50.0%
NLLoss	128	0.001	8	50.0%
NLLoss	128	0.01	2	50.0%
BCEwithLL	64	0.001	1	50.0%
BCEwithLL	256	0.01	16	50.0%
NLLoss	128	0.001	4	50.0%
NLLoss	64	0.01	1	50.0%
BCEwithLL	256	0.01	8	50.0%
NLLoss	128	0.001	32	50.0%
NLLoss	128	0.001	16	50.0%
BCEwithLL	128	0.001	1	50.0%
NLLoss	64	0.001	1	50.0%
BCEwithLL	256	0.01	4	50.0%
NLLoss	128	0.001	2	50.0%
NLLoss	256	0.01	16	50.0%
BCEwithLL	256	0.001	8	50.0%
NLLoss	128	0.01	1	50.0%
BCEwithLL	256	0.001	4	50.0%
BCEwithLL	256	0.001	16	50.0%
NLLoss	256	0.01	32	50.0%
BCEwithLL	256	0.01	2	50.0%
NLLoss	256	0.01	4	50.0%
BCEwithLL	256	0.001	32	50.0%
NLLoss	256	0.01	8	50.0%
NLLoss	256	0.01	2	50.0%
NLLoss	256	0.001	32	50.0%
BCEwithLL	256	0.001	2	50.0%
NLLoss	128	0.001	1	50.0%
BCEwithLL	256	0.01	1	50.0%
NLLoss	256	0.001	16	50.0%
NLLoss	256	0.001	4	50.0%
NLLoss	256	0.001	8	50.0%
NLLoss	256	0.001	2	50.0%
NLLoss	256	0.01	1	50.0%
BCEwithLL	256	0.001	1	50.0%
NLLoss	256	0.001	1	50.0%
BCEwithLL	512	0.01	16	50.0%
BCEwithLL	512	0.01	32	50.0%
BCEwithLL	512	0.01	8	50.0%
BCEwithLL	512	0.001	16	50.0%
NLLoss	512	0.01	16	50.0%

Loss function	**Number of hidden units**	**Learning rate**	**Mini batch size**	**Evaluation accuracy**
BCEwithLL	512	0.001	32	50.0%
NLLoss	512	0.01	32	50.0%
BCEwithLL	512	0.001	8	50.0%
NLLoss	512	0.001	32	50.0%
NLLoss	512	0.01	8	50.0%
NLLoss	512	0.001	16	50.0%
NLLoss	512	0.01	4	50.0%
NLLoss	512	0.001	8	50.0%
BCEwithLL	512	0.001	4	50.0%
BCEwithLL	512	0.01	2	50.0%
NLLoss	512	0.001	4	50.0%
BCEwithLL	512	0.001	2	50.0%
NLLoss	512	0.01	2	50.0%
BCEwithLL	512	0.01	4	50.0%
NLLoss	512	0.001	2	50.0%
BCEwithLL	512	0.01	1	50.0%
NLLoss	512	0.01	1	50.0%
BCEwithLL	512	0.001	1	50.0%
NLLoss	512	0.001	1	50.0%
BCEwithLL	8	0.01	4	50.0%
BCEwithLL	8	0.01	2	50.0%
NLLoss	8	0.01	8	50.0%
BCEwithLL	8	0.01	1	50.0%
NLLoss	8	0.01	1	50.0%
NLLoss	8	0.01	2	45.19%

Table C.1: RNN model hyperparameter tuning results

Appendix D

N-gram model results across all structural error types

Error type	Span	Precision	Recall	F1-score
Replace punctuation (n = 336)	Padded error	0.65	0.3	0.41
	Error + unigram	**0.69**	**0.43**	**0.53**
	Error + bigram	0.62	0.23	0.34
Incorrect tense of verb (n = 267)	Padded error	0.7	0.21	0.32
	Error + unigram	**0.78**	**0.36**	**0.49**
	Error + bigram	0.73	0.22	0.33
Missing determiner (n = 209)	Padded error	**1.0**	**0.43**	**0.6**
	Error + unigram	0.97	0.33	0.5
	Error + bigram	**1.0**	0.35	0.52
Missing punctuation (n = 152)	Padded error	**0.88**	**0.37**	**0.52**
	Error + unigram	0.81	0.18	0.3
	Error + bigram	0.85	0.24	0.37
Wrong verb form (n = 132)	Padded error	0.41	**0.28**	**0.33**
	Error + unigram	**0.42**	**0.28**	**0.33**
	Error + bigram	**0.42**	0.19	0.26
Wrong noun form (n = 93)	Padded error	**0.84**	0.42	0.56
	Error + unigram	0.73	**0.6**	**0.66**
	Error + bigram	0.78	0.44	0.56
Verb agreement error (n = 88)	Padded error	**0.81**	0.24	0.37
	Error + unigram	0.75	**0.28**	**0.41**
	Error + bigram	0.8	0.15	0.25
Unnecessary determiner (n = 75)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Missing preposition (n = 75)	Padded error	**0.93**	**0.4**	**0.56**
	Error + unigram	0.76	0.28	0.41
	Error + bigram	0.78	0.27	0.4

Error type	Span	Precision	Recall	F1-score
Unnecessary punctuation (n = 65)	Padded error	0.15	0.67	0.25
	Error + unigram	**0.17**	**0.83**	**0.29**
	Error + bigram	0.05	0.17	0.08
Noun agreement error (n = 62)	Padded error	0.57	0.2	0.29
	Error + unigram	**0.58**	**0.34**	**0.43**
	Error + bigram	**0.58**	0.27	0.37
Missing anaphor (n = 62)	Padded error	0.68	**0.32**	**0.43**
	Error + unigram	0.5	0.12	0.2
	Error + bigram	**0.73**	0.2	0.31
Unnecessary preposition (n = 59)	Padded error	**0.22**	0.29	0.25
	Error + unigram	0.2	**0.57**	**0.3**
	Error + bigram	0.2	0.43	0.27
Missing error (n = 57)	Padded error	0.63	**0.3**	**0.41**
	Error + unigram	0.67	**0.3**	**0.41**
	Error + bigram	**0.69**	0.28	0.39
Missing verb (n = 54)	Padded error	0.73	**0.23**	**0.35**
	Error + unigram	0.71	0.21	0.33
	Error + bigram	**0.89**	0.17	0.29
Derivation of adjective error (n = 47)	Padded error	**1.0**	0.51	0.68
	Error + unigram	0.89	**0.59**	**0.71**
	Error + bigram	0.81	0.41	0.55
Word order error (n = 44)	Padded error	0.67	0.17	0.27
	Error + unigram	0.7	**0.29**	**0.41**
	Error + bigram	**0.83**	0.21	0.33
Unnecessary verb (n = 44)	Padded error	0.62	0.31	0.42
	Error + unigram	**0.79**	**0.69**	**0.73**
	Error + bigram	0.62	0.31	0.42
Replace determiner (n = 42)	Padded error	**0.75**	0.17	0.27
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.67	**0.22**	**0.33**
Replace anaphor (n = 33)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Derivation of noun error (n = 29)	Padded error	0.5	0.1	0.17
	Error + unigram	0.6	**0.3**	0.4
	Error + bigram	**0.75**	**0.3**	**0.43**
Unnecessary error (n = 25)	Padded error	**1.0**	0.12	0.22
	Error + unigram	0.25	**0.25**	0.25
	Error + bigram	0.33	**0.25**	**0.29**
Derivation of anaphor error (n = 21)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**0.5**	**1.0**	**0.67**

Error type	Span	Precision	Recall	F1-score
Unnecessary anaphor	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
(n = 20)	Error + bigram	0.0	0.0	0.0
Missing link word (n = 20)	Padded error	**1.0**	**0.53**	**0.69**
	Error + unigram	**1.0**	0.47	0.64
	Error + bigram	**1.0**	0.41	0.58
Derivation of adverb error (n = 19)	Padded error	0.29	0.25	0.27
	Error + unigram	0.36	**0.5**	**0.42**
	Error + bigram	**0.4**	0.25	0.31
Anaphor agreement error (n = 19)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Unnecessary adverb (n = 18)	Padded error	**0.33**	**0.5**	**0.4**
	Error + unigram	0.22	**0.5**	0.31
	Error + bigram	0.2	0.25	0.22
Argument structure error (n = 17)	Padded error	**0.5**	**0.17**	**0.25**
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Unnecessary link word (n = 14)	Padded error	0.2	**1.0**	0.33
	Error + unigram	**0.33**	**1.0**	**0.5**
	Error + bigram	0.25	**1.0**	0.4
Replace quantifier (n = 14)	Padded error	**1.0**	**0.29**	**0.44**
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.5	0.14	0.22
Missing noun (n = 14)	Padded error	0.5	0.17	0.25
	Error + unigram	**0.67**	**0.33**	**0.44**
	Error + bigram	0.4	**0.33**	0.36
Missing adverb (n = 13)	Padded error	**1.0**	0.25	0.4
	Error + unigram	0.83	**0.42**	**0.56**
	Error + bigram	**1.0**	0.08	0.15
Incorrect verb inflection (n = 13)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Incorrect determiner form (n = 13)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Derivation of determiner error (n = 12)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Countability of noun error (n = 12)	Padded error	0.67	0.4	0.5
	Error + unigram	0.83	0.5	0.62
	Error + bigram	**1.0**	**0.6**	**0.75**

Error type	Span	Precision	Recall	F1-score
Unnecessary noun (n = 9)	Padded error	0.83	0.62	0.71
	Error + unigram	0.88	**0.88**	**0.88**
	Error + bigram	**1.0**	0.5	0.67
Incorrect noun inflection (n = 9)	Padded error	**1.0**	0.6	0.75
	Error + unigram	0.62	**1.0**	**0.77**
	Error + bigram	0.67	0.8	0.73
Inappropriate register (n = 9)	Padded error	**1.0**	**0.4**	**0.57**
	Error + unigram	**1.0**	0.2	0.33
	Error + bigram	0.5	0.2	0.29
Incorrect negative formation (n = 8)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**1.0**	**0.25**	**0.4**
Quantifier agreement error (n = 7)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**1.0**	**0.14**	**0.25**
Unnecessary adjective (n = 6)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Derivation of verb error (n = 5)	Padded error	**1.0**	0.4	0.57
	Error + unigram	**1.0**	**0.6**	**0.75**
	Error + bigram	**1.0**	0.4	0.57
Wrong quantifier because of noun countability (n = 4)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Wrong adjective form (n = 4)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**1.0**	**0.33**	**0.5**
Missing adjective (n = 4)	Padded error	**1.0**	**1.0**	**1.0**
	Error + unigram	0.5	0.5	0.5
	Error + bigram	**1.0**	0.5	0.67
Determiner agreement error (n = 4)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Unnecessary quantifier (n = 3)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Missing quantifier (n = 3)	Padded error	**0.5**	**0.5**	**0.5**
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Wrong anaphor form (n = 2)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0

Error type	**Span**	**Precision**	**Recall**	**F1-score**
Incorrect adjective inflection (n = 2)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Wrong determiner because of noun countability (n = 1)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Wrong adverb form (n = 1)	Padded error	**1.0**	**1.0**	**1.0**
	Error + unigram	**1.0**	**1.0**	**1.0**
	Error + bigram	**1.0**	**1.0**	**1.0**
Incorrect quantifier inflection (n = 1)	Padded error	**1.0**	**1.0**	**1.0**
	Error + unigram	**1.0**	**1.0**	**1.0**
	Error + bigram	0.0	0.0	0.0

Table D.1: N-gram model results across all structural error types

Appendix E

RNN model results across all structural error types

Error type	Span	Precision	Recall	F1-score
Replace punctuation (n = 336)	Padded error	**0.71**	0.39	0.51
	Error + unigram	0.7	**0.47**	**0.56**
	Error + bigram	**0.71**	0.38	0.5
Incorrect tense of verb (n = 267)	Padded error	0.78	0.39	0.52
	Error + unigram	0.77	**0.41**	**0.54**
	Error + bigram	**0.79**	0.29	0.42
Missing determiner (n = 209)	Padded error	**1.0**	0.26	0.41
	Error + unigram	0.98	**0.45**	**0.61**
	Error + bigram	0.98	0.4	0.57
Missing punctuation (n = 152)	Padded error	**0.88**	**0.33**	**0.48**
	Error + unigram	0.85	0.29	0.43
	Error + bigram	0.84	0.27	0.41
Wrong verb form (n = 132)	Padded error	**0.41**	**0.38**	**0.39**
	Error + unigram	0.3	0.28	0.29
	Error + bigram	0.31	0.21	0.25
Wrong noun form (n = 93)	Padded error	**0.79**	0.56	0.66
	Error + unigram	0.77	**0.78**	**0.78**
	Error + bigram	0.77	0.56	0.65
Verb agreement error (n = 88)	Padded error	0.74	**0.37**	**0.49**
	Error + unigram	0.77	0.31	0.45
	Error + bigram	**0.92**	0.22	0.36
Unnecessary determiner (n = 75)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Missing preposition (n = 75)	Padded error	**0.9**	0.27	0.41
	Error + unigram	0.79	0.28	0.42
	Error + bigram	0.8	**0.36**	**0.49**

Error type	Span	Precision	Recall	F1-score
Unnecessary punctuation (n = 65)	Padded error	**0.17**	**0.5**	**0.25**
	Error + unigram	0.09	0.33	0.14
	Error + bigram	0.08	0.17	0.11
Noun agreement error (n = 62)	Padded error	**0.63**	0.29	0.4
	Error + unigram	0.59	**0.41**	0.49
	Error + bigram	**0.63**	**0.41**	**0.5**
Missing anaphor (n = 62)	Padded error	0.71	**0.41**	**0.52**
	Error + unigram	0.62	0.24	0.35
	Error + bigram	**0.8**	0.2	0.31
Unnecessary preposition (n = 59)	Padded error	0.1	0.14	0.12
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**0.15**	**0.29**	**0.2**
Missing error (n = 57)	Padded error	**0.68**	0.32	0.44
	Error + unigram	0.65	**0.38**	**0.48**
	Error + bigram	**0.68**	0.32	0.44
Missing verb (n = 54)	Padded error	0.77	0.21	0.33
	Error + unigram	**0.82**	**0.3**	**0.44**
	Error + bigram	0.78	**0.3**	0.43
Derivation of adjective error (n = 47)	Padded error	**0.92**	0.56	0.7
	Error + unigram	0.91	**0.76**	**0.83**
	Error + bigram	0.85	0.56	0.68
Word order error (n = 44)	Padded error	0.56	0.21	0.3
	Error + unigram	0.5	0.21	0.29
	Error + bigram	**0.7**	**0.29**	**0.41**
Unnecessary verb (n = 44)	Padded error	**0.73**	**0.69**	**0.71**
	Error + unigram	0.69	**0.69**	0.69
	Error + bigram	0.71	0.62	0.67
Replace determiner (n = 42)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**0.5**	**0.11**	**0.18**
Replace anaphor (n = 33)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Derivation of noun error (n = 29)	Padded error	0.5	0.2	0.29
	Error + unigram	0.67	**0.5**	**0.57**
	Error + bigram	**0.78**	0.35	0.48
Unnecessary error (n = 25)	Padded error	**0.4**	**0.25**	**0.31**
	Error + unigram	0.25	0.12	0.17
	Error + bigram	**0.4**	**0.25**	**0.31**
Derivation of anaphor error (n = 21)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**1.0**	**1.0**	**1.0**

Error type	Span	Precision	Recall	F1-score
Unnecessary anaphor	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
(n = 20)	Error + bigram	0.0	0.0	0.0
Missing link word (n = 20)	Padded error	**1.0**	0.47	0.64
	Error + unigram	0.9	**0.53**	**0.67**
	Error + bigram	**1.0**	0.47	0.64
Derivation of adverb error (n = 19)	Padded error	0.29	0.25	0.27
	Error + unigram	0.38	**0.38**	0.38
	Error + bigram	**0.43**	**0.38**	**0.4**
Anaphor agreement error (n = 19)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Unnecessary adverb (n = 18)	Padded error	**1.0**	**0.5**	**0.67**
	Error + unigram	0.4	**0.5**	0.44
	Error + bigram	0.5	**0.5**	0.5
Argument structure error (n = 17)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**0.67**	**0.33**	**0.44**
Unnecessary link word (n = 14)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**0.2**	**1.0**	**0.33**
Replace quantifier (n = 14)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**0.75**	**0.43**	**0.55**
Missing noun (n = 14)	Padded error	0.33	0.17	0.22
	Error + unigram	0.33	0.17	0.22
	Error + bigram	**0.75**	**0.5**	**0.6**
Missing adverb (n = 13)	Padded error	**1.0**	0.5	0.67
	Error + unigram	**1.0**	**0.58**	**0.74**
	Error + bigram	0.83	0.42	0.56
Incorrect verb inflection (n = 13)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Incorrect determiner form (n = 13)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Derivation of determiner error (n = 12)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Countability of noun error (n = 12)	Padded error	**1.0**	**0.6**	**0.75**
	Error + unigram	0.75	**0.6**	0.67
	Error + bigram	0.83	0.5	0.62

Error type	Span	Precision	Recall	F1-score
Unnecessary noun (n = 9)	Padded error	0.86	0.75	0.8
	Error + unigram	0.89	**1.0**	**0.94**
	Error + bigram	**1.0**	0.62	0.77
Incorrect noun inflection (n = 9)	Padded error	**1.0**	0.4	0.57
	Error + unigram	0.62	**1.0**	0.77
	Error + bigram	0.8	0.8	**0.8**
Inappropriate register (n = 9)	Padded error	**1.0**	0.2	0.33
	Error + unigram	0.67	**0.4**	**0.5**
	Error + bigram	**1.0**	0.2	0.33
Incorrect negative formation (n = 8)	Padded error	**0.5**	**0.25**	**0.33**
	Error + unigram	**0.5**	**0.25**	**0.33**
	Error + bigram	**0.5**	**0.25**	**0.33**
Quantifier agreement error (n = 7)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**1.0**	**0.14**	**0.25**
Unnecessary adjective (n = 6)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Derivation of verb error (n = 5)	Padded error	**1.0**	0.4	0.57
	Error + unigram	**1.0**	**0.6**	**0.75**
	Error + bigram	**1.0**	0.4	0.57
Wrong quantifier because of noun countability (n = 4)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**1.0**	**0.67**	**0.8**
Wrong adjective form (n = 4)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	**1.0**	**0.33**	**0.5**
Missing adjective (n = 4)	Padded error	0.0	0.0	0.0
	Error + unigram	**1.0**	0.5	0.67
	Error + bigram	**1.0**	**1.0**	**1.0**
Determiner agreement error (n = 4)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Unnecessary quantifier (n = 3)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Missing quantifier (n = 3)	Padded error	0.0	0.0	0.0
	Error + unigram	**1.0**	**1.0**	**1.0**
	Error + bigram	**1.0**	0.5	0.67
Wrong anaphor form (n = 2)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0

Error type	**Span**	**Precision**	**Recall**	**F1-score**
Incorrect adjective inflection (n = 2)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Wrong determiner because of noun countability (n = 1)	Padded error	0.0	0.0	0.0
	Error + unigram	0.0	0.0	0.0
	Error + bigram	0.0	0.0	0.0
Wrong adverb form (n = 1)	Padded error	**1.0**	**1.0**	**1.0**
	Error + unigram	**1.0**	**1.0**	**1.0**
	Error + bigram	0.0	0.0	0.0
Incorrect quantifier inflection (n = 1)	Padded error	0.0	0.0	0.0
	Error + unigram	**1.0**	**1.0**	**1.0**
	Error + bigram	0.0	0.0	0.0

Table E.1: RNN model results across all structural error types