SAARLAND UNIVERSITY

DEPARTMENT OF COMPUTATIONAL LINGUISTICS

MASTER THESIS

Artificial Error Generation for Grammatical Error Correction

Author:

Fraser BOWEN

Supervisors:

Prof. Dr. Josef VAN GENABITH

Dr. Jonathan DEHDARI

September 2017

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Declaration

I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged.

Saarbrücken, Germany, _____ _____

Fraser Bowen Date

Universität des Saarlandes

Abstract

Department of Computational Linguistics

Master of Science

Artificial Error Generation for Grammatical Error Correction

by Fraser BOWEN

We investigate using Neural Machine Translation (NMT) for Grammatical Error Correction (GEC). By artificially introducing noise into monolingual language data, an "error corpus" is generated, which is used as the input into an encoder–decoder network. By not using expensive and arguably unreliable manually annotated data, we expect to offer a more consistent and flexible approach to GEC. We use revision data from the Lang-8 revision corpus to generate confusion sets, which inform the probabilistic approach to generating erroneous data. Two separate approaches are taken, one which focuses on individual error types, and one which tackles all error types at once, both with a focus on the effect of error density in the training data.

Acknowledgements

I would like to thank my two supervisors, firstly Jon Dehdari for his unending patience, and his continued support despite moving to the US half way through the thesis. Secondly, Josef van Genabith for his help and faith in the project.

Thank you to Daniel Naber of LanguageTool, who sponsored this thesis, held regular telephone meetings and gave me a fascinating task to work with.

Thank you to Mamoru Komachi for allowing me access to the Lang-8 Revision Corpora. Also to Joel Tetreault who kindly offered his advice as an expert in the field.

Finally, I have my friends and office-mates Stalin and Nima to thank who have been there with me every step of the way. They have answered all my silly questions on implementation, and have offered insightful ideas for the direction of my thesis, as well as the occasional sanity check.

Contents

Chapter 1

Introduction

In a world where there are more learners of English than native English speakers, having the ability to automatically correct learner English is invaluable. Although teaching and manual input is essential to the learning of any language, Artificial Intelligence (AI) can also be helpful to learners by giving an indication of possible corrections to an erroneous sentence. In the vast majority of cases in which a non-native speaker's English must be proofread, there is no access to a native speaker for correction, despite the growing demand for well-formed grammatical English, especially in the workplace.

However, it is not just English which requires correction. The vast majority of research in Language Technology has focused on the English language, but this is not indicative of the demand for Language Technology across all languages. Likewise, AI solutions to Grammatical Error Correction (GEC) have been solely for English at the time of writing. This is partly due to the nature of the errors made in different languages. It is difficult to approach the GEC problem in a language-independent fashion when each language is so structurally different. It is also partly related to the amount of data available in each language. Machine Learning requires data, and English has the largest amount of data, as well as the cheapest and most accessible data. On top of that, error revision data, in which erroneous source sentences are corrected into well-formed target sentences, is very difficult and expensive to procure.

The research conducted in this thesis examines the potential for a transition in GEC

from language-specific approaches to a language-independent approach. One promising option for tackling the variation between languages and a lack of data for resources other than English is Automatic Error Generation (AEG). This aims to generate training data for AI by introducing error into a corpus of already high-quality language. Since correct, monolingual data is easy and cheap to come by, even in low resource languages, this approach could in theory allow it to be possible to train a model for GEC in any language. Not only this, but there is a large potential for domain adaptation, since there is a lot of control over the training data. The data generating using AEG could then be used as data on its own, or be used to adapt a model trained on supervised data.

1.1 Grammatical Error Correction Today

This research is sponsored by the open-source grammar checker, LanguageTool[1]. It is the largest free GEC service available, and it offers rule-based corrections for over 20 languages.

LanguageTool is one of many grammar checkers today to use a rule-based system to process text. Although much research has gone into more advanced methods of correcting grammar, none have proved so commercially viable as an extensive rule-based system. LanguageTool is considered to be the best of its kind due to its large base of contributors, who are helping correct many different languages one error at a time.

In recent years, however, using Neural Machine Translation (NMT) to correct English has begun to appear as a commercially viable product, with paid services such as Grammarly and DeepGrammar already investing in neural network systems.

This thesis intends to explore the potential of using NMT for LanguageTool. Its coverage of several different languages is the motivation for the search for a language-independent approach to the GEC problem.

[1] https://www.languagetool.org/

1.2 How much error to introduce?

The largest gap in knowledge when it comes to AEG is the question of how much error to introduce into the training corpus. There are several variable factors which could affect this, e.g.

- Does an NMT system correct too many or not enough real, human errors?
- What is the interaction between the amount of error (error density) in the training data and the amount of errors the trained model corrects?
- How many errors should be corrected?
- How erroneous is the language of a non-native speaker?

This thesis aims to answer these research questions by running experiments based on two different approaches to AEG.

Firstly, specific error types are tackled individually. This has been a popular approach in the last ten years, because the systems have had much success dealing with those errors in isolated instances. Since LanguageTool aims to tackle all errors at once, this research aims to add to the growing amount of research on enhancing rule-based systems by tackling some of the more 'difficult' error types. An analysis on the feasibility of this, as well as combining several types of error together is carried out in tandem with experiments on models designed to correct prepositions and determiners. In such clear-cut examples, insight into the interaction of the amount of error in the training data with the amount of corrected errors can be gained.

Secondly, all error types are tackled at once. This is an almost entirely unexplored approach in AEG, but a very promising one. Such a system could in theory be used as a replacement to a rule-based system, although using a method which is capable of generalising across all error types will never be as precise as tackling individual errors. Nonetheless, the same relationship between the amount of error introduced into the training data and the number of corrected errors will be analysed and compared with the single-error approach.

This thesis' contribution to the field is twofold:

- A proposed phrase-based approach to introducing error using replacements, insertions and deletions taken from a background corpus.
- An analysis of the effect of error density in an artificially generated corpus on the amount of errors corrected by a model trained on that corpus.

1.3 Structure of the Thesis

The thesis is organised as follows. Chapter two introduces the reader to the field of Grammatical Error Correction by introducing firstly the various shared tasks which have sparked interest in the topic over the last five years, and then exploring the various data sets used in research, publicly available and not. Then, it explores how research has moved from rule-based approaches, through Statistical Machine Translation approaches and finally towards neural approaches. Finally, it details the few but diverse approaches to Automatic Error Generation, from random word replacement, to using Machine Translation.

Chapter three covers evaluation, which is very important in this field, as the standard metrics and test sets are arguably under-developed. Fortunately, recent research has attempted to improve the way in which GEC is evaluated. This section will detail the latest metrics and test sets used in this work, as well as previous ones considered to be benchmark tests against which it is possible to compare results.

Chapter four is devoted to experimental setup. The NMT system used in the project is described, and then the preliminary experiments are introduced. There are three preliminary experiments, all of which aim to get an idea of the baselines needed when introducing error into an error-free corpus.

Chapter five introduces the first approach to AEG, namely tackling individual error types. The chapter is split into a methodology, an explanation of the experiments, and a discussion of results.

Chapter six outlines the experiments on the second approach, which aim to correct all errors simultaneously. Here it is explained how the error introduction algorithm is adapted to tackle all replacements, insertions and deletions. Again the chapter

is split into a methodology, an explanation of the experiments, and a discussion of results.

Chapter seven consists of the conclusions that can be drawn from both approaches, and offers insight into potential research directions for the future.

Chapter 2

Background and previous work

2.1 Shared tasks

There have been numerous shared tasks in the field of GEC, specifically the CoNLL-2013 (Ng et al., 2013), CoNLL-2014 (Ng et al., 2014) and AESW-2016 (Daudaravicius et al., 2016) shared tasks. The CoNLL-2013 task aimed to correct 5 particular errors found in learner English, and the CoNLL-2014 task was an extension of this, which aimed to correct all errors. The submissions to both consisted largely of both Statistical Machine Translation (SMT) models, and machine learning based classifiers, and helped established a baseline for the task for future research. Current research often evaluates on the test data from the CoNLL-2014 Shared Task in order to compare newer models with what was previously considered the state-of-the-art.

The Automatic Evaluation of Scientific Writing task (AESW-2016) was different in that it was closed domain, specifically focusing on the language used in scientific articles. It also invited participants to either correct errors in the test corpus, or simply detect where the errors are. This led to a lot of research focusing on detecting errors, as opposed to correcting them, because the task was easier. Despite these specifications however, many novel approaches were suggested for Grammatical Error Correction, and the winning participants won by implementing Neural Machine Translation for the first time.

The following sections will outline some of the most relevant models from these tasks, and other important research in the area. Although this research focuses on

Artificial Error Generation, it is important to understand in what context these errors will be used. For this reason, the various approaches to Grammatical Error Correction will be covered first. This will then be followed by coverage of the data available to the public, as well as an overview of the few attempts so far to generate more training data automatically.

2.2 Grammatical Error Correction

2.2.1 Rule Based Approaches

Rule based approaches have dominated the market until recently. Using Natural Language Processing, it is possible to exploit the morphological, syntactic and even semantic information of an input sentence and correct it accordingly. Many such approaches are in some cases better performing than some machine learning based approaches, as shown by Sidorov et al., 2013 in their well-performing entry to the CoNLL-2013 shared task. LanguageTool, also a rule-based system, still performs comparably with the best applications today. However, it is difficult to target semantic errors, and more complicated syntactical or lexical errors require numerous overlapping rules.

2.2.2 Machine Learning Based Classifiers

Dahlmeier, Ng, and Ng, 2012 created a pipeline which uses classifiers to suggest improvements to three specific errors one after the other. For example, it would classify a determiner problem by producing a confusion set of replacement determiners, including the "empty" determiner, and use a language model to suggest which one would be best.

Rozovskaya and Roth, 2014 achieved the second rank in the CoNLL-2014 task with the same technique run on five different errors. Notably, they improve the model using a language model which is adapted to learner data. One possible direction for future research is improving models depending on the native language of the writer

of the input data. This is particularly pertinent for generating artificial data, as it would be possible to adapt the data according to the native language.

2.2.3 Statistical Machine Translation Approaches

Using Statistical Machine Translation (SMT) has many advantages over classification, mainly that it can approach the problem more generally. There is no need for the manual input of several types of error corrections, as a trained model can in theory translate them all at once. Using SMT also allows for a very flexible approach. Domain-specific data can be used to specialise a pre-trained general model in order to tackle certain specific types of document, or correct certain types of errors. It is also data-driven, and does not require hand-written rules, which is a worthy consideration when finding a language-independent approach.

In order to see how SMT differs from the classifier approach, Rozovskaya and Roth, 2016 compare the leading systems in the CoNLL-2014 shared task, the classifier discussed above (Rozovskaya and Roth, 2014), and the tuned SMT approach in third place from Junczys-Dowmunt and Grundkiewicz, 2014. Despite the classifiers performing better on specific errors, such as article omissions and long-distance dependencies, due to it abstracting beyond lexical context, the SMT approach is more capable of correcting complex errors. This means that SMT can make use of the data which trains it in order to correct errors which are harder to define with hard-coded rules.

The winner of the CoNLL-2014 GEC task combined an SMT approach with a classification approach (Felice et al., 2014). It is clear that both methods have advantages, and it is possible to combine both in a pipeline structure. By using the classifier to identify the corrections made by an MT system, a supervised "reranking" procedure can then be applied to the translated outputs to decide which errors are most pertinent (Yuan, Briscoe, and Felice, 2016) (Mizumoto and Matsumoto, 2016) (Hoang, Chollampatt, and Ng, 2016). By adding this reranking feature, the best non-neural approach to the problem was found.

There has also been encouraging research from Chollampatt, Hoang, and Ng, 2016

which successfully adapts the SMT model using a Neural Network Joint Model (Devlin et al., 2014), which includes source data information from particular L1 languages, in a similar way to how the classifier approach would adapt to learner English.

2.2.4 Neural Machine Translation Approaches

In the past two years, attention has shifted towards using Neural Networks to correct grammatical errors. The AESW-2016 Shared Task was the first to showcase successful implementations of Neural Networks for the task. For example, the team in second place (Lee et al., 2016) used a Convolutional Neural Network to extract features from the source sentence, which were then used to determine if there was an error or not. Detecting errors is of course a different end-goal to that of this research, but it is nonetheless worth noting the success of the approach. Other recent success comes from Rei and Yannakoudakis, 2016, who experimented with different types of compositional neural network architectures in order to achieve the same goal of detecting errors.

However, the best performing neural models are sequence to sequence models which "translate" the error-prone English into correct English. These models, ported over directly from the field of Neural Machine Translation (NMT), have proven to be the best approach so far to correcting errors. Yuan and Briscoe, 2016's NMT system is, at the time of writing, the best performing model on the CoNLL-2014 test set. It uses a simple Bi-directional RNN encoder and decoder with attention.

Several modifications to the standard NMT system have been suggested for the task of GEC. The winner of the Error Detection AESW-2016 task (Schmaltz et al., 2016) also made an attempt to correct the errors it detected using a character based CNN in conjunction with an LSTM encoder-decoder sequence to sequence model with attention as well as a highway network. Rei and Yannakoudakis, 2016's research concludes that architectures using two Bi-Directional LSTM layers with attention returns a higher F-score than simple Recurrent Neural Networks.

Ji et al. (2017) have improved on the system by including a nested attention model in which both characters and words are taken into account, instead of just words. This

improves the system overall, but noticeably improves how well the model corrects spelling errors. Due to the fact that words with spelling errors are more often than not out of vocabulary words, it is often difficult for a word based model to correctly find the appropriate replacement word from the vocabulary. By giving weight to the characters making up each word, finding the connection between the misspelled input and the intended word becomes easier.

Sakaguchi, Post, and Durme (2017) have criticised the use of the until then standard objective function called the Maximum Likelihood Estimate, which is based on word-level accuracy. This is problematic in the case of GEC because once a model fails to predict one word correctly, the entire sentence's loss is increased. They propose a new sentence-level objective function based on Neural Reinforcement Learning which improves on the word-based variant.

Currently, the field of Grammatical Error Correction (GEC) is dominated by neural translation models. However, despite offering substantial improvements on the well-established SMT approaches to GEC, neural networks come with their own challenges.

Neural models require a large amount of training data. However, the amount of annotated learner English consisting of source sentences (original text) and target sentences (corrected text) is low. It is expensive to source erroneous English from non-native speakers, and then have language experts correct it. Due to the low volume of data, models are at risk of over-fitting.

The data that has been used up until now does not generalise very well across different test sets. This means that there has been some success in correcting errors, but only from test sets that are in some sense similar to the training data. This is in general due to a lack of open-domain training data. Language data containing erroneous English tends to come from schools and universities who correct the essays of English learners in a language class. The essays, as a result, are often about the same topics, and the majority of writers tend to have the same language background.

It is generally unknown how many errors need to be corrected at test time. Current models are unable to leave correct sentences unchanged, as well as make drastic changes to very badly formed sentences, because they are very reactive of how many

Training Data	Sentences	Error
NUCLE (train)	45,721	38.20%
FCE Corpus	25,057	64.83%
Lang-8 Learner Corpus	509,162	97.95%

TABLE 2.1: Statistics on each of the three principle training sets used in the field of GEC.

errors are corrected in the training data. The relationship between the distribution of errors in the training data and the errors corrected in a given test set is as of yet unclear.

2.3 Training data

All of the Machine Learning based researchers cited so far use supervised data to train their models. There is a variety of manually-corrected learner English data available online, however there are a few issues with a supervised approach using this data:

- Almost all available data is in English. There are few resources for other languages.
- All the data comes in different formats. Some have corrections on the word-level and others on the character-level. Different annotation systems are used in each corpus.
- There is not always a "correct correction", and there is large variation between human annotators. The data is therefore inconsistent.
- The datasets are all very small.

The motivation for using Automatic Error Generation is then clear. Data could be generated in any language, in any format, and in any manner of ways. Crucially, it can increase the size of training sets for NMT. Nonetheless, the data is still useful for other purposes. The principle datasets available for this task will be outlined in the following section. Table 2.1 displays the size of each corpus, as well as a percentage showing how much error is in each corpus.

2.3.1 NUCLE Corpus

The CoNLL-2013 Shared Task (Ng et al., 2013) was the first time this corpus saw use in the field GEC, and quickly became the benchmark corpus used to compare a model against the state of the art. It was created by Dahlmeier, Ng, and Wu, 2013 as the NUS Corpus of Learner English. It consists of 1,414 essays written by students at the National University of Singapore (NUS) who are non-native speakers of English. The essays are related to many different specific topics, and are all manually corrected by several different English language experts using a graphical user interface created specifically for the project. For the purpose of the shared task, the corpus is split into a training set and a test set. A separate test set was also kept for the 2014 Shared Task, which used the same corpus.

Of particular note is the remarkably low error rate in this corpus. This means that the many studies which have been carried out in the field of GEC have trained their models on data in which less than half of the sentences are corrected at all. This plays an important role on how often errors are corrected at test-time.

A second consideration to be taken into account is the fact that the vast majority of authors of the essays share the same language background. This is because the corpus was created using only students from the University of Singapore.

2.3.2 FCE Corpus

The Cambridge Learner Corpus (CLC) is a large collection of learner English developed by Cambridge University Press and Cambridge Assessment. It is mostly not available for the public, and instead used for internal teaching research purposes. However, a subset of the corpus was processed and made publicly available for research on Grammatical Error Detection. It consists of 1,244 corrected exam scripts from the Cambridge ESOL First Certificate in English between 2000 and 2001[1].

This corpus is smaller than other available corpora, but it has one main advantage in that the data carries a lot of metadata, for example the native language of the writers, as well as tags for which kind of error is corrected. This makes this dataset

[1]The corpus is available online here: https://www.ilexir.co.uk/datasets/index.html

appropriate for research on surrounding issues related to GEC, such as the effect of L1, or particular errors.

2.3.3 Lang-8 Corpus

The Lang-8 Website[2] is an online community in which native speakers of a particular language correct the language of learners, who post sentences that they are having trouble with online. Advertising over 90 languages being used, the website has the potential to provide learner data for not only English, but many other languages.

The Lang-8 Corpus of Learner English is one of several Corpora made available online on request from the creators of the website[3]. It consists of 100,051 written "entries" into the website, which themselves are made up of several sentences. These entries are contributed by 29,012 different users, who all speak a large variety of native languages.

The corpus was created by crawling the website, which means that it is possible that despite pre-processing, the data may be noisy from other text on the web pages. It also relies on the corrections of other users, as opposed to trained language experts. It is hard to guarantee the strength of the corrections, but the size of the corpus more than makes up for any noise. It is 10 times larger than the NUCLE corpus, and since the website was only crawled in 2011, there is potential for even more data already today, and certainly in the future.

The entire available corpus also contains English sentences which were untouched. Only around one third of the sentences are actually corrected. The error rate is so high because only sentences with corrections were taken into account. It should be noted that this corpus cannot reliably be compared with previous research due to a general confusion as to which version of the corpus has been distributed.

[2]http://lang-8.com/

[3]The Lang-8 Learner Corpora are available on request from http://cl.naist.jp/nldata/lang-8/

2.3.4 Wikipedia

Wikipedia pages undergo revision very regularly. Some of those revisions are grammatical error corrections, and it is possible to extract them for use in GEC. Cahill et al., 2013 extract preposition errors by parsing revision data, and specifically looking for revisions in which only one word is changed. These revisions are singled out using an efficient diff algorithm, and as long as the word is never reverted back to something it once was, it is counted as a preposition revision.

Wikipedia naturally comes with disadvantages. The pages can be changed by anybody, and are very susceptible to vandalism. It is also full of every kind of revision, and not necessarily grammatical correction. Parsing specific data from such a large dataset is also a very unexplored practice, with no tools available for extracting revisions specifically. Analysing such large quantities of data is also very demanding on disk space, as well as time. Despite following the methodology of Cahill et al., 2013, it was decided that this source of data would not be used for this research due to hardware and time constraints.

2.3.5 2015 News Crawl

In this thesis, artificial errors are introduced into a corpus of "good" English. The biggest benefit of doing this is the ease with which one can source a very large, publicly available corpus. For English in particular, it is very easy to find good quality monolingual data.

In this project, the News Crawl of articles from 2015 was taken from the recurring translation tasks of the WMT workshops, which are part of the ACL conference[4]. It contains 21,789,157 sentences of clean, processed English sentences from a variety of news websites. It is widely regarded as a high quality, open-domain source of correct English sentences. The data is tokenised in keeping with previous research using the NLTK-Toolkit (Bird, Klein, and Loper, 2009).

[4]The data is available online at http://www.statmt.org/wmt16/translation-task.html

Source	I liked the colour combination and the designs on her sari .
Human Error	I liked colour combination and designs on her sari .
Machine Error	I liked liked colour combination and the designs on on . .

TABLE 2.2: Comparison of human and machine error.

2.4 Artificial Error Generation

By taking monolingual data, and artificially injecting errors into it, it is possible to generate parallel corpora for GEC suitable for training an NMT model. The injected errors should ideally replicate the errors made by real learners of a language, so that the model would be trained to correct such errors, rather than other types of noise.

An example of an appropriate introduction of error is shown in Table 2.2.

The goal is to introduce error in a way that a human would, so that the model learns to correct human errors, not machine errors. At test time, machine errors will not be present, but the model, having learned how to correct them, may use this information when dealing with human errors. Therefore, unnecessary errors must be avoided. On the other hand, there will always be errors which haven't been made yet, and a liberal coverage of potential errors, even if many of them are not human-like, may result in an improved ability to correct new errors. Nonetheless, an analysis of the errors made by humans is needed to inform which type of errors should be introduced.

2.4.1 Types of human error

TABLE 2.3: Table taken from Ng et al., 2014 listing the different error types used in the CoNLL-2014 Shared Task. The list is ordered according to the percentage rates of how often each error type occurs according to the NUCLE corpus.

Error Type	**%**	**Description**
ArtOrDet	14.8	Article or Determiner
Wci	11.8	Wrong collocation/idiom
Rloc-	10.5	Redundancy
Nn	8.4	Noun number
Vt	7.1	Verb tense
Mec	7.0	Spelling, punctuation, capitalisation etc.
Prep	5.4	Preposition
Wform	4.8	Word form
SVA	3.4	Subject–verb agreement
Vform	3.2	Verb form
Trans	3.1	Linking words/phrases
Um	2.6	Unclear meaning
Pref	2.1	Pronoun reference
Sun	1.9	Run-on sentences, comma splices
WOinc	1.6	Incorrect word order
Cit	1.5	Citation
Wtone	1.3	Tone (formal/informal)
Others	1.3	Other errors
Spar	1.2	Parallelism
Vm	1.0	Verb modal
V0	0.9	Missing verb
Ssub	0.8	Subordinate clause
WOadv	0.8	Incorrect adject/adverb order
Sfrag	0.6	Sentence fragment
Npos	0.5	Noun possessive
Pform	0.4	Pronoun form
Wa	0.1	Acronyms
Smod	0.1	Dangling modifiers

Table 2.3 refers to the list of 28 error types which are considered for the CoNLL-2014 Shared Task. Examples of each error type are in the Appendix in Table A.1. One of them is simply labeled "Other", meaning that this list covers all types of error. The table has been ordered according to how often each type of error appears in the NUCLE corpus, which was used for the same shared task.

It was found that Article and Determiner misuse is the most common mistake for English learners. It is however worth noting that this corpus is from the University of Singapore, where many of the students have Chinese Mandarin as their first language. Mandarin does not use articles as they appear in English. This could be a contributing factor towards the high proportion of these errors.

In the case of Automatic Error Generation, many of these error types are simple. For example, Article and Determiner misuse could be emulated by simply swapping single words. Likewise, the error types 'Nn', 'Vt', 'Mec', 'Prep', 'Wform', 'SVA', 'Vform', amongst errors could potentially be represented with word replacements. However, around half of the errors could not be emulated with a simple script. Collocations and idioms form 11.8% of the error distribution, and remain an untouched problem. Many other high scoring error types, such as 'Rloc-', 'Trans', 'Um', 'Pref', are also difficult to introduce.

It is also worth noting that many error types combine, often within a single word. For example, in the sentence "Early examination is health and will cast away unwanted doubts.", the word health has two errors: Firstly, a word form error, where it would be corrected to "healthy", and secondly a wrong collocation error, where it would be changed to "advisable". This, coupled with the fact that the manual annotators of each of these 28 error types may not always be certain about the correct category for each edit, suggests that forming the error correction problem as a series of 28 separate problems is not complex enough. This is evidence for a more general solution, and a reason why machine translation approaches may be more appropriate than rule-based approaches.

2.4.2 Introducing specific errors

The most researched approach to AEG is the introduction of specific errors. In most cases, a specific type of error, for example article misuse, is targeted and generated using a script. This has the advantage of being flexible, and specific errors which might be difficult for rule-based systems to tackle can be targeted. It is however very language specific, and these deterministic approaches to noise generation are similar to the rules of a rule-based system, with each new rule requiring a lot of effort to implement.

The first attempt to introduce noise was by swapping every article in a POS-tagged text with a random selection of either 'the', 'a' or 'an' (Izumi et al., 2003). It was the first to report improved precision of the correction task thanks to artificial error data. Since then, several novel approaches have been taken to tackle various errors, and some in other languages too, for example particles in Japanese (Imamura et al., 2012), and split compounds in Swedish (Sjöberg, 2005).

In the above Section 2.4.1 it was discussed that the CoNLL-2014 Shared Task specifies 28 different types of error, where the 28th error is 'Other', which effectively requires every type of error possible to be corrected. Liu and Liu, 2016 successfully tackle 16 of these errors by creating a substitution set for each, and similar to the example with articles, replacing words with a random substitution from that word's potential error set. Sennrich, 2016 tackles five difficult German-specific errors using random substitutions in a similar way. They use a character based model, allowing them to transliterate unseen names. Swapping certain characters to create unseen words is also used to train the character based model to carry out appropriate transliterations. These two approaches saw success in their limited domains, and inspired the first approach taken in this thesis.

Brockett, Dolan, and Gamon, 2006 was the first to take artificially generated errors and use them to train a machine translation system. They trained an SMT system on data containing artificial mass noun errors from Chinese speakers. Their methodology was simple, but their results reflected the fact that being able to generate data is very valuable. At this point in the field, there was not much data available, and enhancing what was available with artificial data made a large difference.

One other notable work concerning the usefulness of unlabeled data is from Liu and Liu, 2016, although it is for a related problem, Grammatical Error Detection. They formed the problem into a binary classification problem, in which they derived whether a sentence is well-formed or not from a small amount of data, and used this information to train the classifier. Its success shows the potential for using unlabeled data for the task.

Rozovskaya and Roth, 2010 look at article error generation in detail, and experiment with different substitution methods. Most interestingly, information from learner English from authors with three different native languages without article usage (Czech, Russian and Chinese) is used to estimate how often a certain article should be used in the substitution. They find that different an L1 affects the error distribution, and that emulating this specific learner text means that models can be trained to better correct the errors from that particular L1. This precedes the research on SMT by Devlin et al., 2014, which shows how this kind of adaption can be engineered with AEG.

One of the most pertinent studies in AEG is from Cahill et al., 2013, who take information about the learner English directly from Wikipedia, a community website on which everybody is free to correct language, and use information about how often certain words are replaced to generate confusion sets. These confusion sets are then used to inform a stochastic sampling of replacements when introducing error. Finally, the produced training error is fed into an SMT system. This approach has its flaws: Wikipedia, as discussed in Section 2.3.4, is difficult to work with, and not necessarily reliable. The study also only concerns itself with prepositional error. However, the methodology is very flexible and has the potential to be generalised. This is why this study has inspired much of the methodology used in this thesis.

This work, along with Rozovskaya, Sammons, and Roth (2012), are the only two studies to research the effects of the amount of error in the training corpus when using Machine Translation to correct grammar. Their experiments involving different error densities hint at a phenomenon known as “Error Inflation”, which is where models trained with more errors than in the test set perform better. This is explored further in this thesis.

2.4.3 Introducing all errors

Very few attempts have been made to introduce all types of error at once into a corpus. The main study to do so is Rei et al. (2017), who took two separate approaches to the problem. The first is a pattern extraction based approach, in which a similar approach to Cahill et al., 2013's is taken, but instead of focusing solely on prepositions, all replacements are targeted. In order to focus the data, a context consisting of the POS-tags of the words preceding and following each replacement is taken into account. Confusion sets are generated from this information, which is used to inform the stochastic sampling of replacements when introducing the errors. This approach is similar to the second approach taken in this thesis. However in this work, correcting errors is the end-goal instead of detecting errors, as is done in this work.

The second approach makes use of SMT to generate the error. The process consists of translating good English into erroneous English using a small, background corpus as training, but crucially using the corrected sentence as the source, and the incorrect sentence as the target. By doing so, the generated error should resemble the error in the background corpus. This round–trip use of MT has already seen success with Madnani, Tetreault, and Chodorow, 2012, who showed that it was possible to correct errors by translating a sentence through several languages before translating back into the original language again.

Chapter 3

Evaluation

It is important for every field of research to have its benchmark test to see how well a particular approach performs in comparison with others. However, in smaller, less researched fields such as GEC, it may be difficult to find the perfect test set and scoring system. In GEC, many evaluation techniques have been borrowed from the larger fields of Machine Translation and Machine Learning. As a result, the evaluation may not be entirely appropriate for monolingual error correction.

In this section, the two most popular test sets will be presented. Then, the various scoring metrics are discussed, with justifications given as to why this research only makes use of two of them.

3.1 Test sets

3.1.1 NUCLE Test Set – CoNLL-2014 Shared Task

The last three years has seen the field of Grammatical Error Correction almost entirely focus their efforts on developing models which perform on the test set supplied for the CoNLL-2014 Shared Task the NUCLE corpus (Dahlmeier, Ng, and Wu, 2013). It can be used in order to compare with older approaches, and for this reason alone it is worth considering using. There was also a very similar test set from the previous year's shared task, although it was the 2014 test set which has been set as the benchmark.

It is a set of 1327 sentences taken from the essays of learners of English from the University of Singapore, as described in **??**. It does however have several flaws:

- Despite the corpus that this is derived from containing essays about a wide range of topics, this test set only has essays concerning the health and technology domains. An ideal test set would be open domain, so that using training data from these two domains would not affect the results.
- The sentences vary only slightly in length and form. When using a real world GEC application, the input sentence could be of any length, and this test set does not reflect that. Neural models for example have great success correcting small sentences, and great trouble correcting larger ones. Meanwhile, rule-based systems perform similarly across all lengths of sentence. It would be better to have many types and sizes of sentence to avoid any potential bias for a particular approach.
- This test set still suffers from the problem of having all the authors of the source sentences have similar language backgrounds. An ideal test set would have sentences written by people from all over the world.

3.1.2 JFLEG Test Set

Napoles, Sakaguchi, and Tetreault, 2017 were the first to criticise the widespread use of the NUCLE test set, and as such, they published an alternative corpus, the JFLEG corpus. At the time of writing, this corpus has only been available for four months, meaning that it has not seen widespread use.

Sakaguchi et al., 2016 explained that GEC evaluation needs to change to suit the changing nature of the field. Since research has changed from solving individual error types to open error correction, our approach to evaluating also needs to reflect that. Specifically, there should be less reward for changing individual words, and more reward for achieving fluency. The JFLEG corpus was created with this in mind, moving away from "minimal edits", in which a single error is corrected regardless of context, and instead having "fluency edits". This means that the corrected sentences in the JFLEG corpus are more liberal, and have more rephrasing

Original	they just creat impression such well that people are drag to buy it .
Minimal edit	They just create an impression so well that people are dragged to buy it .
Fluency edit	They just create such a good impression that people are compelled to buy it

TABLE 3.1: Example of a fluency edit

to make the sentence look like it was written by a native speaker of English. An example of this by Napoles, Sakaguchi, and Tetreault, 2017 is given in Table 3.1.

The corpus makes use of 1,511 sentences taken from the GUG (Grammatical/Ungrammatical) corpus (Heilman et al., 2014), which is a corpus of ungrammatical English written by language learners taking the TOEFL exam. The language covers a wide range of topics and is written by speakers of a variety of languages. The JFLEG corpus corrects these sentences using five crowd-sourced workers and one expert. Each correction was done with fluency in mind, instead of minimal edits. By the nature of its fluency edits, the corpus consistently returns lower scores than the NUCLE test set, which consists only of "minimal edits". It also has a higher error density (85%) than the NUCLE test set (72%), which by definition means that there is more to correct. These two factors result in JFLEG being the more difficult test.

3.2 Evaluation Metrics

In this section, the various approaches to generating a score for a given correction are presented. The evaluation metrics have changed over the course of the development of the field, resulting in many results from different studies becoming incomparable without reimplementation.

3.2.1 BLEU Score

Since many approaches towards GEC stem from machine translation, it follows that the main method of evaluating the models was for a long time borrowed from it. The BLEU score (Papineni et al., 2002) has often been utilised as the evaluation metric, relying on the assumption that evaluating on corrected language is much the same as evaluating on translated language.

These scores are, however, less appropriate for GEC, because the majority of the words remain unchanged. In Machine Translation, each word should be changed, whereas in GEC, some, but not all regions of the source sentence must be altered. BLEU score and F-score assign weight equally to how well each word was changed, and as a result, small but correct changes are not appropriately rewarded.

3.2.2 M^2 Score

For the CoNLL-2013 and 2014 Shared Tasks, a new metric was used, the M^2 scorer (Dahlmeier and Ng, 2012), also known as the MaxMatch scorer. This aimed to tackle some of the problems with simply using precision and recall, namely that it was not possible to pick up longer, phrase-based error corrections, and that it only compared how close the output was to the gold standard without taking specific correction information into account. The M^2 score compares the changes made by the system with the changes made in the gold standard, and calculates the F-score from that. Thus, it is no longer a case of evaluating the output itself, but rather the changes the system makes. This has since become the most widely used evaluation method, due to its use in this benchmark evaluation set.

3.2.3 I-Measure

One extension of this was developed by Felice, 2016, called the I-measure. It is an extension of the M^2 Score which uses labeled annotation data at a word level. Most crucially, it calculates a weighted accuracy in an attempt to redistribute the rewards and punishments of editing the input. The "do nothing" baseline often receives a higher score than a good correction, but this measure would give more weight to a correct edited word than a correct instance of leaving a word unchanged and give more weight to the punishment of unnecessary corrections than uncorrected errors.

3.2.4 GLEU Score

More recently, Napoles et al., 2015 tested the use of these metrics against human evaluation, and found that they did not strongly correlate. In order to develop a

new metric which represents what is presumed to be the ground truth, human evaluation, they return to the BLEU score. They argue that using the specific annotation information in the M^2 Score and the I-Measure is risky because the annotations themselves are almost always ambiguous. The very nature of GEC means that annotations are variable, and there is no single correct annotation. The type and form of these labeled annotations also vary from task to task and are difficult to work with. Simply using the BLEU score however is not appropriate, as it returns unintuitive results—it ranks the source sentence higher than 13 out of 14 of the system outputs from the CoNLL-2014 Shared Task. As explained in Section 3.2.1, machine translation tasks and monolingual re-writing tasks are different, and the BLEU score caters more towards machine translated output.

Instead they propose a modification of it known as the Generalised Language Evaluation Understanding (GLEU) Score. This method calculates a weighted precision of n-grams, in a similar way to how the I-Measure calculates a weighted accuracy of edits. On top of rewarding correct edits, it also aims to reward instances when a word is deemed not in need of correction, as well as to punish incorrect edits. Taking a standard metric from Machine Translation and adapting it for GEC results in the most human-like automatic evaluation, according to comparisons made in the study.

In this research, three scores are reported. Firstly, the GLEU score using the JFLEG corpus. Secondly, the GLEU score using the NUCLE corpus. Thirdly, the M^2 Score using the NUCLE corpus. The first is because both the corpus and the evaluation metric are better suited to the translation task, particularly as fluency is an important goal for NMT systems. The latter is so that the model can be compared with previous models, although it should be noted that this older form of evaluation rewards minimal edits over fluency edits, and favours rule-based systems over MT systems. Where possible, GLEU scores on the NUCLE corpus are compared in order to give a fairer outlook on the relative performance.

Chapter 4

Experimental Setup

This chapter explains the framework for the experiments described in Chapters 5 and 6. First, it contains an introduction to the Neural Translation System chosen for the project, OpenNMT, as well as a justification for this robust implementation. Then, the concept of LSTM-based encoder–decoder neural systems are explained. Finally, the chapter covers a series of preliminary experiments which were conducted in order to decide which parameters offered by OpenNMT should be used.

4.1 OpenNMT

The chosen implementation for this research is OpenNMT[1] ("OpenNMT: Open-Source Toolkit for Neural Machine Translation"), an open-source Neural Machine Translation implementation which utilises the Torch toolkit. The model consists of a bidirectional 2-layer LSTM in a sequence to sequence encoder–decoder system with attention. It was chosen because of its ease of use and robustness. It also reflects, to a large degree, the architecture of the state of the art at the time of writing (Yuan and Briscoe, 2016). Other modifications to this "vanilla" Machine Translator have proven to improve results for GEC, as discussed in section 2.2.4. However, the research questions in this thesis relate almost entirely to the data being used to train the MT system, and not the MT system itself. Running custom modifications on a sequence to sequence system would be preferable if it means the results are better, but it is beyond the scope of the project, and will not affect the lessons learned about

[1] http://opennmt.net/

the data. For this reason, OpenNMT, a robust, easy to use, out-of-the-box implementation was chosen. A brief overview on how OpenNMT works is given in the following section.

The experiments were run on NVIDIA GPU clusters available at the Universität des Saarlandes.

4.2 Encoder–Decoder Neural Networks

OpenNMT uses a standard neural approach to Machine Translation called sequence-to-sequence translation. This where a sequence of words is inputted into an encoder, where is it transformed into an internal representation. This internal representation is then transformed back into a new sequence of words by means of a decoder. The transformations are carried out according to weight matrices, which are trained on data containing source sentences and target sentences, known as parallel data. The encoder and decoder are both Recurrent Neural Networks (RNN), and both make use of LSTM cells to help with long distance dependencies across several time steps. Finally, OpenNMT also implements an attention network, which improves translation quality by connecting individual words from the encoder and the decoder.

4.2.1 Recurrent Neural Networks

A Recurrent Neural Network (RNN) is a neural network function which is applied iteratively on a sequence of inputs, in this case word embeddings. At each time step in the sequence, an RNN instance takes the corresponding word along with the output from the RNN instance of previous time step. A simple RNN takes the output of the previous time step which has been translated by a corresponding weight matrix (known as the hidden state) in addition to the next word embedding and models the probability distribution at the current time step as follows:

$$s_t = tanh(Ww_t + Us_{t-1} + b)$$

$$y_t = Vs_t$$

$$P(w_t|w_1..w_{t-1}, \Theta) = softmax(y_t)$$

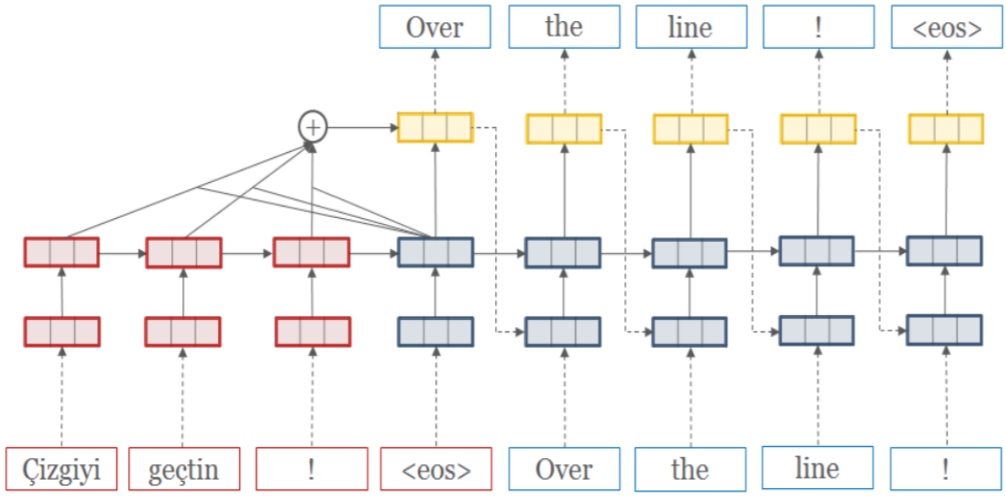

FIGURE 4.1: Schematic view of neural machine translation taken from "OpenNMT: Open-Source Toolkit for Neural Machine Translation". The red source words are first mapped to word vectors and then fed into an RNN. Upon seeing the <eos> symbol, the final time step initialises a target blue RNN. At each target time step, attention is applied over the source RNN and combined with the current hidden state to produce a prediction of the next word. This prediction is then fed back into the target RNN.

where $\Theta = \{W, U, V, b, WE\}$. W, U, V are matrix parameters, WE is the word embedding matrix, b is the bias, w_t is the word embedding from the word embedding matrix WE and s_{t-1} is the hidden state at time t-1. The softmax function occurs in the final layer, and is over the entire vocabulary. The objective function is further optimised using the "Adam Optimiser". The size of the RNN units are defined by the hyper-parameter "RNN size". A higher RNN size means a higher dimensionality, which in turn means more information can be stored inside each unit.

The parameters are updated via back-propagation through time. During this process, the development of the back-propagated error depends exponentially on the amount of weights there are. As RNNs become larger, they are therefore more at risk of a common problem in Machine Learning known as exploding or vanishing gradient. One solution to this problem is a specific alteration of the RNN called the LSTM.

4.2.2 Long–Short Term Memory

A Long–Short Term Memory (LSTM), proposed by Hochreiter and Schmidhuber, 1997, is a specific type of RNN which overcomes problems with gradient descent by ensuring constant back-propagated error flow using gates. There are three different gates, each of which are trained alongside the rest of the network. They dictate how much information is carried on through the time steps. By doing so, it is possible to retain information from words at the beginning of a sentence. The usage of LSTMs has become popular in MT due to the number of long-distance dependencies in language which must be taken into account when translating. For example, the translation of "went" in German has two words that are likely to appear far from one another in a sentence:

I went to school $\rightarrow$ Ich bin zur Schule gegangen

The use of gates allows a model to consistently translate words which are sent to the end of the sentence ("gegangen") in German when they are at the beginning of a phrase in English. In GEC, there are similar long-distance dependencies. For example:

*The three boys, as well as the cat, is ready

The verb "is" refers to their subject, "the three boys", and in order for a GEC system to know to correct this term into the plural "are", it must be able to reason over a long distance. For this reason, OpenNMT's use of LSTMs are appropriate not only for MT, but also GEC.

The LSTM function is formulated in the following way:

$$
\begin{aligned}
f_t &= \sigma(W_f w_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma(W_i w_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma(W_o w_t + U_o h_{t-1} + b_o) \\
c_t &= f_t \odot c_{t-1} + i_t \odot tanh(W_c w_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \odot \sigma(c_t)
\end{aligned}
\tag{4.1}
$$

$$P(w_t|w_1..w_{t-1}, \Theta) = softmax(W_s h_t)$$

where $\Theta = \{W_f, W_i, W_o, U_f, U_i, U_o, b_f, b_i, b_o, b_c, W_s, WE\}$. All of the W's and U's are matrix parameters, the b's are bias parameters, the WE is the word embedding matrix, w_t is the word embedding (from the WE matrix) at time t, h_{t-1} and c_{t-1} are the hidden state vectors and context vectors at time t-1 respectively. The symbol $\odot$ represents element-wise multiplication. σ refers to the element-wise application of the sigmoid function on the input vector, which produces an output vector whose elements are in the range $[0, 1]$. A gate is a vector of floats in the range $[0, 1]$, and therefore, i_t,f_t and o_t in the above equations are gates. They are named "input gate", "forget gate" and "output gate" respectively as per their roles in the final prediction.

4.2.3 Attention models

Despite the fact that LSTM gates should encode all of the context for long-distance dependencies, in practice there are still issues, particularly in long sentences. One workaround is to feed a reversed version of the sentence into the encoder as well as the normal version. This is called using a bi-directional model, and is also implemented in OpenNMT. However, one of the most successful developments has been adding an attention mechanism.

Adding an attention layer to an encoder–decoder system means adding a simple Feed–Forward neural network for each time step in the decoder. The mechanism decides how much information from each of the input words should be used to decide the outcome. In theory, a well-trained attention model will tell the English to German translator to take the verb into account when reaching the end of the sentence, and will tell a GEC system to take the subject of the sentence into account when reaching the verb, so that it can select an appropriate verb form.

A sample neural network layer that performs attention is defined as follows:

$$attn = softmax(h^T W S)$$

where h^T is the transpose of the hidden vector of an RNN instance at some time–step, W is a matrix parameter, S is a matrix whose columns are vector representations of the sub-units of the context on which the attention is required. $attn$ is be a vector of scores each signifying the importance of a sub-unit in the context.

4.3 Preliminary Experiments

In order to establish some baselines for research, a number of preliminary experiments were carried out. Firstly, all available supervised data was used as standalone training data for the NMT system. Secondly, an auto–encoder approach in which no error is generated is tested in order to learn about using large volumes of monolingual data. Finally, random article errors are introduced into a corpus to test the plausibility of targeting errors using AEG.

4.3.1 Supervised Data

The three main datasets available for use in research are the NUCLE corpus, the FCE subset of the Cambridge Learner Corpus (CLC), and the Lang-8 English Revision Corpus. All three are detailed in Section 2.3. Table 4.1 displays firstly the GLEU scores of the three datasets on the JFLEG test set, secondly the GLEU scores from the NUCLE test set, and finally the M^2 scores, also on the NUCLE test set. Also given is the winning M^2 score reported in the CoNLL-2014 Shared Task. The system outputs were released, so it was also possible to get the GLEU score of this approach on the NUCLE test set. Furthermore, the M^2 from Yuan and Briscoe (2016)'s state of the art neural model trained on the entire CLC dataset is included for reference.

The models were trained with an RNN Size of 1000, a word vector size of 500, a vocabulary size of 50,000 and a dropout of 0.3. These parameters were selected after a small amount of tuning. At the beginning of the research, RNN Sizes between 100 and 1500 were tested on the Lang-8 corpus, with 1000 having a higher M^2 score. A low RNN Size resulted in under-fitting, in which only gibberish is output, and a high RNN Size resulted in the output copying the input word for word. A similar effect was observed when varying the word vector size, and a value of 500 was picked in

Training corpus	**GLEU score - JFLEG**	**GLEU score - NUCLE**	**M^2 score - NUCLE**
FCE	34.91	41.65	9.47
NUCLE	38.41	57.45	10.21
Lang-8	49.56	61.30	28.89
CoNLL-2014 Winner	–	64.84	37.33
Yuan & Briscoe (2016)	–	–	39.90

TABLE 4.1: GLEU scores and M^2 scores of models trained on different available supervised datasets

a similar fashion. Having too large a vocabulary size resulted in lengthy training times, so this was limited to 50,000. Dropout values between 0.1 and 0.7 were tested, with 0.3 achieving the best score. 15 epochs were used.

The size of the corpora reflect their performance with the smaller FCE and NUCLE datasets having much lower scores across all evaluation metrics than the larger Lang-8 corpus.

Somewhat surprisingly, the model trained on the NUCLE test performs significantly worse than the winner of the CoNLL-2014 Shared Task (Felice et al., 2014). Their approach combines a rule-based approach with an SMT system, and is very successful. It should be noted, however, that the M^2 score benefits rule-based systems such as this, as is reflected by the smaller improvement in GLEU on the same test set. Using NMT on such a small dataset did not function very well. There is evidence of under-fitting in the output of this model, which means that an NMT approach simply needs more data. The other NMT system compared in the table is from Yuan and Briscoe (2016). It makes use of the entire CLC corpus, which comprises almost 2 million sentences. This is four times the size of the Lang-8 Corpus. It is also of a higher quality, as the corrections are done by experts and teachers, and not members of the public. These results support the need for more data, which motivates the Artificial Error Generation approach taken in this thesis.

4.3.2 Auto–encoder

In order to test how viable it would be to put large amounts of monolingual data through OpenNMT, a preliminary experiment in which no noise is introduced was carried out. Monolingual WMT English data was taken as both the input and the

Training data	**GLEU score - JFLEG**	**GLEU score - NUCLE**	**M^2 score - NUCLE**
1 Epoch: 10m Sentences	46.21	64.96	6.21
1 Epoch: 22m Sentences	46.66	65.24	6.63
2 Epochs: 22m Sentences	46.78	65.45	6.78

TABLE 4.2: GLEU scores and M^2 scores of models trained on different amounts of sentences from the WMT corpus with either one or two epochs.

output at training time, making an Auto–Encoder. Models were trained with various values for RNN size, Source vocabulary size, and Dropout in an attempt to see at what capacity the neural network starts to generalise when given new input at test time. The goal was to create a baseline of sorts for artificial data. How would artificially generated erroneous data compare with the auto–encoder?

An appropriate RNN size was a low value of 100, because the model was very quick to start over-fitting, and outputting copies of the input. By reducing the RNN size, information which is sent through the encoder–decoder system is reduced to a much lower dimensionality. This effectively allows the decoder to learn to decode a more generalised version of the training data. The result is that the model no longer overfits, and has the potential to correct errors that it has not seen in the training data. In Machine Translation, a much higher RNN size is generally used, because when the input and output do not share the same language, more dimensions are needed for the internal representations of sentences. However, in monolingual translation, there is less input to transmit, and in the case of an auto–encoder, the optimal capacity drops even lower.

In Table 4.2, GLEU scores and M^2 scores are reported for different amounts of the WMT data. The entire dataset consists of over 20 million sentences. However, when introducing errors using one method mentioned in Section 6.1, issues with computer memory were encountered when dealing with so many sentences. For some experiments, only 10 million sentences were used. This table shows that when dealing with so many sentences, the quantity stops making a difference after a certain point. Firstly, the scores for 10 million sentences do not differ significantly from those for the entire dataset. Secondly, only a single epoch is used, because already one pass through the data requires an entire day of processing. In the table, the scores from the second epoch are also displayed to show that there is minimal improvement for training any longer than a day.

Source	**Target**
and you can have **chanse** to make a your person to help you .	and you can have **chase** to make a your person to help you .
traditional **enclopedias** are written by trained expert for academic rigor that can not really achieve .	traditional **encyclopedias** are written by trained expert for academic rigor that can not really achieve .
Forexample , My cousin is **12years** old .	Forexample , My cousin is **years** old .

TABLE 4.3: Examples of the output of a trained auto–encoder with an RNN size of 100.

A GLEU score of 46.66 on the JFLEG test set is significant, considering that the GLEU score of the best supervised data, Lang-8, is only a couple of points higher at 49.56. The M^2 is, however, considerably lower than that of Lang-8. Qualitative analysis shows that some changes are made, but only sometimes do they appear to be corrections. The only changes that are made appear to be on out-of-vocabulary words. In some cases, incorrectly spelled words are corrected by the model, despite the words not appearing in the vocabulary. In other cases, unrecognised words are replaced with something unrelated, or incorrect. Some examples of the changes made are given in Table 4.3.

4.3.3 Random Article Errors

This preliminary experiment followed the methodology of Izumi et al., 2003 in their article swapping experiment, due to its simplistic implementation. A simple script was made to replace every instance of the words 'the', 'a' and 'an' randomly with one of those same words. This is taken from a uniform distribution, meaning that one third of these particular determiners remained the same, and two thirds were changed to one of the other two. The aims of this experiment were the following:

- To see if articles in particular are targeted and changed in the output.
- To make appropriate adjustments of the hyper-parameters when noise is introduced.

The errors were introduced into the entire WMT Corpus, resulting in a corpus with a determiner error rate of 66%. An example of such an introduction is as follows:

Source	**Target**
How much a genetic change tells us about your chance of developing a disorder is not always clear .	How much the genetic change tells us about your chance of developing the disorder is not always clear .
An example that may serve as an illustration is breast cancer .	An example that may serve as a illustration is breast cancer .
If there is no social media , it is difficult for people to know about a stranger .	If there is no social media , it is difficult for people to know about the stranger .

TABLE 4.4: Examples of output of system trained on random article error corrections.

By the time he came, we were already finished with the work → By a time he came, we were already finished with an work.

Originally, an RNN size of 100 was chosen, as it was the appropriate capacity for identical inputs and outputs. However, the model trained on the random article errors under-fit with such a low RNN size, and a new RNN size of 150 was used. Considering that the best RNN size for the supervised data was significantly higher, it seems that the more errors which are introduced into the corpus, the higher the RNN size must be.

On the JFLEG test set, the model achieved a GLEU score of 36.64, and on the NUCLE test set, it achieved an M2 score of 9.77, and a GLEU score of exactly 60. All of these values are lower than the models without any artificial noise. This is because the system incorrectly changes the vast majority of articles. A qualitative analysis of the system output reveals that the articles are indeed targeted, but when they are changed, it is likely to be incorrectly. Examples are given in Table 4.4. The output reflects the lack of heuristic in the introduction of error in that the articles are handled seemingly randomly, despite the 22 million corrected sentences in the training data. But, the fact that the articles are targeted validates AEG as a good approach. With a more reasonable heuristic behind the error generation, an appropriate amount of errors introduced, and the optimal parameters, an error type could be successfully targeted and corrected using NMT.

Chapter 5

Tackling Individual Error Types

Although the field of GEC is moving away from tackling individual error types since the CoNLL-2014 Shared Task, it is still the case that much can be learned from specific error types. When dealing with all errors at once, the effects of all the different types of error interact with each other, creating a lot of noise. As a result, it is difficult to attach a reason as to why particular models behave certain ways. This project aims to learn about the effects of introducing a single type of error, and using that information to inform a more general approach.

Single error type correction also has its place in applications such as LanguageTool. Due to the currently low precision of neural network approaches to open error correction, it is conceivable that single error types which are difficult to tackle with a rule-based system could be corrected using NMT. This could be used on top of any other system. An example of such an error would be sentence fragments. In the NUCLE corpus, these accounted for but 0.6% of the errors, which means that there are not very many available examples of this error type. However, it could in theory be easy to generate training data containing this error type by simply deleting parts of otherwise grammatical sentences or splitting them into two sentences, and train an NMT model to solve the error.

With this example, as with all types of Artificial Error Generation, the question of how much error to introduce arises. In such an application as LanguageTool, it would be necessary to convert sentence fragments into complete sentences only some of the time, namely when the model recognises that there is indeed a fragment. A model trained purely on corrected fragments would over-correct on many grammatical sentences. Therefore a balance between the amount of error in the training

data and the amount of expected segment fragments at test time is needed.

This experiment explores this relationship, but by using simpler error types than sentence fragments, specifically prepositions and determiners. Up until now, the effect of the amount of errors in the training corpus has only been explored with prepositions specifically (Cahill et al., 2013). This experiment will expand on the work done in this paper by reimplementing their methodology for prepositions using the more accessible Lang-8 corpus, and applying it to determiners as well as prepositions.

5.1 Introducing the Errors

We follow the same methodology of Cahill et al. (2013) to generate noise. Specifically, supervised revision data is used to see how often particular words are corrected into specific prepositions or determiners. The revision data which is used for our research is the Lang-8 corpus. More details on this corpus are given in Section 2.3.3.

The process of introducing errors into the WMT data using the Lang-8 corpus is as follows:

1. Extract plain text versions of the Lang-8 corpus, consisting solely of sentences with corrections.
2. Compare source sentence with corrections using an efficient *diff* algorithm.[1] Note that this often included several steps of revisions.
3. Prepare a list of all prepositions/determiners. This is taken from the tags of the WMT data retrieved from the Stanford tagger.
4. Remove all sentences that do not contain a single revision involving a preposition or a determiner. Using a hand-crafted set of possible prepositions/determiners, it is determined for each sentence whether it involves a deletion (eg. "for" → "NULL"), an addition (eg. "NULL" → "the"), or a replacement (eg. "on" → "in").

[1] `http://code.google.com/p/google-diff-match-patch`

5. Generate confusion sets for each preposition/determiner by listing all the deletions which are replaced by that word, and counting the frequency of each specific revision.

6. Insert the target word itself into the distribution with a frequency relative to the error rate. An 80% error rate for example means that 20% of the time, the same word is selected, effectively leaving it in its "correct" form.

7. Systematically replace prepositions/determiners in the WMT corpus with one of the options in their respective probability distributions, selected at random by a sampler[2].

Table **??** shows an example of a confusion set. Each source word is associated with a frequency value which shows how often that word was corrected into the target word in the background Lang-8 corpus. By sampling from this set, errors can be introduced according to a heuristic based on real learner English data. In the case of "onto", the word "onto" is included in the confusion set as a way of not always introducing this error. Other notable inclusions are "into", "on" and "to", which all seem to be common mistakes. On top of this, there is a large number of exemplars which appeared only once, for example the spelling error "ointo".

"onto"	**Frequency**
aboard	1
above	1
across	1
against	1
along	2
as	1
at	3
atop	3
by	2
from	2
in	12
into	245
of	1
ointo	1
on	268
onto	278
ontop	1
oto	1
over	6
through	2
to	98
until	4
unto	2
upon	4
with	1

The output of this methodology is convincing, although there are still a number of areas for improvement. For example, a Part Of Speech (POS) tagged version of the corpus was considered a more reliable way of finding the specific errors, but problems relating to the inefficiency of many unsupervised POS-tagging techniques appeared. Instead, a word list of tags taken from the Stanford POS-tagger was collected and used. However, words tagged as prepositions did not necessarily correlate to the errors relating to prepositions. One example of this is the tag "TO",

[2] "NULL" is included in the confusion sets, resulting in the deletion of some words. The insertion of erroneous prepositions is not handled in this experiment. This is because the methodology for doing so was developed during the second approach, after already having carried out this approach. A full analysis of replacing, deleting and inserting errors is given in Section 6.1.1

Source sentence	Error introduced	Changes made
We rather lose sight **of** the fact that so many of those products that make cyberspace work have to be brought to us **from** the other side of the globe .	We rather lose sight the fact that so many of those products that make cyberspace work have to be brought to us **at** the other side in the globe .	1 deletion, 1 replacement
There 's certainly a lack of maturity **on** his part , certainly in dealing **with** women .	There 's certainly a lack of maturity **at** his part , certainly in dealing **from** women .	2 replacements
Brett Lee fiery till the last ball **in** career swan song for Sydney Sixers	Brett Lee fiery till the last ball **for** career swan song for Sydney Sixers	1 replacement
The second was the result of a car accident five years later .	The second was the result of a car accident five years later .	No change
For Mr Sampson , it 's the second appointment to a high-profile board in less than a year , having become a Fairfax director in May .	Mr Sampson , it 's the second appointment to a high-profile board in less than a year , having become a Fairfax director in May .	1 deletion

TABLE 5.1: Random subset of 10 sentences from the 2015 News Crawl which had errors introduced into them.

which covers the multi-functional particle "to" in English. One function of this word is as a preposition, and all errors relating to it were not included. One potential option for future research would be the use of more general word clusters. This might combat the unreliability of POS-tags and allow for the inclusion of more difficult prepositions such as "to".

A second weakness of the methodology is the way in which target words are inserted into the confusion sets as well as all of the other possible erroneous versions. This is done in order to regulate how many times an error is introduced into the training data. Due to the random sampling in the final step of the process, it is possible that the amount of error is not exactly what is wanted. For example, if a training corpus with 80% preposition error is generated, 20% of the counts in the confusion sets for each preposition will be the same word. Each time an error is encountered when introducing error, there is a 20% chance that it will remain the same. This does not mean that there will be a guaranteed 20% of error in the final corpus.

Table 5.1 shows five examples of prepositions being introduced into the 2015 News Crawl corpus. The majority of prepositions are left alone, due to the presence of the source word in the confusion sets. This results in some sentences not having any changes at all, and other sentences containing several prepositions only have some of them changed.

One final issue with the methodology is the lack of control over deletions and insertions. Insertions were not included because it was difficult to consistently select the correct place into which a preposition or determiner should be inserted. This is instead implemented in Chapter 6, where all error types are introduced, and the position of the insertions is less specific. Including "NULL" words in the confusion sets resulted in words being liberally deleted in the corpora. This is not necessarily a disadvantage, as the model still learns where there are particular errors with prepositions or determiners, but the artificial data was more believable in Chapter 6, when the number of sentences with deletions was kept to 26%, which correlates with the amount of deletions in the Lang-8 corpus.

5.2 Experiments

5.2.1 Single Error Corpora

The first experiment aims to analyse the effect of error density in the training set when testing on test sets, which themselves have varying error densities. This is done by generating several sets of training data containing varying amounts of error. The first set contains solely preposition errors at error densities of 20%, 40%, 60% and 80%. These corpora are used to train four different models which correct specifically designed test sets explained in Section 5.3. The scores of each of these models on each of the test sets give an indication as to how many errors should be introduced into the data in order to get the best performance for each test set. Some qualitative analysis is also given to provide a different angle for the results.

Using the exact same methodology as with prepositions, confusion sets for determiners are created. Using them, error corpora containing 20%, 40%, 60% and 80% of errors are generated. Treated in the same way as prepositions, the scores reveal whether the correction of artificial determiner error differs at all from preposition error. Again, qualitative analysis will shed light on how and why the two error types might diverge in this sense.

5.2.2 Combined Error Corpora

Error corpora containing both preposition and determiner error are created. In this case, 20%, 40%, 60% and 80% of all prepositions and determiners are replaced with alternatives sampled from the confusion sets derived from Lang-8. This is to see how effective it is to combine error types into one corpus when training a GEC model. This is partly in order to analyse the effect of prepositions and determiners, and partly to test the potential for creating an open error-type correction model using error-specific heuristics. In Section **??** we discussed the potential for tackling most of the errors presented in the CoNLL-2014 Shared Task, and it seems unlikely that it will be a viable option. Nonetheless, these results could confirm this.

5.2.3 Comparison of Background Corpora

Although we follow the same methodology as Cahill et al. (2013), our revision data is extracted from the Lang-8 corpus, while theirs is taken from Wikipedia. As discussed in Section 2.3.4, there are various drawbacks to using this source. The data is unreliable, and can be edited by anyone. The process of extracting revision data is also a difficult one, not only involving very large XML files, but also involving a lot of time investment scraping each article. The lack of time and resources, and the availability of the more accessible Lang-8 corpus caused the decision to not make use of it. On the other hand, the revision data taken from Wikipedia is 50 times the size of the revision data taken from Lang-8.

The first experiment checks which corpus is more suited as a background corpus in this task. Using the revision data made available by Cahill et al. (2013), confusion sets are created using the same methodology as with Lang-8, explained in the above Section 5.1. Both confusion sets are for prepositions only. These two sets of confusion sets are used to generate two sets of error corpora, in which 20%, 40%, 60% and 80% of prepositions are altered according to the aforementioned error introduction procedure.

It is worth noting that Cahill et al.'s research does not include the empty "NULL" preposition, meaning that errors in which a preposition is missing are not accounted for. By contrast, in our work we include every case in which a preposition is added

in that we add the "NULL" preposition to the confusion sets, allowing for words to be deleted when introducing error.

5.3 Evaluation

In order to test the effects of mismatching error density and type between training and test data, each model is tested on specially created test sets with varying amounts of error in them. Cahill et al. (2013) found that the highest scores came from models both trained and tested on similar error rates. Our research aims to build on this finding.

The first test sets are made using Lang-8, which is also used to create the confusion sets for the training data. Specifically, only the sentences in which prepositions, determiners, and a mix of both are corrected are included. These sentences are mixed with corrected sentences, where the revised sentence is used as both source and target, to varying degrees. In each case, 1000 sentences of erroneous data are mixed with either 4000, 1500, 666, or 250 sentences of "correct" English, also taken from Lang-8. This is in order to create test sets in which 20%, 40%, 60%, and 80% of sentences are erroneous, similar to the training data. Table 5.2 shows the test sets created out of the Lang-8 corpus.

The NUCLE corpus (Ng et al., 2014), described in Section 2.3.1, was used as training and test sets for the CoNLL-2014 Shared Task Ng et al., 2014 on GEC, and since then has been commonly used in the field for comparison with previous work. The NUCLE corpus is used in our research in order to generate test sets from a different domain, despite those test sets being smaller. Again, prepositions, determiners and a combination of both are extracted and mixed with corrected sentences from the same corpus. Due to the smaller amount of relevant errors, as many sentences containing each error as possible are taken. For prepositions, this amounts 332 sentences, for determiners, 595 sentences, and for both, 169 sentences. Table 5.3 shows the test sets created out of the NUCLE corpus.

Whereas the M^2 score is reported for most experiments in this thesis, the nature of the custom test sets makes it difficult to report on these experiments. This is because the M^2 scorer is taken directly from the CoNLL-2014 Shared Task and is made for

Test sets	Error type	Error Rate	Size
test-l8p20	Preposition	20%	5000
test-l8p40	Preposition	40%	2500
test-l8p60	Preposition	60%	1666
test-l8p80	Preposition	80%	1250
test-l8d20	Determiner	20%	5000
test-l8d40	Determiner	40%	2500
test-l8d60	Determiner	60%	1666
test-l8d80	Determiner	80%	1250
test-l8b20	Both	20%	5000
test-l8b40	Both	40%	2500
test-l8b60	Both	60%	1666
test-l8b80	Both	80%	1250

TABLE 5.2: Lang-8 Corpus test sets

Test sets	Error type	Error Rate	Size
test-np20	Preposition	20%	1660
test-np40	Preposition	40%	830
test-np60	Preposition	60%	553
test-np80	Preposition	80%	415
test-nd20	Determiner	20%	2975
test-nd40	Determiner	40%	1487
test-nd60	Determiner	60%	992
test-nd80	Determiner	80%	744
test-nb20	Both	20%	845
test-nb40	Both	40%	423
test-nb60	Both	60%	282
test-nb80	Both	80%	211

TABLE 5.3: NUCLE Corpus test sets

the NUCLE test set only. It would require sizeable adaptation to make it function for these custom test sets. For this reason, only the GLEU score is reported for this approach.

5.4 Results and Discussion

5.4.1 Single Error Corpora

For prepositional errors, the GLEU scores of the four different models are in Table A.2, and the results are plotted in Figure **??**. The first thing of note is that all scores across the board decrease as the error in the test sets increase. This indicates that these models are simply not very good at correcting errors. The more errors to be corrected, the worse the models perform. This trend is predictable, as this approach is far from perfect, however there is one other factor at play. NMT systems often have problems with out-of-vocabulary words. Names, numbers and large, unusual words often do not fall under the scope of a limited vocabulary. Attention

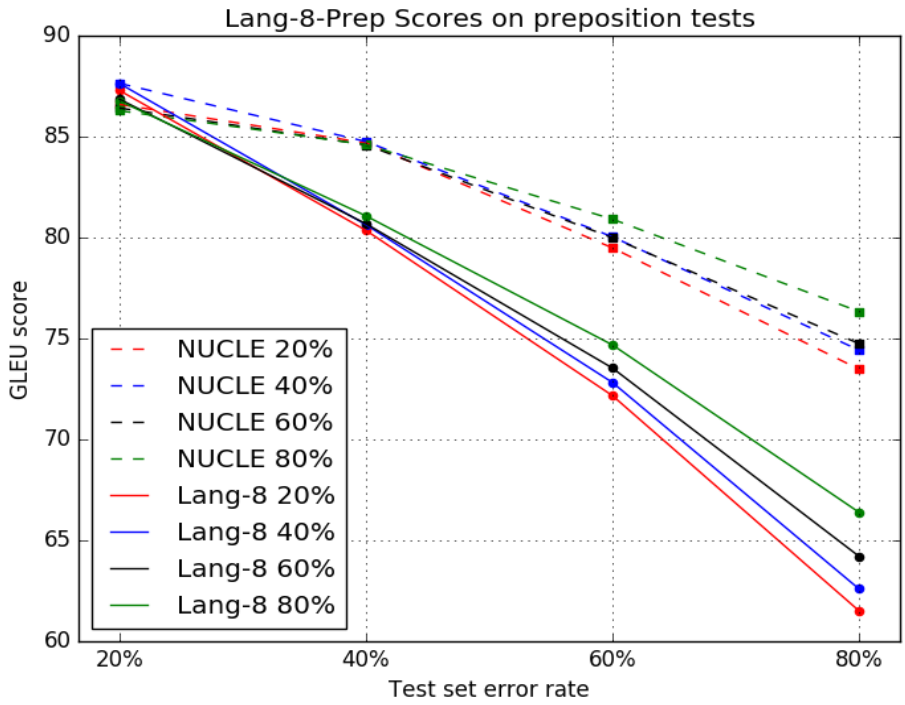

FIGURE 5.1: Plot of the data in Table A.2

mechanisms have helped alleviate this problem by assigning these translations the word from the source sentence with the largest attention. So, instead of translating the unknown word, it is mostly left alone.

When using NMT for GEC, this becomes even more of a problem, due to spelling errors. The vast amount of spelling errors made by learners of English are almost all out-of-vocabulary, and thus need to be dealt with. The solution of using a hybrid model which combines word-based and character-based information has already been tested by Ji et al. (2017) with promising results, however this is out of the scope of this project. A work-around was made for the open error approach in Section 6.2.

In this experiment, outside of the normal out-of-vocabulary options, only spelling errors within prepositions are an issue. Because this issue was first identified when tackling all errors, the work-around was made for those experiments, and not for these. Nonetheless, OpenNMT has an option which allows the replacement of out-of-vocabulary words with source words taken from the attention model, and thanks to the spelling errors included in the confusion sets, it was capable of correcting many spelling errors relating to prepositions. However, it did not always function perfectly. For example, in this sentence from the JFLEG test set, the unknown word

"breaktime" is changed to "had":

I had long breaktime → I had long had

It is reasonable to assume that the decline in scores with an increase in test set error density is due to a combination of this issue with unknown words and the prepositions being incorrectly changed. Each model is trained under the same circumstances, and has to correct the same test sets, which means that whatever issues there are on non-prepositional words are shared. This means that the difference between the performances of the various models is the important part, and not necessarily the score itself.

In Figure **??**, the dotted lines depict the scores of the four preposition models on the NUCLE test set, whereas the solid lines show the scores on the Lang-8 test set. The models all perform better on the NUCLE sets, which is unexpected, because the models are trained on corpora generated from information gathered from the Lang-8 dataset. It means that the prepositions in the Lang-8 corpus must be harder to correct than those from NUCLE, which could be for a variety of reasons, given the large difference between the two corpora.

When comparing GLEU scores of models trained on data with different error densities, the scores are relatively similar. This is because adding errors into the training corpus encourages corrections for a few identified problem words in the test set. In the preliminary experiments, when GLEU scores were drastically different, it was due to models under- or over-fitting. The GLEU score, as discussed in Section 3.2.4, rewards systems for keeping the majority of the words in a sentence the same. Even when using models trained to aggressively correct many errors, the vast majority of words are kept the same, meaning the change in score will be small, but significant.

On test corpora with an error density of 80%, the trained models perform as expected. The more error present in training, the more the models are capable of correcting the many errors in the test set. However, for the 60% test set, it is not the model trained on 60% error which performs the best, but rather the 80% model. This is also true for the 40% model. When testing with 20%, the model trained on 40% has the highest score. This is evidence for the concept of "Error Inflation", introduced by Rozovskaya, Sammons, and Roth (2012), in which including more errors than

needed in the training data results in better scores.

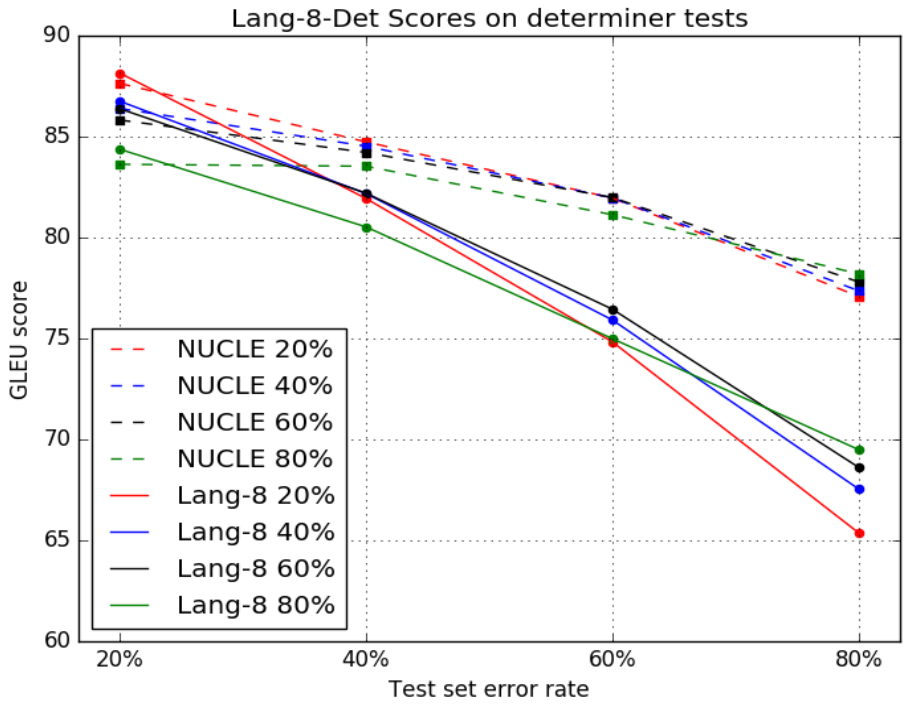

FIGURE 5.2: Plot of the data in Table A.3

Testing on determiner errors revealed similar results. The results are provided in Table A.3, and plotted in Figure **??**. Again, the models perform better on the NUCLE test sets, and the decline in scores with an increase in errors is also present when dealing with determiners. There is one main difference, which is that the concept of "Error Inflation" does not appear to be present. On the Lang-8 test sets, the best model is the one correlating to the error density of the test set.

The presence of "Error Inflation" in preposition errors, and its absence in determiner errors suggests that different error types carry different properties when used in AEG. One glaring example of this is the variability of potential preposition corrections in comparison with determiner corrections. Correcting prepositions requires more examples in the training data because there is a higher number of likely corrections than with determiners. The chances of the determiner corrections being one of 'a', 'an' or 'the' is much higher than the rarer determiners such as 'those' or 'many'.

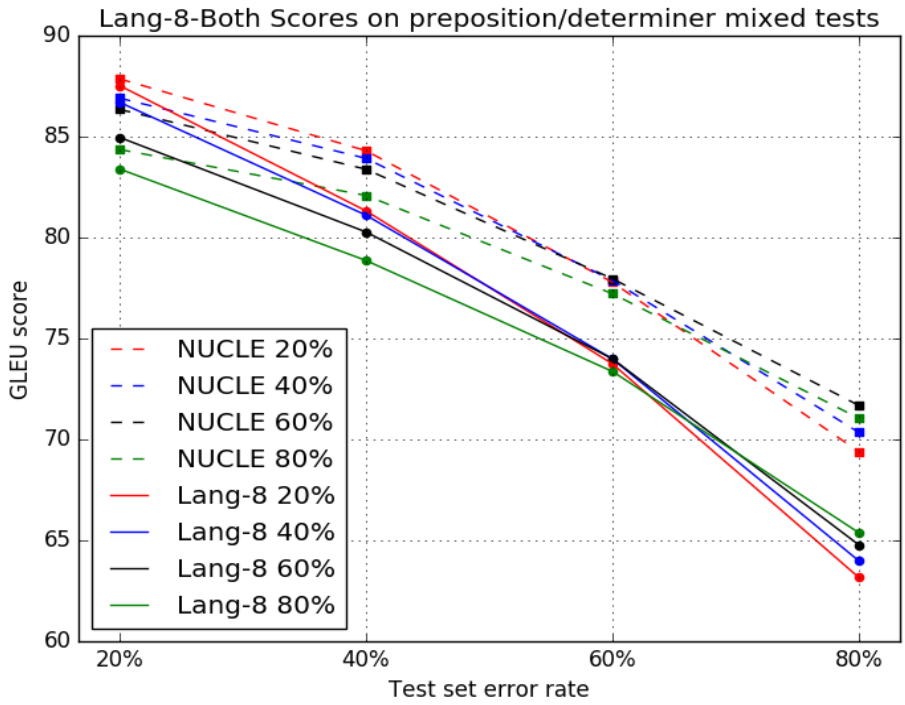

FIGURE 5.3: Plot of the data in Table A.4

5.4.2 Combined Error Corpora

The results of training models on data containing a combination of both kinds of error on combined preposition and determiner test data is shown in Table A.4 and Figure **??**. The data is consists of slightly lower scores in general, suggesting that mixing error types does not have as high a quality of correction as single errors.

For the majority of the models, it seems that the same pattern as the determiner models is followed, in which the error density of the training data correlates with that of the test data. This would then indicate that the "Error Inflation" property that may be assigned to prepositions is effectively erased in this case.

Another model of note is the model trained on 80% error. It appears to have significantly lower scores on all test sets, particularly those from the NUCLE corpus. This could be related to the fact that there are too many errors in the training corpus. When the errors were introduced, all prepositions and determiners were altered according to the confusion sets, which means that there are twice as many errors in this corpus compared to the single error experiments. The test sets should in theory also contain so many errors, but in reality, there are likely to be fewer, because a sentence

qualified as being erroneous in the test set when there was at least one preposition or determiner error. This means that several sentences with a single prepositional error and potentially several correct determiner errors were included. Such a mismatch between error densities resulted in the 80% model over-correcting.

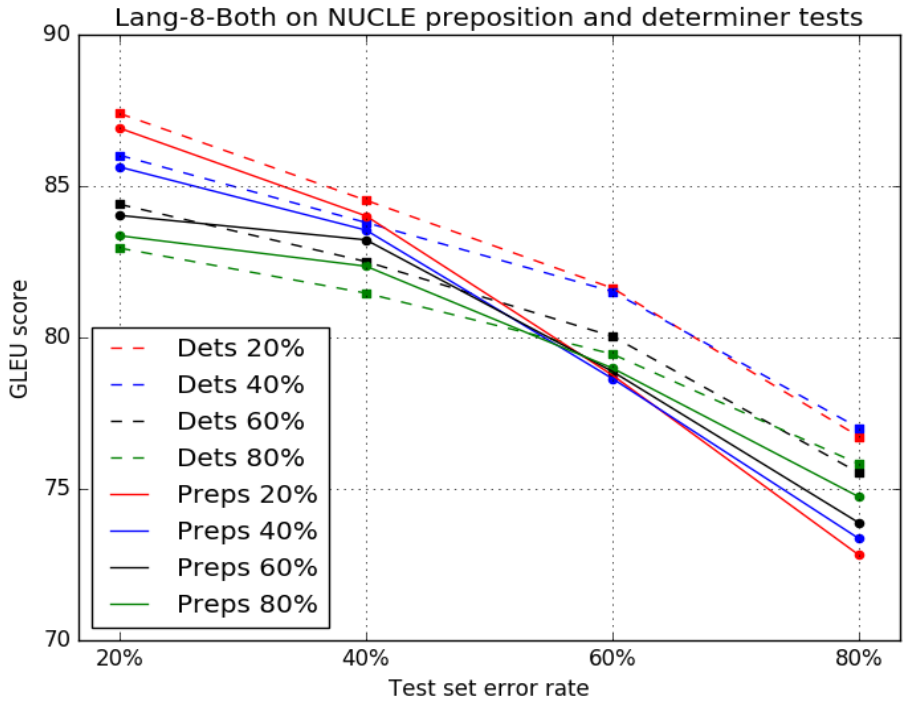

FIGURE 5.4: Plot of the data in Table A.5

This effect is also visible when looking at how the combined models perform on individual error test sets. Table A.5 and Figure **??** show how well the combined model performs on NUCLE test sets with individual error types only. In the figure, the dashed line refers to determiner error and the solid line refers to preposition error. In these cases, the combined models correct both types of error, which inevitably results in an over-correction of an unrelated error type, which explains the slightly lower scores. However, the fact that one model could almost achieve the same scores as the individual models shows that there is potential for a combination of error types in one model.

The preposition tests exhibit the "Error Inflation" phenomenon once again, whilst there is no such behaviour for the determiners. This is further proof that the nature of error types determine how well certain error densities in training data performs. The 80% error model, which likely has too many errors, performed badly on determiner

tests, and still moderately well on the preposition tests, most notably having the highest score for the 80% test set.

One more thing is clear when the two error types are directly compared in this fashion: correcting determiners appears to be the easier task, as the scores are in general higher. This can again be attributed to the varying nature of the error types. As shown in the example above, the vast majority of determiners are either "a", "an" and "the", whereas prepositions are far more diverse, which could contribute to these higher scores.

5.4.3 Comparison of Background Corpora

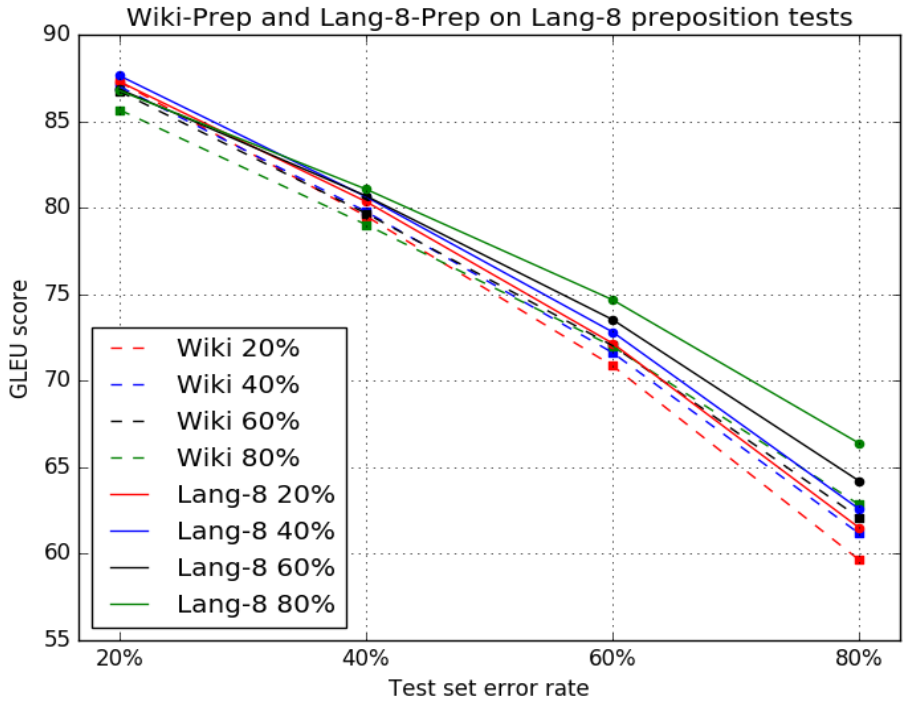

FIGURE 5.5: Plot of the data in Table A.6

Table A.6 and Figure **??** show the results of preposition models informed by Wikipedia and Lang-8 tested on Lang-8 test sets. Table A.7 and Figure **??** show the results of the same models on the NUCLE test sets.

The Lang-8 test sets pose as much of a problem to the Wikipedia models as the Lang-8 models, with very similar scores across the board. For the NUCLE test set, however, the Wikipedia models significantly outperform the Lang-8 ones. This shows

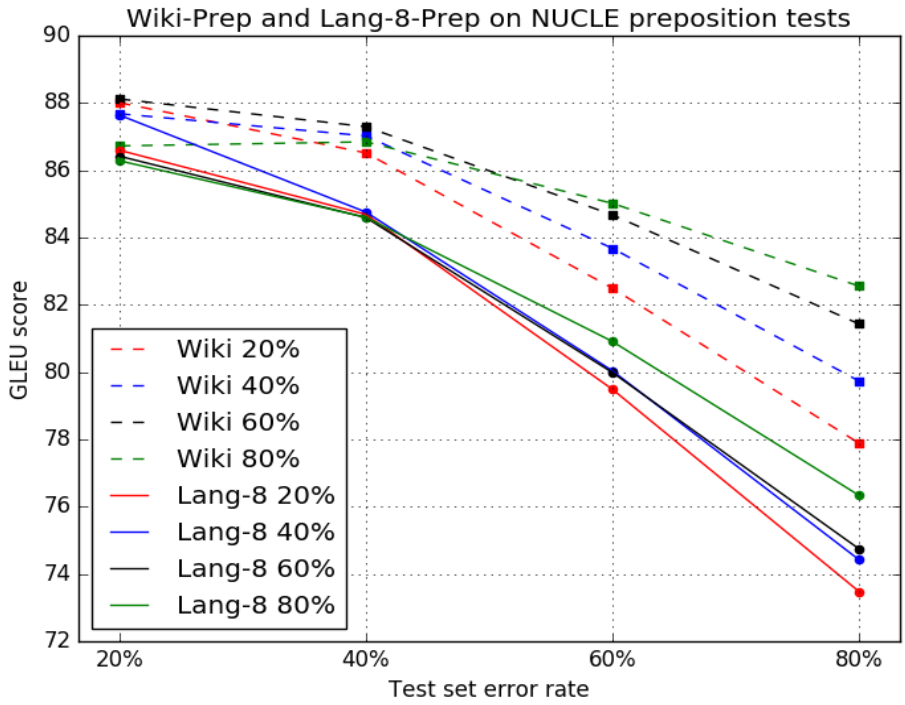

FIGURE 5.6: Plot of the data in Table A.7

two things, firstly that the test sets are indeed significantly different, as discussed already. Secondly, the Wikipedia models are better, due to the revision data being 50 times larger than that of Lang-8. This means that when crafting confusion sets, quantity is more important than quality.

5.4.4 Conclusion

Our research aims to shed light on the issue of choosing how many errors to include in artificially generated erroneous data. We find that in most cases there is a correlation between the amount of error in the training data of the best performing model and the test set's error rate. The more errors in the test set, the more errors are required in the training set. However, in the case of prepositions, the principle of "Error Inflation" applies. Including more errors than in the test set is a useful and easy way to improve results. This is not the case for determiners, and the effect also appears to disappear when combining the two errors in a single model. This, along with generally higher scores for determiners, highlights the differences between error types in this field. Future research should therefore be wary of the effects of the specific error types, as each seems to have its own unique properties. It is possible

to combine two error types together into one training set, and tackle two error types at once at test time, although the scores are not as high as when solving only individual errors. This does mean it may be possible to look towards an approach that combines many different types of errors in this way. Finally, the confusion set generated from the much larger set of Wikipedia revisions proved to yield better results than that generated from Lang-8.

Chapter 6

Open Error Correction

The field of Grammatical Error Correction aims to be able to correct ungrammatical language consistently, irrespective of error type. While the first approach detailed in this thesis is useful for gauging how many errors should be introduced into the training data, this approach alone does not achieve the end-goal. An ideal GEC application would correct all errors. Even if a deterministic approach on individual errors was taken, in which several different types of errors are introduced one after the other, only a certain number of the errors could be modeled, as discussed in Section 2.4.1.

Secondly, the vast majority of research, including this study, focuses solely on English, but in a more connected multilingual world, research should begin to focus on methods which can be adapted for several languages. Applications such as LanguageTool are in use today, and aim to correct the grammar of over 20 languages. An error-specific approach requires language-specific rules, meaning separate research needs to be done for each language. Replacing prepositions or determiners, as detailed in the previous section, is appropriate for English, but will need to be revised for other languages. Finnish, for example, attaches prepositional information to the noun, which means that noun endings would have to be targeted.

This is the motivation for a more general approach. If there was an approach to generating error which is not language-specific or indeed error type specific, then it would be possible to flexibly create training data for many languages, which would be an asset not only to the potential grammar correctors of the future, but also to popular applications of today, such as LanguageTool. This section details a novel

approach based off of the deterministic approach which aims to fulfill these requirements. The way in which the artificial data is generated is detailed in the following sections.

It is also important to evaluate how many errors need to be introduced. This research question is tackled in much the same way as the previous section, by controlling how many sentences contain an error. All experiments are detailed below in Section 6.3.

6.1 Methodology

The method undertaken in this approach is similar to the method introduced in Chapter 5, but adapted for all error types. It is similar to the recent paper by Rei et al. (2017), which takes two different approaches to generating artificial error. The first is using MT, and the second takes a pattern extraction approach. Their research is tested on error detection benchmarks, which is a different end-goal, but the pattern extraction approach is nonetheless comparable with this research's approach.

Patterns of corrected errors are collected from a background corpus, and used to generate confusion sets. These confusion sets are again sampled at random when introducing errors into an error-free corpus. Instead of just prepositions and determiners being targeted, all differences between sentences are taken. This research's major contribution is a method for including deleted phrases and inserted phrases into the generated error corpus, and not just replaced phrases. The entire methodology is explained in detail in the following sections.

6.1.1 Extracting replacements, insertions and deletions

Once again, the background corpus used in these experiments is the Lang-8 corpus. In Section 5.2.3 we learned that the larger Wikipedia is a better background corpus than Lang-8. However, extracting revision data for prepositions is a lot easier than extracting all types of error. This is because revisions in Wikipedia are not necessarily grammatical ones, but also often content-related. Extractions could result in revisions which entirely change the meaning of the sentence. Meanwhile, Lang-8

consists only of grammatical revisions. It is easily accessible and well formatted, and functions well for research purposes.

The revision data is examined using an efficient *diff* algorithm[1]. This combines the source sentence and its revision into one sentence, where each word is marked either as an "insertion" or a "deletion". For example:

I wanted to (- travel) (+ go) to the shop .

The word "travel" is marked as a deletion, and the word "go" is marked as an insertion. In this case, the extracted revision is a replacement of the word "travel" with the word "go". In many cases, however, there is no such replacement, and instead, a word or phrase is simply deleted, or inserted. Here is an example of this:

Actually , (+ he) (+ was) (+ the) (+ one) who let me know about Lang - 8 (- was) (- him) .

In this case, the entire phrase "he was the one" is marked as an insertion, and "was him" is marked as a deletion. This is because neither of the edited sections are directly adjacent to another one. In summary, corrections are labeled accordingly:

- Insertion: If an inserted phrase is not adjacent to any deletion phrases, it is classed as an insertion.
- Deletion: If a deleted phrase is not adjacent to any insertion phrase, it is classed as a deletion.
- Replacement: If an inserted phrase is adjacent to a deletion phrase, both phrases are taken and classed as a replacement. If for example a deletion is adjacent on both sides to two insertions, only the leftmost insertion is taken as its replacement, and any remaining insertions are considered separately.

An algorithm was designed to loop through each word individually and extract these replacements, insertions and deletions by iteratively adding words to insertion and deletion phrases. When a change in a word's correction status (insertion, deletion, unchanged) is detected, either an insertion, deletion or replacement is logged.

[1] `http://code.google.com/p/google-diff-match-patch`

Figure **??** shows the exact process of collecting each word into phrases and classifying them. At each step, the word is analysed along with variables:

- Rev: The current phrase being logged (either 'ins', 'del', 'rep1' or 'rep2', where the two types of replacement are insertions followed by deletions and vice versa).
- Prev: The previous word's status (Insertion: '+', Deletion: '-', Unchanged: 'O').
- Cur: The current word's status (Insertion: '+', Deletion: '-', Unchanged: 'O').

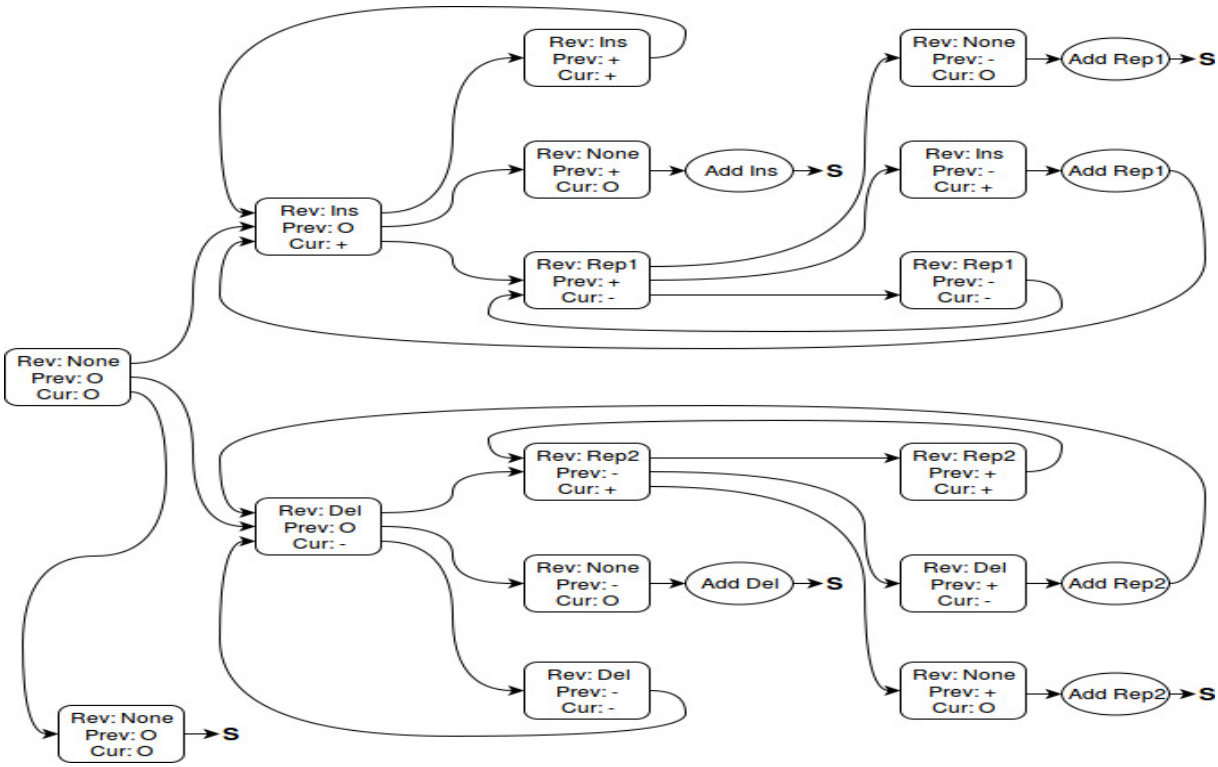

FIGURE 6.1: Flowchart depicting how insertion, deletion and replacement phrases are extracted from revision data. The "S" symbol means going back to the beginning.

Taking an approach in which every type of error comes with risks. In the example given above, "he was the one" is actually a replacement of the "was him" phrase at the end of the sentence. However, it is difficult to detect the connection between the insertion and the deletion when they are not adjacent in the correction data. For this reason, the insertions and deletions, although in many cases genuine, may in certain cases be replacements which are split up.

Type of revision	Frequency	Percentage
All errors	1100256	100%
Replacements	652517	59%
Insertions	296050	15%
Deletions	151689	26%

TABLE 6.1: Lang-8 Statistics

Lang-8 contains 509162 revised sentences, and in total 1100256 revised errors. The numbers of replacements, insertions and deletions are given in Table 6.1. Replacements cover the vast majority (59%) of the revisions, but considering that all research so far has not attempted to cover insertions and deletions, this is a remarkably low percentage.

6.1.2 Making confusion sets

Replacements

The data containing which words and phrases are replaced, inserted or deleted in the corpus will be used to inform the generation of these very errors on a new corpus. This means that the reverse of these revisions are used. In the case of replacements, if "happyness" is corrected to "happiness" in the corpus, then this information is used to replace the word "happiness" in a corpus of good English back to "happyness". Furthermore, more weight is given to replacements which are seen multiple times in the corpus, meaning that if "happyness" is corrected into "happiness" more than any other word, it will most likely be selected as the appropriate error. The selection is taken from random sampling from a probability distribution. The probability distribution is informed by the confusion sets generated from the data extracted from Lang-8. By doing this, it will be possible to emulate the errors found in the revision corpus on an entirely new corpus.

Each replacement is collected into a data structure which organises them into confusion sets. Each replaced phrase has a confusion set associated with it, consisting of all of the erroneous phrases which were corrected into it in the revision data. Each possible source phrase has a probability assigned to it informed by its frequency in the corpus. For example, the confusion set for the word "happiness", a corrected word, contains both "happyness" and "happy", which have both been seen to be corrected into "happiness". In Lang-8, "happy" was corrected to "happiness" 30 times, while

"happyness" was corrected to "happiness" only 5 times. Their respective probabilities reflect this frequency, meaning that when new errors are generated, the word "happiness" more 6 times more like to be replaced with "happy" than "happyness".

On top of listing all the possible revisions for each replacement, the amount of error that would be introduced in the final corpus is controlled by these confusion sets. In each confusion set, the corrected word itself is also planted in, at varying frequencies, depending on an input. The input percentage dictates help how many errors are introduced by adjusting the frequency of the revision not taking place. For example, if the input is 10%, it would mean that there is a 10% chance of the error being introduced, because in 90% of the cases, the targeted word would be replaced by itself. In the first approach showcased in this thesis, this could be seen as a weakness, because it is not possible to guarantee that the appropriate number of errors are generated when the confusion set is sampled at random. However, in the context of replacing entire phrases, there is an advantage to doing so. It is possible to give a bias to longer phrases, which are less likely to appear. A 5 word phrase found in the error-free corpus would then have a higher chance of being replaced in the process of error introduction than a popular single-word replacement because the same word phrase would have a lower probability of appearing in the confusion set.

"because of the"	**Frequency**
cuz	1
about	1
beacuse	1
because of the	30
becouse	1
by	11
by this	1
due to	1
either by very	1
for	6
for its	1
for my	1
in	3
in order to this	1
in price	1
is about	1
owing to	1
scheduled	1
since	1
that is rerated to	1
to me by its	1
under	1
was to go to	1
with	4
with a	1

TABLE 6.2: Confusion set for the phrase "because of the"

An example of a replacement confusion set is given in Table 6.2 for the phrase "because of the". This phrase was added as a replacement for 44 phrases in the Lang-8 corpus, the most popular of which being the word "by" with a frequency of 11. In this example, the same phrase "because of the" is put in the confusion set as well with a frequency of 30 in order to ensure that it is selected 40% of the time. This value can be changed depending on the experiment.

Insertions and Deletions

The confusion sets do not contain the NULL word, because insertions and deletions must be handled differently. When a word is deleted in the revision data, it needs to be inserted into the new data in order to replicate the error. However, it is impossible to know where in the sentence these phrases which were previously deleted should be put. This motivates the following approach, which takes the immediate context of the deletion into account. Likewise, when a word or phrase is inserted in the revision data, it needs to be deleted from the new data. However, this also poses problems. Common words, such as "a", have a high probability of being deleted according to Lang-8, but deleting the word "a" at random will not make natural-seeming errors. Making use of insertion data also requires the immediate context.

When considering insertions and deletions, the context which is taken into account is simply the words appearing immediately before and after the phrase. This can also include beginning- or end-of-sentence markers. By doing so, information about the point in the sentence at which the insertions and deletions should be made is retained. If any more context is taken, then the chances of the particular context appearing in a new corpus becomes very low, which is why only two words were chosen. The insertions and deletions are then represented like replacements, for example:

Insertion: "go shop" $\rightarrow$ "go to the shop"

Deletion: "some of food" $\rightarrow$ "some food"

By homogenising the way all three types of revision are handled, it made it possible to treat all of the errors similarly. This means that the insertions and deletions were also sorted into confusion sets in much the same way as replacements. The difference is that since each context is unlikely to occur many times in Lang-8, the vast majority of examples are unique. The only effect of this is that there is little variability in how often some phrases are deleted or inserted compared to others.

This method is similar to the recent paper by Rei et al. (2017). In their experiments, they take only replacements from their background corpus, and generate confusion sets using the POS tags of the words immediately before and after each word. This

has the advantage of populating their confusion sets with more examples and thus, more information. However, POS tags come with their own disadvantages as discussed in Section 5.1.

When introducing the error, deleting words and phrases which were originally inserted into the background corpus did not pose a problem. An example of this is the word "will", taken from this sentence in Lang-8:

I hope someone (+ will) see my diary and correct it .

In this case, if the phrase "someone will see" is found in the error-free corpus, there is a chance of it being replaced with "someone see", thereby replicating the original error, potentially in new contexts.

Some particularities had to be controlled when generating the confusion sets. An example of this are the insertions added between the "." symbol and the end of the sentence. These were often comments added by the annotators, which would be out of place when added to an error-free corpus. For this reason these particular insertions were not included in the confusion sets. An example of this is the following, which appeared in Lang-8:

I received the acceptance letter and scholarship from the Berklee College of Music . (+ This) (+ is) (+ good) (.)

Creating the confusion set for the deletions from the background corpus which should be re-inserted into the error-free corpus was also straightforward. An example of this is as follows:

I should (- to) study again .

In this case, the phrase "should study", when encountered in the error-free corpus, has a chance of being replaced by "should to study".

6.1.3 Introducing error

New errors are generated by taking an entirely new corpus, and introducing errors taken from Lang-8 into it. When generating errors, the aim was for the distribution

of replacements, insertions and deletions to be the same as in Lang-8, because this is the most informed look available for the kinds of errors to be introduced, and how often. Having learned that the random sampling of the confusion sets can lead to unpredictable error rates, the rates of replacements, insertions and deletions were controlled simply by adjusting the number of sentences in the training data with each of the three options. This way, the bias for longer phrases to be changed is kept, and the distribution of different kinds of correction is in keeping with the distribution from the Lang-8 corpus, shown in Table 6.1.

Algorithm 1 below shows the way in which error is introduced into the corpus. Each line is handled separately, and treated with both sentence markers on either end. Then a subset of the revision data is taken and iterated over. This subset is simply the revisions which contain at least one of the "corrected" target words or phrases which are present in the sentence. Taking an appropriate subset of the data prevented the system from iterating over the entire revision data at each step.

Replacements are replaced first, then the insertions are deleted, and finally the deletions are inserted. At any point, a word that has already been changed is ignored, preventing the possibility of a good replacement being deleted. A limit of 2 replacements, 1 deletion and 1 insertion was used, because in rare cases, sentences were

transformed beyond recognition.

```
for line in corpus do
    add sentence markers take shortlist of replacement data;
    for replacement in shortlist do
        replace(replacement);
    end
    take shortlist of insertion data;
    for insertion in shortlist do
        delete(insertion);
    end
    take shortlist of deletion data;
    for deletion in shortlist do
        insert(deletion);
    end
end
```

Algorithm 1: Procedure for introducing error into an error-free corpus.

6.1.4 Examples of generated error

Table A.8 in the Appendix shows a random subset of 10 sentences from the 2015 News Crawl, and examples of those sentences again with error introduced into them. Here are some observations about this subset of the data:

- Sentence length is variable, but errors are added appropriately nonetheless. Some long sentences contain errors at the beginning, and some at the end.
- Some sentences, both short and long, are ignored. This is fine, as error density on a sentence level can be controlled by removing or adding already correct sentences to the corpus.
- There are insertions and deletions, but the majority of the errors introduced are replacements.
- Some errors are believable, such as “came to attention” becoming “become attention”.

- Other errors are not, such as "helped secure the 2012 Olympic and Paralympic Games for the capital" receiving an insertion to become "helped secure the 2012 Olympic and Paralympic Games for **gratitude toward** the capital"

Ideally, all introduced errors would resemble the errors made by a typical non-native speaker of English. In order to achieve this, revision data and its context is taken from examples of this English. However, such a small context still means that many errors which are introduced are out of context. This is particularly the case for insertions, which only have two words of context before and afterwards. In many cases, those two context words are common words such as "for" and "the" which do not contain very much information. This means that when such common words are found in the corpus, many unrelated insertions could be added. Any future research on inserting errors into a corpus in this fashion should be aware of the amount of context which is included.

Although the goal of this research is to generate non-native-like English, it could be argued that in the framework of neural machine learning, it is not necessarily this which will provide the best results. The network learns to generalise based on the examples given to it, and as long as the target side of the training data is correct English, it can in theory learn to correct all kinds of error, not necessarily just non-native-like English. The example given above with the irrelevant insertion is still a valid correction, even though it is hard to believe that a non-native speaker would write it. Earlier experiments resulted in machine errors being introduced, due to under-fitting of the data. Words were repeated unnecessarily, and symbols were inserted at random into the sentence. Including "bad" errors such as these into a training data of such size as 10 or 20 million sentences will still help train a valid GEC model. In fact, it is possible that a model that can better generalise than one trained uniquely on supervised data, could perform better.

6.2 Dealing with out-of-vocabulary words

As discussed in Section 5.4.1, there are a large number of out-of-vocabulary words which do not appear in the training data when dealing with learner English. Spelling errors constitute the vast majority of them. When dealing with single errors, this

was not too much of an issue due to the ability to ignore the rest of the sentence, and focus only on prepositions or determiners. When dealing with all errors however, there are many spelling errors across all word types which are simply not seen in the training data. Although the neural network is capable of correcting some errors, those which it does not recognise are left as unknown words. There are several potential solutions for this:

- Take character-based information into account. This is what is done by Ji et al. (2017), whose model combined both character-based and word-based inputs at once. However, this project uses OpenNMT as the model, and this does not change, so that the effect of the different training sets can be compared.
- Use a subword embedding such as BPE, which splits larger words into smaller sections in order to reduce the number of unknown words. This does not however solve the problem of the numerous and various spelling errors present in learner English.
- Put unknown spelling errors through a separate spelling corrector and replace them accordingly. If there is no correction, simply return the source word.

The final option was chosen, because it allowed the most amount of control over the data.

6.2.1 Peter Norvig's Spelling Correction

Peter Norvig released a blog describing a simple spelling corrector[2]. The corrector is widely praised for its small size, ease of use, and speed. It is not of the same calibre as some of the spelling correction services available today, but it purported to achieve 80 to 90% of the accuracy of Google's industrial sized correction system. This was chosen as a fast and easy way to deal with unknown words[3].

[2] `http://norvig.com/spell-correct.html`

[3] The following extension of Norvig's implementation was used: `https://github.com/pirate/spellchecker`

6.2.2 Implementing Spelling Corrections

After the model trained on artificial data has corrected the test data, the output looks like this:

Source: Sanitasion is keye for to cure problems with disese .

Target: <unk> is the key to curing problems with <unk> .

In this example, "is keye for to cure" is corrected into "is the key to curing". This is an example of a fluency edit which a neural network is capable of doing, despite the fact that "keye" is out of vocabulary. This means that despite there being three unknown words in the source sentence, only two of them must be corrected in the target sentence.

Firstly, all of the corrections of the unknown source words are prepared. In this case, "sanitasion" is corrected into "sanitation", "keye" is corrected into "key", and "disese" is corrected into "disease". Then, the <unk> symbols are associated with the corresponding misspelling the source sentence by finding the one with the nearest position in the sentence. This puts the correct correction in the appropriate target positions the vast majority of the time.

6.3 Experiments

Three different experiments were carried out using this approach to error generation. The first experiment aimed to boost supervised data with artificial data, and compare the results with those obtained from using just supervised data on its own. Boosting supervised data in this way has shown promising results in NMT: Sennrich, Haddow, and Birch (2016) successfully used back-translated monolingual data to double the size of their supervised parallel corpus to improve results.

The second experiment saw noise being introduced into the same corpus three times. Due to the random sampling involved in the process, each corpus featured different errors. The experiment aimed to see if variation caused by the stochastic sampling affects the results in a large way.

Training corpus	**GLEU - JFLEG**	**GLEU - NUCLE**	**M^2 - NUCLE**
Lang-8	49.56	61.30	28.89
Lang-8 + 70%	52.01	66.16	35.74
Lang-8 + 80%	51.66	65.55	35.84
Lang-8 + 90%	51.39	65.09	35.13
CoNLL-2014 Winner	–	64.84	37.33
Yuan & Briscoe (2016)	–	–	39.90

TABLE 6.3: GLEU scores and M^2 scores of models trained on different available supervised datasets

The third experiment examined the relationship between the amount of error introduced into the training error and the amount of corrections made on the test set. Training data containing error densities between 0% and 100% was tested on the JFLEG corpus. Examples were examined to see how effective the different models were at correcting English grammar. Furthermore, similar to the experiments in Section 5, this experiment also aimed to see if the effect of "Error Inflation" also applied in a more general context. All experiments made use of the 2015 News Crawl from the WMT Translation Task, as described in Section 2.3.5, as the error-free corpus.

6.3.1 Boosting supervised data

We used the automatically generated data to boost the supervised data. We took approximately the same number of sentences present in the Lang-8 corpus from the artificially generated data, and combined it with the entirety of the Lang-8 corpus, effectively doubling its size. This was done with three different types of data, one including 70% percent of erroneous sentences, one with 80%, and one with 90%. Considering the fact that the Lang-8 corpus has an error density of 98%, this means that the total error densities of the models were 84%, 89%, and 94% respectively.

The results are given in Table 6.3. Both evaluation metrics show an increase in performance when adding artificial data to the already existing corpus. This could be for one of two reasons:

- The artificial data provides the necessary boost in training data size. The Lang-8 corpus is impressively large, but not large enough to make a consistently precise GEC system on its own.

- The artificial data helps the model generalise over a wider range of errors. As seen in Section 6.1.4, some errors introduced artificially do not resemble human error. They are nonetheless valid corrections, and it could be that learning how to correct a wider range of error than just "human" error helps the system tackle new errors, which are written by humans, better.

The JFLEG test set has an error density of 85%, which explains why the Lang-8 + 70% model achieves the best score on this test set, as the trained error density of 84% is the closest to this value. The NUCLE test set's error density is lower, at 72%. The GLEU score reflects this by rewarding the training data with fewer errors, but the M^2 reports a slight improvement on the model with 89% error. Crucially, the difference in scores across different error densities is minimal, which suggests a weak link between error densities in training and test sets when supervised corpora are being boosted.

According to the GLEU scores of the models tested on the NUCLE corpus, the artificially boosted models outperform the winner of the CoNLL-2014 Shared Task. However, the M^2 score reveals that the results still do not compete with neither this benchmark, nor the NMT system trained on the CLC corpus. No matter which evaluation metric is used, the fact that there is an improvement means that using artificial data to boost a larger corpus such as the CLC corpus could result in improved scores.

6.3.2 Checking consistency

The methodology used in this research for generating error uses random sampling, which means that one particular sentence could have error introduced in many different ways. One question that arises from this is whether generating error in this way is consistent. Over 10 million artificial errors are created, but what if a different version of those errors performed better?

In order to answer this, an experiment was created in which the News Crawl had error introduced into it three different times. This resulted in three different versions of the same sentences, but each time having introduced different errors. The three corpora were used to train five separate models each according the standard

methodology reported in Section 6.1, and were compared against each other. Each corpus had five corresponding models trained on varying error densities between 60% and 100%. The GLEU score on the JFLEG corpus is reported in Figure **??**.

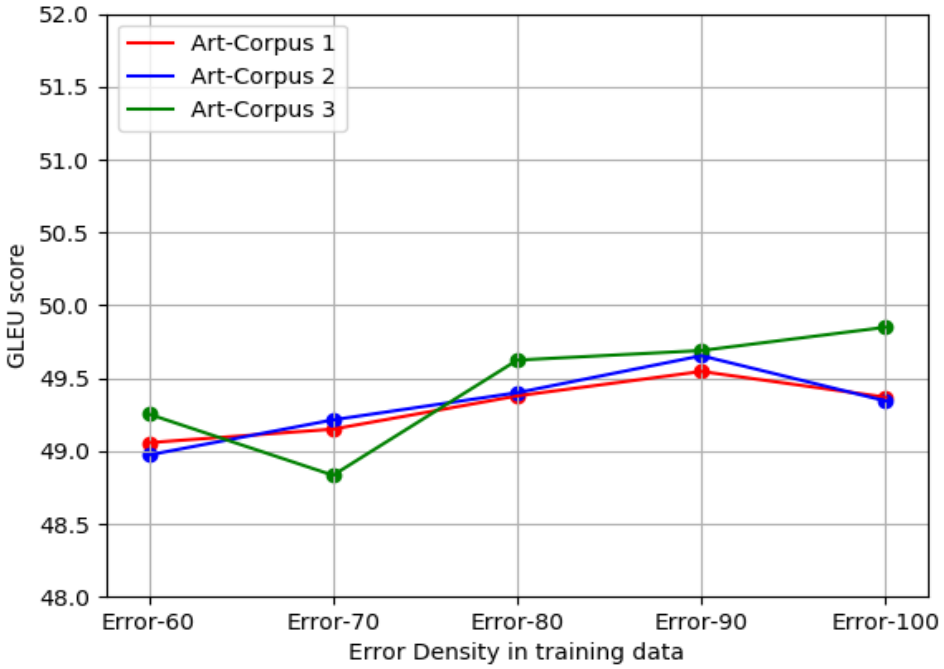

FIGURE 6.2: The GLEU score of three models each trained on artificial corpora with five different error densities. Tested on the JFLEG corpus.

The idea behind this experiment was that the models should learn to generalise the revision data learned from Lang-8 in all three cases. Since the new corpus used was over 20 times as large as Lang-8, it was conceivable that on average, all of the errors would be included in the new corpus around 20 times more often than in Lang-8. Despite the fact that the errors were randomly sampled, the fact that the entire distribution was sampled 20 times over suggests that the error distribution would be preserved.

At any error density, the largest difference between two models was 0.5 points in the GLEU score. The graph shows a consistency between the three training corpora. The third artificial corpus seemed to showcase a little more variation, particularly for an error rate of 70%. Nonetheless, this corpus achieved the highest scores overall, and was thus chosen as the corpus for the next experiment on error densities.

6.3.3 Using as standalone data

These experiments were designed primarily to see how possible it is to use solely artificially generated training data to train a GEC model. Because the only input required is monolingual English data, the size of the training is no longer a problem. By simply copying the errors extracted from Lang-8 into many different contexts, it is possible that the training data can cover more potential errors in total. Due to the sheer number of models being trained, only 10 million of the News Crawl sentences were used, which is shown in Section 4.3.2 to have the same effect as 20 million sentences.

It was found from the first approach taken in this thesis that the amount of error introduced into artificial data for GEC plays a very important role in how a neural model corrects errors at test time. It is advantageous to have control over how often changes are made, because one single model will never perform optimally across different error rates. The ideal GEC model would leave one hundred sentences written in perfect English alone, and entirely correct the next one hundred riddled with errors. However, measures of how "correct" a sentence should be and the nature of its correction are stored in a continuous vector space, which means that it can never be black and white. The results from simply changing prepositions reflects this. In fact, it was learned that the relationship between the error density of the training data and the test data is complex. Prepositions required the use of "error inflation" in order to achieve the best results, whereas this was not the case for determiner errors.

In this experiment, the models shown in Table 6.4 were tested against the JFLEG and NUCLE corpora. There were 11 models each trained on an artificial corpus. Each artificial corpus combined correct English sentences being left alone, and incorrect English sentences being corrected. The error rate shows the percentages of the sentences in the error corpus being corrected.

Table 6.5 shows the results of the 11 models on the two test sets. The scores are also represented in Figure **??**. The first thing to note is that the GLEU scores hardly vary across different error densities, whereas the M^2 score shows a more predictable trend. The GLEU score's variation is narrower, but more significant. As discussed

Name of model	Error rate
TRAIN-0	0%
TRAIN-10	10%
TRAIN-20	20%
TRAIN-30	30%
TRAIN-40	40%
TRAIN-50	50%
TRAIN-60	60%
TRAIN-70	70%
TRAIN-80	80%
TRAIN-90	90%
TRAIN-100	100%

TABLE 6.4: Models that were trained from the training sets generated from the artificial error corpus

Training corpus	GLEU JFLEG	GLEU NUCLE	M^2 NUCLE
TRAIN-0	47.01	65.47	16.98
TRAIN-10	47.63	66.07	16.87
TRAIN-20	48.21	66.13	19.30
TRAIN-30	48.29	66.39	22.79
TRAIN-40	48.55	66.09	25.68
TRAIN-50	48.61	66.29	25.81
TRAIN-60	49.25	65.35	28.13
TRAIN-70	48.83	66.46	29.27
TRAIN-80	49.62	66.49	29.33
TRAIN-90	49.69	66.35	30.98
TRAIN-100	49.85	66.16	29.88
CoNLL-2014 Winner	–	64.84	37.33
Yuan & Briscoe (2016)	–	–	39.90

TABLE 6.5: GLEU scores and M^2 scores of models trained on different available supervised datasets

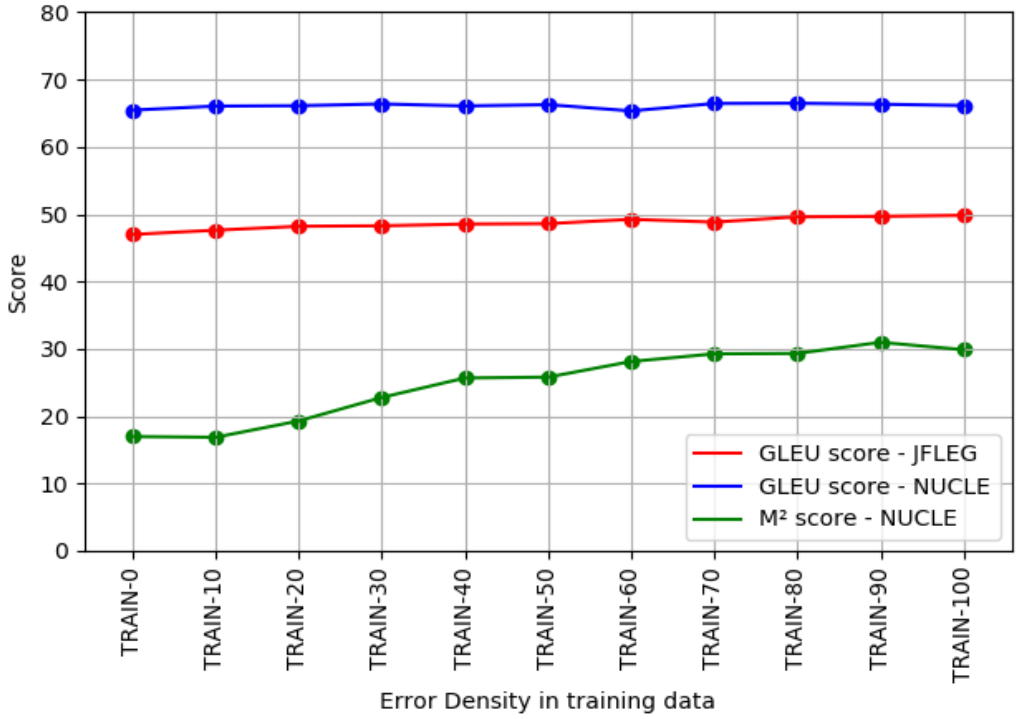

FIGURE 6.3: GLEU scores and M^2 scores of 11 models each trained on artificial corpora with different error densities.

in Section 3.2.4, the GLEU score rewards systems for leaving most of the sentence unchanged. When comparing the outputs of models trained with different error densities, some models will change more things than others, but most of the words are nonetheless left the same, even on TRAIN-100. As a result, the GLEU score does not change very much. The M^2 score, however, is based on the edits which are made, which means the unchanged words are not even a part of the equation. This is why the increase in performance is so much stronger according to the M^2 score.

Considering that the NUCLE corpus has an error density of 72%, it follows that models trained on 70% or 80% of error should perform the best on this test set. The M^2 score does reveal that training with lower error densities is not suitable for this test set—scores are up to 14 points lower than the highest. However, the highest M^2 score came from the model trained on 90% of error. This is slightly higher than the test set's error density, which suggests that "Error Inflation", as discussed in Chapter 5 does indeed apply when considering all types of error at once. The GLEU score provides a less clear picture of how the error density affects the output. On the fluency-based JFLEG test set, a similar trend emerges, with the best results being from TRAIN-100, despite the error density of the test set being 85%. The GLEU scores according to the NUCLE corpus do not reveal any correlation, as all models performed on average very similarly. This is likely due to the fact that there are fewer edits in total in the NUCLE corpus, since it is only focused on minimal edits. In the JFLEG corpus, the fluency edits result in more rephrasing, which in turn results in more words being changed.

Since the evaluation metrics, although indicative of performance, are not entirely representative of how correct the output is, we evaluate some examples seen in Tables 6.6, 6.7, 6.8 and 6.9. In each example, the outputs of all of the 11 models are given alongside the source sentence, and the four reference sentences available with the JFLEG corpus.

In the first example, all four annotators agree that there are three errors which need correcting: "Forexample" should become "For example", "My" should become "my", and "12years" should become "12 years". The first and last errors are both examples of failing to use a space between two words, thus creating an out-of-vocabulary word. The models are sometimes capable of successfully dealing with out-of-vocabulary

Source	Forexample , My cousin is 12years old .
TRAIN-0	Forexample , My cousin is years old .
TRAIN-10	Forexample , My cousin is years old .
TRAIN-20	Forexample , My cousin is years old .
TRAIN-30	Forexample , My cousin is 12 .
TRAIN-40	Forexample : My cousin is 12 .
TRAIN-50	Forexample : My cousin is 12 years old .
TRAIN-60	Forexample , My cousin is 12 years old .
TRAIN-70	Forexample : My cousin is 12 years old .
TRAIN-80	My cousin is 12 .
TRAIN-90	Forexample : My cousin is 12 .
TRAIN-100	Forexample .
Reference-1	For example , my cousin is 12 years old .
Reference-2	For example , my cousin is 12 years old .
Reference-3	For example , my cousin is 12 years old .
Reference-4	For example , my cousin is 12 years old .

TABLE 6.6: Example 1

words, as discussed in Section 6.2.2, which is shown in TRAIN-50, TRAIN-60 and TRAIN-70 when "12years" is corrected. When it does not, the spelling correction system deals with them instead. In this example, it corrects "12years" into "years". The higher the error density, the more likely it is that the model can deal with the out-of-vocabulary word. Meanwhile, the word "Forexample" remains unchanged. This could be due to the lack of context. "12 years" is surrounded by the relatively specific construction "is — old", whereas "Forexample" is only behind a comma. In fact, the models which make more aggressive corrections replace this comma with a colon, which shows a lack of contextual understanding.

In the second example, all four annotators do not approve of the use of "having" as being the appropriate verb when talking about doing through a change. This is an example of a sentence which is grammatically correct, but not fluent. None of the models, even those which are more likely to make changes deem the sentence to be in need of correction. The word "having" has a large amount of potential contexts in which it could be deemed "correct", which means that common words such as this are less likely to be identified as problem words. For example, the sentence would be fluent if the word "change" was replaced with "party", or a whole range of other nouns.

Example 3 contains some more traditionally easier to correct errors. The word "tends" should agree with "People" and not have the "s". "medias" is acceptable both in plural and singular, and "litterature" is a spelling error. The models successfully correct

Source	Nevertheless , this city is having a big change .
TRAIN-0	Nevertheless , this city is having a big change .
TRAIN-10	Nevertheless , this city is having a big change .
TRAIN-20	Nevertheless , this city is having a big change .
TRAIN-30	Nevertheless , this city is having a big change .
TRAIN-40	Nevertheless , this city is having a big change .
TRAIN-50	Nevertheless , this city is having a big change .
TRAIN-60	Nevertheless , this city is having a big change .
TRAIN-70	Nevertheless , this city is having a big change .
TRAIN-80	Nevertheless , this city is having a big change .
TRAIN-90	Nevertheless , this city is having a big change .
TRAIN-100	Nevertheless , this city is having a big change .
Reference-1	Nevertheless , this city is going through a big change .
Reference-2	Nevertheless , this city is experiencing a big change .
Reference-3	Nevertheless , this city is undergoing a big change .
Reference-4	Nevertheless , this city is undergoing a big change .

TABLE 6.7: Example 2

Source	People tends to choose other medias , and that is why litterature is in danger .
TRAIN-0	People tends to choose other medias , and that is why literature is in danger .
TRAIN-10	People tends to choose other media , and that is why literature is in danger .
TRAIN-20	People tends to choose other media , and that is why literature is in danger .
TRAIN-30	People tend to choose other media , and that is why literature is in danger .
TRAIN-40	People tends to choose other media , and that is why literature is in danger .
TRAIN-50	People tend to choose other media , and that is why literature is in danger .
TRAIN-60	People tends to choose other media , and that is why literature is in danger .
TRAIN-70	People tend to choose other media , and that is why literature is in danger .
TRAIN-80	People tend to choose other media , and that is why literature is in danger .
TRAIN-90	People tend to choose other media , and that is why literature is in danger .
TRAIN-100	People tend to choose other media , and that is why literature is in danger .
Reference-1	People tend to choose other medias , and that is why literature is in danger .
Reference-2	People tend to choose other types of media and that is why literature is in danger .
Reference-3	People tend to choose other media types , and that is why literature is in danger .
Reference-4	People tend to choose other media , which is why literature is in danger .

TABLE 6.8: Example 3

these errors, and show a steady improvement as the training data becomes more erroneous. All models use the spelling mechanism to correct "litterature", TRAIN-30 is the first to correct "tends", and from TRAIN-10 onwards is "medias" reduced to "media", probably due to the fact that the plural is less common in English. Despite the successful correction, the output matches perfectly with none of the reference sentences, which serves as a reminder that even with four fluency edits by different annotators, it is hard to write all possible correct versions of a sentence.

The 4th example showcases more of the behaviour of the models. The phrase "these thing seem" is successfully corrected from TRAIN-10 onwards, due to the specific nature of the context. When there are two words denoting "plural" either side of a singular noun, it is relatively easy for the system to see. The word "be" is inserted

before the word "hard", which suggests that using insertions in the training data makes a difference. However, none of the reference sentences deem this to be correct, despite it being so. The word "hardly" is changed into "hard", which is not the correct English, but shows evidence that the model tends to know the word "hardly" only in different contexts. "have health body" has plenty of possible corrections, and the different models try adding in determiners (TRAIN-40), and replacing "health" with an adjective (TRAIN-60, TRAIN-100). However, it fails to notice the problem with "this thing like sports". This might be due to the fact that certain faulty constructions are very unlikely to appear in the training corpus. If very few people on Lang-8 wrote the phrase "this thing like", then it would also rarely appear in the training data. On top of this, as seen in example 2, common words such as "thing" tend to be seen as already correct by the system.

These four example show that the models are very capable of learning how to deal with errors, despite what their GLEU scores and M^2 scores might say. The references do not always acknowledge the correct changes made by the system, which reflects the nature of the task. Similar to Natural Language Generation, there is no one correct answer, and the evaluation metrics do not entirely respect that. Other errors successfully which would have taken up too much space include the following:

- Deletions: "I like about that" $\rightarrow$ "I like that"
- Spelling errors which are not out-of-vocabulary: "we are planing to" $\rightarrow$ "we are planning to"
- Inserting and deleting punctuation: "Besides it is better" $\rightarrow$ "Besides , it is better"

There are however still many issues left unsolved by the models:

- Spelling errors which Norvig's spelling correct cannot tackle, such as "pargarghs". A more sophisticated system could improve this, and more training data could help the neural network pick up on the stranger spelling errors.
- Capitalisations in the middle of the sentence, or making the first word lowercase. This is due to the appearance of some words primarily at the beginning of sentences. One option would be to use only lowercase data, but many

Source	But these thing seem to hard for old people , they move so hardly , and do n't have health body to play this thing like sports .
TRAIN-0	But these thing seem to hard for old people , they move so hardly , and do n't have health body to play this thing like sports .
TRAIN-10	But these things seem to hard for old people , they move so hard , and do n't have health body to play this thing like sports .
TRAIN-20	But these things seem to hard for old people , they move so hard , and do n't have health body to play this thing like sports .
TRAIN-30	But these things seem to hard for old people , they move so hard , and do n't have health body to play this thing like sports .
TRAIN-40	But these things seem to be hard for old people , they move so hard , and do n't have the health body to play this thing like sports .
TRAIN-50	But these things seem to be hard for old people , they move so hard , and do n't have health body to play this thing like sports .
TRAIN-60	But these things seem to be hard for old people , they move so hard , and do n't have a great body to play this thing like sports .
TRAIN-70	But these things seem to be hard for old people , they move so hard , and do n't have health body to play this thing like sports .
TRAIN-80	But these things seem to be hard for old people , they move so hard , and do n't have health body to play this thing like sports .
TRAIN-90	But these things seem to be hard for old people , they move so hard , and do n't have a health body to play this thing like sports .
TRAIN-100	But these things seem to be hard for old people , they move so hard , and do n't have a healthy body to play this thing like sports .
Reference-1	But these things seem too hard for old people , they move so difficultly , and do n't have a healthy body to play things like sports .
Reference-2	But these things seem too hard for old people , they move so slowly , and do n't have health bodies to use this thing like athletes .
Reference-3	But these things seem too hard for old people since they hardly move and do n't have healthy bodies to play things like sports .
Reference-4	These thing seem too hard for old people , because they move with so much difficulty and do n't have health bodies to play sports .

TABLE 6.9: Example 4

acronyms and company names would be made lower case, which counts as an error, according to the CoNLL-2014 Shared Task.

- Although the system deals well with number, it struggles with mass nouns which often require different determiners.

6.3.4 Conclusion

It is clear that using Artificial Error Generation to create open-error correction data is a valid way of creating training data for an NMT-based GEC system. These experiments show that using such data to boost already available supervised data can boost results according to standard evaluation metrics. It is also possible to use the data as standalone data, and it performs almost as well as the boosted data. Finally, evidence of "Error Inflation" is present in the results, and there appears to be little difference between error corpora when sampled differently from the same confusion sets.

Chapter 7

Conclusions

This thesis aimed to contribute to the field of Grammatical Error Correction by analysing the potential of using Artificial Error Generation to create training data for use in a Neural Machine Translation system which corrects grammar. We explored a new methodology for generating errors involving replacing, deleting and inserting errors according to confusion sets extracted from a background corpus. We also explored the effect of introducing different amounts of error into a corpus of correct English.

The research was successful in that the models trained on supervised data boosted with artificial data, and artificial data alone gave encouraging results. According to two benchmark evaluation metrics, doubling the size of supervised training data by supplementing it with artificial data increased performance. Also, using a very large amount of artificial data on its own was shown to have only slightly worse results than the supervised data. This validates AEG as an approach to the problem of a lack of available training data, and is very promising for several reasons:

- Introducing error into monolingual data has the potential to be used in other languages. Although all research involving NMT for GEC systems has so far been done on English, there is demand for GEC systems across all languages. Only a small amount of supervised data is required as the background corpus. In this research, the Lang-8 corpus was used as the background corpus, and proved to be of suitable size and quality. However, the Lang-8 corpora in other languages are considerably smaller. On the other hand, the website was crawled to generate this corpus in 2011, which means that the data has only

increased since then. Wikipedia was also discussed as a potential background corpus, and although difficult to use, has much content in many languages.

- Artificial data allows for much more control over the data than supervised data. It is possible to control how many corrections are made, for instance, and which corrections appear more often than others. This allows a lot of room for adaptation. If a specific model is wanted which aims to correct English with a special emphasis on missing determiners, perhaps due to a user base of people whose native languages do not use determiners, it is easy to adjust the training data to adapt accordingly. It also allows for domain adaptation, in case the desired system should only correct sentences from a closed domain.

- It also makes compensating for "Error Inflation" possible. Prepositions were shown to be an example of an error type which requires more errors to be included in the artificial data than the test set in order to be appropriately corrected. Determiners did not seem to exhibit the same behaviour. However, in the open error experiments, the models with the highest scores were those trained on data with more errors than the test sets. AEG allows for more errors to be included into a training set in order to compensate for this effect.

- It is possible to target specific errors. In this research, prepositions and determiners were examined, but more "difficult" errors could in theory be implemented as well, in order to make up for gaps in a rule-based system. NMT is shown to be capable of targeting specific errors well, and could target the errors for which it is difficult to write manual rules.

Despite the potential of the AEG approach, there are still some drawbacks, which prevent its use in today's applications:

- The output is still inconsistent. Neural Machine Translation is very well developed, but the same volume of data available for multilingual translation is simply not available for GEC. Even the best models explored in this research did not consistently make appropriate corrections.

- Evaluating GEC models remains very difficult, as there are many corrections for any given erroneous sentence. The vast majority of research is based on the

M^2 score of a model tested on the NUCLE corpus. Both the evaluation metric and the test set are outdated, and not representative of the problem. The GLEU score and the JFLEG corpus both improve on this, but still do not always reflect what a qualitative analysis of examples reveals.

- Certain error types are difficult to deal with in the context of open error correction. Some examples of this include extreme spelling errors, capitalisation and mass nouns. However, the models are capable of picking up information which might be difficult for rule-based systems, such as long-distance grammatical effects between words which are far apart.

- User-generated content needs to be used to generate the background corpus, particularly in the case of other languages. Only in English is there some correction data created by experts in controlled settings, and even then there is not very much. This is why the Lang-8 corpus was used in this thesis, and why Wikipedia appears to be the best possible source for a background corpus. The problem is that user-generated content is not as trustworthy as supervised data generated specifically for research purposes.

Despite these complications, Automatic Error Generation has a huge potential in the field of Grammatical Error Correction, thanks to its ability to increase and to control the training data for Neural Machine Translation systems.

Appendix A

Appendix - Tables of results

TABLE A.1: Table taken from Ng et al., 2014 listing the different error types used in the CoNLL-2014 Shared Task. The list is ordered according to the percentage rates of how often each error type occurs according to the NUCLE corpus.

Error Type	**Example**
ArtOrDet	It is obvious to see that [internet → the internet] saves people time and also connects people globally.
Wci	Early examination is [healthy → advisable] and will cast away unwanted doubts.
Rloc-	It is up to the [patient's own choice → patient] to disclose information.
Nn	A carrier may consider not having any [child → children] after getting married.
Vt	Medical technology during that time [is → was] not advanced enough to cure him.
Mec	This knowledge [maybe relavant → may be relevant] to them.
Prep	This essay will [discuss about → discuss] whether a carrier should tell his relatives or not.
Wform	The sense of [guilty → guilt] can be more than expected.
SVA	The benefits of disclosing genetic risk information [outweighs → outweigh] the costs.
Vform	A study in 2010 [shown → showed] that patients recover faster when surrounded by family members.
Trans	It is sometimes hard to find [out → out if] one has this disease.
Um	Genetic disease has a close relationship with the born gene. (i.e., no correction possible without further clarification.)
Pref	It is everyone's duty to ensure that [he or she → they] undergo regular health checks.
Sun	The issue is highly [debatable, a → debatable. A] genetic risk could come from either side of the family.
WOinc	[Someone having what kind of disease → What kind of disease someone has] is a matter of their own privacy.
Cit	Poor citation practice.
Wtone	[It's → It is] our family and relatives that bring us up.
Others	An error that does not fit into any other category but can still be corrected.
Spar	We must pay attention to this information and [assisting → assist] those who are at risk.
Vm	Although the problem [would → may] not be serious, people [would → might] still be afraid.
V0	However, there are also a great number of people [who → who are] against this technology.
Ssub	This is an issue [needs → that needs] to be addressed.
WOadv	In conclusion, [personally I → I personally] feel that it is important to tell one's family members.
Sfrag	However, from the ethical point of view.
Npos	Someone should tell the [carriers → carrier's] relatives about the genetic problem.
Pform	A couple should run a few tests to see if [their → they] have any genetic diseases beforehand.
Wa	After [WOWII → World War II], the population of China decreased rapidly.
Smod	[Undeniable, → It is undeniable that] it becomes addictive when we spend more time socializing virtually.

Test Sets	20%	40%	60%	80%
test-l8p20	87.29	87.63	86.85	86.77
test-l8p40	80.35	80.63	80.68	81.07
test-l8p60	72.17	72.82	73.54	74.68
test-l8p80	61.51	62.60	64.21	66.39
test-np20	86.60	87.64	86.42	86.28
test-np40	84.69	84.76	84.59	84.60
test-np60	79.49	80.04	79.99	80.91
test-np80	73.48	74.43	74.75	76.34

TABLE A.2: GLEU score according to how much **preposition** error in training data informed by Lang-8, tested on test sets with varying amounts of error from **Lang-8 and NUCLE**.

Test Sets	20%	40%	60%	80%
test-l8d20	88.16	86.76	86.39	84.38
test-l8d40	81.94	82.18	82.20	80.53
test-l8d60	74.83	75.92	76.45	74.99
test-l8d80	65.36	67.54	68.61	69.49
test-nd20	87.64	86.40	85.82	83.63
test-nd40	84.74	84.53	84.22	83.53
test-nd60	81.95	81.96	81.98	81.13
test-nd80	77.08	77.36	77.79	78.20

TABLE A.3: GLEU score according to how much **determiner** error in training data informed by Lang-8, tested on test sets with varying amounts of error from **Lang-8 and NUCLE**.

Test Sets	20%	40%	60%	80%
test-l8b20	87.53	86.70	84.96	83.40
test-l8b40	81.32	81.11	80.28	78.86
test-l8b60	73.74	73.99	73.99	73.37
test-l8b80	63.16	63.98	64.77	65.38
test-nb20	87.86	86.91	86.35	84.36
test-nb40	84.30	83.92	83.38	82.09
test-nb60	77.78	77.87	77.96	77.22
test-nb80	69.39	70.35	71.68	71.05

TABLE A.4: GLEU score according to how much **combined** preposition and determiner error in training data informed by Lang-8, tested on test sets with varying amounts of error from **Lang-8 and NUCLE**.

Test Sets	20%	40%	60%	80%
test-nd20	87.40	86.02	84.40	82.95
test-nd40	84.53	83.80	82.51	81.47
test-nd60	81.63	81.52	80.04	79.45
test-nd80	76.73	77.01	75.53	75.82
test-np20	86.91	85.63	84.03	83.36
test-np40	84.02	83.55	83.22	82.35
test-np60	78.75	78.64	78.88	78.99
test-np80	72.82	73.36	73.88	74.74

TABLE A.5: GLEU score according to how much **combined** preposition and determiner error in training data informed by Lang-8, tested separately on **NUCLE** test sets with varying amounts of **determiner** error, and then **preposition** error.

Test Sets	20%	40%	60%	80%
test-l8p20	87.27	87.00	86.72	85.66
test-l8p40	79.52	79.78	79.67	79.03
test-l8p60	70.87	71.63	72.03	71.99
test-l8p80	59.66	61.18	62.09	62.85
test-l8p20	87.29	87.63	86.85	86.77
test-l8p40	80.35	80.63	80.68	81.07
test-l8p60	72.17	72.82	73.54	74.68
test-l8p80	61.51	62.60	64.21	66.39

TABLE A.6: GLEU score according to how much preposition error in training data informed by **Wikipedia** and **Lang-8**, tested on test sets with varying amounts of **preposition** error from Lang-8.

Test Sets	20%	40%	60%	80%
test-l8p20	88.01	87.68	88.12	86.72
test-l8p40	86.52	87.02	87.29	86.84
test-l8p60	82.49	83.68	84.67	85.02
test-l8p80	77.89	79.74	81.43	82.56
test-l8p20	86.60	87.64	86.42	86.28
test-l8p40	84.69	84.76	84.59	84.60
test-l8p60	79.49	80.04	79.99	80.91
test-l8p80	73.48	74.43	74.75	76.34

TABLE A.7: GLEU score according to how much preposition error in training data informed by **Wikipedia** and **Lang-8**, tested on test sets with varying amounts of preposition error from **NUCLE**.

Before Error Introduction	**After Error Introduction**
The 25-year-old competed in 34 races during the 2013 and 2014 seasons , scoring the first-ever **championship** points for the Manor-Marussia team by finishing ninth at last year 's Monaco Grand Prix .	The 25-year-old competed in 34 races during the 2013 and 2014 seasons , scoring the first-ever **championships** points for the Manor-Marussia team by finishing ninth at last year 's Monaco Grand Prix .
To be doing **what you** love with your mates **gives** plenty of reason to smile .	To be doing **how 's** love with your mates **give** plenty of reason to smile .
Dinner is served from 5pm until 5.45pm .	Dinner is served from 5pm until 5.45pm .
Boehner sings **for the** cameras	Boehner sings **at** cameras
The London mayoral candidate , part of the team that helped secure the 2012 Olympic and Paralympic Games for the capital , branded the coalition government 'wicked and negligent ' .	London mayoral candidate , part of the team that helped secure the 2012 Olympic and Paralympic Games for **gratitude toward** the capital , branded the coalition government 'wicked and negligent ' .
The proposal for a 20-storey apartment tower in Woolloogabba at 64 Logan Road includes 174 apartments along with one level of retail space and five levels of car parking .	The proposal for a 20-storey apartment tower in Woolloogabba at 64 Logan Road includes 174 apartments along with one level of retail space and five levels of car parking .
The map , stitched together from **images** taken by **the New** Horizons space probe as it makes its journey towards the dwarf planet , reveals the intriguing markings on the strange world in detail .	The map , stitched together from **image** taken by **new** Horizons space probe as it makes its journey towards the dwarf planet , reveals the intriguing markings on the strange world in detail .
If you 're more bullish , **then** maybe even up to 25 % .	If you 're more bullish , maybe even up to 25 % .
Maple first **came to** attention in 2007 after winning Channel 4/Saatchi gallery 's New Sensations prize .	Maple first **became** attention in 2007 after winning Channel 4/Saatchi gallery 's New Sensations prize .
Greece is also accused of having squandered the years since its bailout : it has failed to reform its bloated pensions system and failed to push through privatisations or reform the notoriously inefficient and complicated tax system .	Greece is also accused of having squandered the years since its bailout : it **also** has failed to reform its bloated pensions system and failed to push through privatisations or reform the notoriously inefficient and complicated tax system .

TABLE A.8: 10 sentences taken from the 2015 News Crawl before and after having error introduced into them.

Bibliography

Bird, Steven, Ewan Klein, and Edward Loper (2009). *Natural Language Processing with Python*. 1st. O'Reilly Media, Inc. ISBN: 0596516495, 9780596516499.

Brockett, Chris, William B. Dolan, and Michael Gamon (2006). "Correcting ESL Errors Using Phrasal SMT Techniques". In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. ACL-44. Sydney, Australia: Association for Computational Linguistics, pp. 249–256. DOI: 10.3115/1220175.1220207. URL: http://dx.doi.org/10.3115/1220175.1220207.

Cahill, Aoife et al. (2013). "Robust Systems for Preposition Error Correction Using Wikipedia Revisions". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, pp. 507–517. URL: http://www.aclweb.org/anthology/N13-1055.

Chollampatt, Shamil, Duc Tam Hoang, and Hwee Tou Ng (2016). "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*. Austin, Texas: Association for Computational Linguistics, pp. 1901–1911. URL: https://aclweb.org/anthology/D16-1195.

Dahlmeier, Daniel and Hwee Tou Ng (2012). "Better Evaluation for Grammatical Error Correction". In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Montréal, Canada: Association for Computational Linguistics, pp. 568–572. URL: http://www.aclweb.org/anthology/N12-1067.

Dahlmeier, Daniel, Hwee Tou Ng, and Eric Jun Feng Ng (2012). "NUS at the HOO 2012 Shared Task". In: *Proceedings of the Seventh Workshop on Building Educational*

Applications Using NLP. Montréal, Canada: Association for Computational Linguistics, pp. 216–224. URL: http://www.aclweb.org/anthology/W12-2025.

Dahlmeier, Daniel, Hwee Tou Ng, and Siew Mei Wu (2013). "Building a Large Annotated Corpus of Learner English: The NUS Corpus of Learner English". In: *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*. Atlanta, Georgia: Association for Computational Linguistics, pp. 22–31. URL: http://www.aclweb.org/anthology/W13-1703.

Daudaravicius, Vidas et al. (2016). "A Report on the Automatic Evaluation of Scientific Writing Shared Task". In: *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*. San Diego, CA: Association for Computational Linguistics, pp. 53–62. URL: http://www.aclweb.org/anthology/W16-0506.

Devlin, Jacob et al. (2014). "Fast and Robust Neural Network Joint Models for Statistical Machine Translation". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, pp. 1370–1380. URL: http://www.aclweb.org/anthology/P/P14/P14-1129.

Felice, Mariano (2016). *Artificial error generation for translation-based grammatical error correction*. Tech. rep. UCAM-CL-TR-895. University of Cambridge, Computer Laboratory. URL: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-895.pdf.

Felice, Mariano et al. (2014). "Grammatical error correction using hybrid systems and type filtering". In: *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*. Baltimore, Maryland: Association for Computational Linguistics, pp. 15–24. URL: http://www.aclweb.org/anthology/W14-1702.

Heilman, Michael et al. (2014). "Predicting Grammaticality on an Ordinal Scale". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Baltimore, Maryland: Association for Computational Linguistics, pp. 174–180. URL: http://www.aclweb.org/anthology/P14-2029.

Hoang, Duc Tam, Shamil Chollampatt, and Hwee Tou Ng (2016). "Exploiting N-Best Hypotheses to Improve an SMT Approach to Grammatical Error Correction". In: *CoRR* abs/1606.00210. URL: http://arxiv.org/abs/1606.00210.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Comput.* 9.8, pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Imamura, Kenji et al. (2012). "Grammar Error Correction Using Pseudo-error Sentences and Domain Adaptation". In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*. ACL '12. Jeju Island, Korea: Association for Computational Linguistics, pp. 388–392. URL: http://dl.acm.org/citation.cfm?id=2390665.2390752.

Izumi, Emi et al. (2003). "Automatic Error Detection in the Japanese Learners' English Spoken Data". In: *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*. Sapporo, Japan: Association for Computational Linguistics, pp. 145–148. DOI: 10.3115/1075178.1075202. URL: http://www.aclweb.org/anthology/P03-2024.

Ji, Jianshu et al. (2017). "A Nested Attention Neural Hybrid Model for Grammatical Error Correction". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, pp. 753–762. URL: http://aclweb.org/anthology/P17-1070.

Junczys-Dowmunt, Marcin and Roman Grundkiewicz (2014). "The AMU System in the CoNLL-2014 Shared Task: Grammatical Error Correction by Data-Intensive and Feature-Rich Statistical Machine Translation". In: *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*. Baltimore, Maryland: Association for Computational Linguistics, pp. 25–33. URL: http://www.aclweb.org/anthology/W/W14/W14-1703.

Klein, G. et al. "OpenNMT: Open-Source Toolkit for Neural Machine Translation". In: *ArXiv e-prints*. eprint: 1701.02810.

Lee, Lung-Hao et al. (2016). "The NTNU-YZU System in the AESW Shared Task: Automated Evaluation of Scientific Writing Using a Convolutional Neural Network". In: *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*. San Diego, CA: Association for Computational Linguistics, pp. 122–129. URL: http://www.aclweb.org/anthology/W16-0513.

Liu, Zhuoran and Yang Liu (2016). "Exploiting Unlabeled Data for Neural Grammatical Error Detection". In: *CoRR* abs/1611.08987. URL: http://arxiv.org/abs/1611.08987.

Madnani, Nitin, Joel Tetreault, and Martin Chodorow (2012). "Exploring Grammatical Error Correction with Not-So-Crummy Machine Translation". In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. Montréal, Canada: Association for Computational Linguistics, pp. 44–53. URL: http://www.aclweb.org/anthology/W12-2005.

Mizumoto, Tomoya and Yuji Matsumoto (2016). "Discriminative Reranking for Grammatical Error Correction with Statistical Machine Translation". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, pp. 1133–1138. URL: http://www.aclweb.org/anthology/N16-1133.

Napoles, Courtney, Keisuke Sakaguchi, and Joel Tetreault (2017). "JFLEG: A Fluency Corpus and Benchmark for Grammatical Error Correction". In: *Proceedings of the 2017 Conference of the European Chapter of the Association for Computational Linguistics*. Valencia, Spain: Association for Computational Linguistics. URL: https://arxiv.org/abs/1702.04066.

Napoles, Courtney et al. (2015). "Ground Truth for Grammatical Error Correction Metrics". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, pp. 588–593. URL: http://www.aclweb.org/anthology/P15-2097.

Ng, Hwee Tou et al. (2013). "The CoNLL-2013 Shared Task on Grammatical Error Correction". In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*. Sofia, Bulgaria: Association for Computational Linguistics, pp. 1–12. URL: http://www.comp.nus.edu.sg/~nlp/conll13st/CoNLLST01.pdf.

Ng, Hwee Tou et al. (2014). "The CoNLL-2014 Shared Task on Grammatical Error Correction". In: *Proceedings of the Eighteenth Conference on Computational Natural*

Language Learning: Shared Task. Baltimore, Maryland: Association for Computational Linguistics, pp. 1–14. URL: http://www.aclweb.org/anthology/W14-1701.

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

Rei, Marek and Helen Yannakoudakis (2016). "Compositional Sequence Labeling Models for Error Detection in Learner Writing". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 1181–1191. URL: http://www.aclweb.org/anthology/P16-1112.

Rei, Marek et al. (2017). "Artificial Error Generation with Machine Translation and Syntactic Patterns". In: *CoRR* abs/1707.05236. URL: https://arxiv.org/pdf/1707.05236.pdf.

Rozovskaya, Alla and Dan Roth (2010). "Training Paradigms for Correcting Errors in Grammar and Usage". In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Los Angeles, California: Association for Computational Linguistics, pp. 154–162. URL: http://www.aclweb.org/anthology/N10-1018.

— (2014). "Building a State-of-the-Art Grammatical Error Correction System". In: *Transactions of the Association for Computational Linguistics*. San Diego, California: Association for Computational Linguistics, pp. 419–434. URL: http://www.aclweb.org/anthology/Q14-1033.

— (2016). "Grammatical Error Correction: Machine Translation and Classifiers". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 2205–2215. URL: http://www.aclweb.org/anthology/P16-1208.

Rozovskaya, Alla, Mark Sammons, and Dan Roth (2012). "The UI System in the HOO 2012 Shared Task on Error Correction". In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. Montréal, Canada: Association for Computational Linguistics, pp. 272–280. URL: http://www.aclweb.org/anthology/W12-2032.

Sakaguchi, Keisuke, Matt Post, and Benjamin Van Durme (2017). "Grammatical Error Correction with Neural Reinforcement Learning". In: *CoRR* abs/1707.00299. URL: http://arxiv.org/abs/1707.00299.

Sakaguchi, Keisuke et al. (2016). "Reassessing the Goals of Grammatical Error Correction: Fluency Instead of Grammaticality". In: *Transactions of the Association for Computational Linguistics* 4, pp. 169–182. ISSN: 2307-387X. URL: https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/800.

Schmaltz, Allen et al. (2016). "Sentence-Level Grammatical Error Identification as Sequence-to-Sequence Correction". In: *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*. San Diego, CA: Association for Computational Linguistics, pp. 242–251. URL: http://www.aclweb.org/anthology/W16-0528.

Sennrich, Rico (2016). "How Grammatical is Character-level Neural Machine Translation? Assessing MT Quality with Contrastive Translation Pairs". In: *cs.CL*. URL: https://arxiv.org/abs/1612.04629.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (2016). "Improving Neural Machine Translation Models with Monolingual Data". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 86–96. URL: http://www.aclweb.org/anthology/P16-1009.

Sidorov, Grigori et al. (2013). "Rule-based System for Automatic Grammar Correction Using Syntactic N-grams for English Language Learning (L2)". In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*. Sofia, Bulgaria: Association for Computational Linguistics, pp. 96–101. URL: http://www.aclweb.org/anthology/W13-3613.

Yuan, Zheng and Ted Briscoe (2016). "Grammatical error correction using neural machine translation". In: *Proceedings of the 2016 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California: Association for Computational Linguistics, pp. 380–386. URL: http://www.aclweb.org/anthology/N16-1042.

Yuan, Zheng, Ted Briscoe, and Mariano Felice (2016). "Candidate re-ranking for SMT-based grammatical error correction". In: *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*. San Diego, CA: Association for Computational Linguistics, pp. 256–266. URL: http://www.aclweb.org/anthology/W16-0530.