



UNIVERSITY OF
CAMBRIDGE

**Text readability and summarisation
for non-native reading comprehension**

Menglin Xia



Lucy Cavendish College

This dissertation is submitted on June 7, 2018 for the degree of Doctor of Philosophy

Abstract

Developing reading ability is an essential part of language acquisition. However, finding texts at the appropriate reading level for training second language (L2) learners and examining their reading comprehension is a demanding task for English instructors as well as the learners themselves. In this thesis, we focus on two important aspects of non-native reading comprehension: text readability assessment, which estimates the reading difficulty of a given text for L2 learners, and learner summarisation assessment, which evaluates the quality of learner summaries to assess their reading comprehension. We approach both tasks as supervised machine learning problems and present automated assessment systems that achieve state-of-the-art performance. The techniques developed for both tasks have critical applications in language learning to help learners improve their reading ability.

We first address the task of text readability assessment for L2 learners. One of the major challenges for a data-driven approach to text readability assessment is the lack of significantly-sized level-annotated data aimed at L2 learners. We present a dataset of Common European Framework of Reference for Languages (CEFR)-graded texts tailored for L2 learners and look into a range of linguistic features affecting text readability. We compare the text readability measures for native and L2 learners and explore methods that make use of the more plentiful data aimed at native readers to help improve L2 readability assessment.

We then present a summarisation task for evaluating non-native reading comprehension and demonstrate an automated summarisation assessment system aimed at evaluating the quality of learner summaries. We propose three novel machine learning approaches to assessing learner summaries. In the first approach, we examine using several NLP techniques to extract features to measure the content similarity between the reading passage and the summary. In the second approach, we calculate a similarity matrix based on sentence-to-sentence similarity between the reading passage and the summary and apply a Convolutional Neural Network (CNN) model to assess the summary quality using the similarity matrix. In the third approach, we build an end-to-end summarisation assessment model using Recurrent Neural Networks (RNNs). Further, we combine the three approaches to a single system using a simple parallel ensemble modelling technique. We show that our models outperform traditional approaches that rely on exact word match on the task and that our best model produces quality assessments close to professional examiners.

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Preface and specified in the text. It is not substantially the same as any that I have submitted, or am concurrently submitting, for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. I further state that no substantial part of my dissertation has already been submitted, or is being concurrently submitted, for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. This dissertation does not exceed the prescribed limit of 60 000 words.

Menglin Xia
June 7, 2018

Acknowledgements

I would like to express my deepest gratitude to my supervisor Professor Ted Briscoe, for his invaluable advice and support in my PhD. His patient guidance and encouragement have been of great importance to me. I am especially indebted to Dr Ekaterina Kochmar who has always been very supportive and offered me a lot of advice and help in the realization of this work. I am very grateful to my examiners, Dr Paula Buttery and Dr Advait Siddharthan, for their valuable advice in improving this thesis. I would also like to thank Professor Ann Copestake and Professor Simone Teufel for their feedback in the early stages of this work. My gratitude also extends to everyone in the research group and friends in the Computer Laboratory for their assistance and useful discussions over the past four years: Meng Zhang, Yimai Fang, Zheng Yuan, Mariano Felice, Chris Bryant, Helen Yannakoudakis, Marek Rei, Ronan Cummins, Øeistein Andersen, Youmna Farag, Ahmed Zaidi, and Yang Liu. It has been a pleasure to work with them. I would also like to acknowledge the assistance from Cambridge Assessment with the collection of my data.

Most importantly, I wish to thank my family, especially my grandparents, who have always been there for me and helped me find the strength to carry on. My special thanks go to my best friend Luwen Meng from Lucy Cavendish College, who has always been a great friend and fed me in my years in Cambridge. I would like to thank Kevin Li for proofreading this thesis. Finally, many thanks to all my friends, for their love and support, no matter where we are.

Contents

1	Introduction	15
1.1	Assessing text readability for L2 learners	16
1.2	Assessing learner summarisation for reading comprehension	18
1.3	Research questions	19
1.4	Contributions	20
1.5	Thesis outline	21
2	Background	23
2.1	Text readability assessment	23
2.1.1	Related work on readability assessment	23
2.1.2	Assessing L1 and L2 readability	26
2.1.3	Automated readability assessment for L2 learners	27
2.1.4	Overview of existing readability corpora	28
2.1.5	Readability assessment for educational applications	31
2.2	Assessing learner summarisation	33
2.2.1	Assessing reading	33
2.2.2	Automated evaluation of machine-generated summaries	35
2.2.3	Related work on automated evaluation of learner summaries	39
3	Assessing text readability	41
3.1	Introduction	41
3.2	Native data: the WeeBit corpus	41
3.3	Readability measures	42
3.3.1	Traditional features	44
3.3.2	Lexico-semantic features	44
3.3.3	Parse tree syntactic features	46
3.3.4	Language modelling features	47
3.3.5	Discourse-based features	50
3.4	Experiments	52
3.4.1	Method	52

3.4.1.1	Classification model	53
3.4.1.2	Ranking model	54
3.4.1.3	Implementation	55
3.4.2	Evaluation	55
3.4.3	Results and analysis	57
3.5	Summary	59
4	Readability assessment on L2 data	61
4.1	Introduction	61
4.2	L2 Data: the Cambridge Exams dataset	61
4.3	Experiments	63
4.3.1	Generalization of classification and ranking models	63
4.3.2	Mapping ranking scores to the CEFR levels	65
4.3.2.1	Regression-based approach	65
4.3.2.2	Learning the cut-off boundary	66
4.3.2.3	Classification-based approach	67
4.3.2.4	Results and analysis	67
4.3.3	Domain adaptation from native to L2 data	68
4.3.4	Using self-training to enhance the classification model	70
4.3.5	Comparing readability features on native and L2 data	71
4.4	Summary	72
5	Automated evaluation of learner summarisation for reading comprehension	73
5.1	Introduction	73
5.2	Summarisation task	73
5.3	Data	76
5.3.1	Simulated learner data	76
5.3.2	Real learner data	77
5.4	Method 1: Measures for summary assessment	78
5.4.1	Verbatim features	81
5.4.2	Semantic similarity features	83
5.4.3	Distributed vector representations of the summary	89
5.4.4	Discourse and other textual features	92
5.4.5	Models for assessing summary	93
5.5	Method 2: Assessing summaries with a similarity matrix	95
5.5.1	Motivation and method	95
5.5.2	The convolutional neural network model	97
5.5.3	CNN model for assessing the summary quality	99
5.6	Method 3: LSTM-based models for summary assessment	100

5.6.1	Motivation	100
5.6.2	Recurrent neural networks	101
5.6.3	Merged LSTM model	104
5.6.4	Attention-based LSTM model	106
5.7	Ensemble modelling	108
5.8	Experiments and evaluation	109
5.8.1	Experimental set-up	109
5.8.2	Evaluation metrics	110
5.8.3	Results on the simulated learner data	110
5.8.4	Results on the real learner data	112
5.8.5	Comparing the automated system with human performance	115
5.8.6	Comparing the systems with previous work	115
5.8.6.1	ETS-RUF data	116
5.8.6.2	Results on the ETS-RUF data	116
5.9	Summary	118
6	Conclusion	119
	Bibliography	123
A	Examples of the original WeeBit corpus	137
B	Examples of the modified WeeBit corpus	139
C	An example from the Cambridge English Exams data	143
D	Examples of “good” and “bad” summaries in the simulated learner data	147
E	Examples of summaries in the real learner data	149

Glossary

ACC accuracy

BNC British National Corpus

CAE Cambridge Advanced Certificate in English

CEFR Common European Framework of Reference for Languages

CLC Cambridge Learner Corpus

CNN Convolutional Neural Network

CPE Cambridge Certificate of Proficiency in English

EVP English Vocabulary Profile

FCE First Certificate in English

GR grammatical relation

GRU Gated Recurrent Unit

IDF inverse document frequency

KET Key English Test

KRR Kernel Ridge Regression

L1 first language

L2 second language

LDA Latent Dirichlet Allocation

LM language model

LSA Latent Semantic Analysis

LSTM Long Short-Term Memory

NLP Natural Language Processing

PCC Pearson Correlation Coefficient

PET Preliminary English Test

POS part-of-speech

RASP Robust Accurate Statistical Parsing

RHO Spearman's Rank-order Correlation

RMSE Root Mean Square Error

RNN Recurrent Neural Network

SVM Support Vector Machine

SVR Support Vector Regression

TF term frequency

WeeBit the WeeBit corpus

Chapter 1

Introduction

Developing reading ability is an important part of second language acquisition. Reading forms the basis of many aspects of language learning. Through reading, learners acquire vocabulary, develop grammar, and improve their ability to extract information and reconstruct the meaning of a text (Day and Bamford, 1998).

However, finding the proper reading materials for training second language (L2) learners at a specific level of proficiency and designing tasks to assess their reading comprehension is a demanding and time-consuming task for English instructors as well as the readers themselves. Therefore, a system that integrates automated reading material selection and automated generation and evaluation of reading tasks for non-native learners can be useful in helping learners to acquire reading skills efficiently. Such a system consists of two major components: a text selection module that chooses reading materials matching the reading proficiency of learners and an assessment module that examines the learner's understanding of the selected material and provides feedback to the text selection module for adapting the difficulty of reading materials for the learners. The system enhances many pedagogical applications by supporting readers in their second language education.

This thesis describes novel techniques of Natural Language Processing (NLP) to automate the assessments of the two key components of non-native reading comprehension: text readability assessment, which examines factors contributing to the reading process and aids the selection of reading materials for learners, and summarisation assessment, which measures the outcome of the reading process, or equivalently reading comprehension, and automates the evaluation of the learner's understanding.

In this chapter, we describe the motivation and challenges in the assessment of L2 text readability and the assessment of learner summarisation for evaluating reading comprehension. At the end of this chapter, we summarize the contributions of this work and give an outline of this thesis.

1.1 Assessing text readability for L2 learners

To automate the process of reading material selection, a system that focuses on text readability analysis for L2 learners can be developed to accurately predict the difficulty of a reading text and ensure that the selected text matches the reader's proficiency.

Text readability, which has been formally defined as the sum of all elements in textual material that affect a reader's understanding, reading speed, and level of interest in the material (Dale and Chall, 1949), is affected by multiple variables. These may include the style of writing, its format and organization, as well as various contextual dimensions of the text, such as its lexical and syntactic complexity, level of conceptual familiarity, logical sophistication and so on.

Except for the difference in genre, style and topic, reading difficulty of texts can differ, even if they are expressing similar content. The following example shows three texts¹ on the same topic but at different levels of readability:

Example 1: "Alice's Adventures in Wonderland" is a famous children's book. It is full of strange events and creatures. Lewis Carroll wrote "Alice's Adventures in Wonderland" in 1865.

Example 2: "Alice's Adventures in Wonderland" is one of the most famous children's books. The novel is full of charm and absurd events and creatures. It was written by Lewis Carroll in 1865.

Example 3: "Alice's Adventures in Wonderland" is one of the most famous and enduring children's classics. The novel, written by Lewis Carroll in 1865, is full of whimsical charm, and a feeling for the absurd that is unsurpassed.

An automated text readability assessment system aims to identify the reading difficulty of the texts and can be used in this situation to find the text that best matches the learner's proficiency. Figure 1.1 illustrates using readability assessment to estimate the reading difficulty of the example texts, which could be further used for reading material selection.

More formally, the task of text readability assessment is to quantify the difficulty for comprehension of a given reading material to a specific group of readers by assigning a numerical score or other form of estimation to the text (Collins-Thompson, 2014). We approach automated text readability assessment as a supervised machine learning problem, which allows us to make use of annotated data, extract features to measure different aspects of reading difficulty, and investigate the importance of the features for L2 readability.

Researchers have looked into text readability assessment for decades (Dale and Chall, 1949; Collins-Thompson, 2014). However, most previous research on text readability assessment focused on estimating text difficulty for readers with English as their first language (L1). Due to

¹The example texts are adapted from articles on <https://newsela.com/>

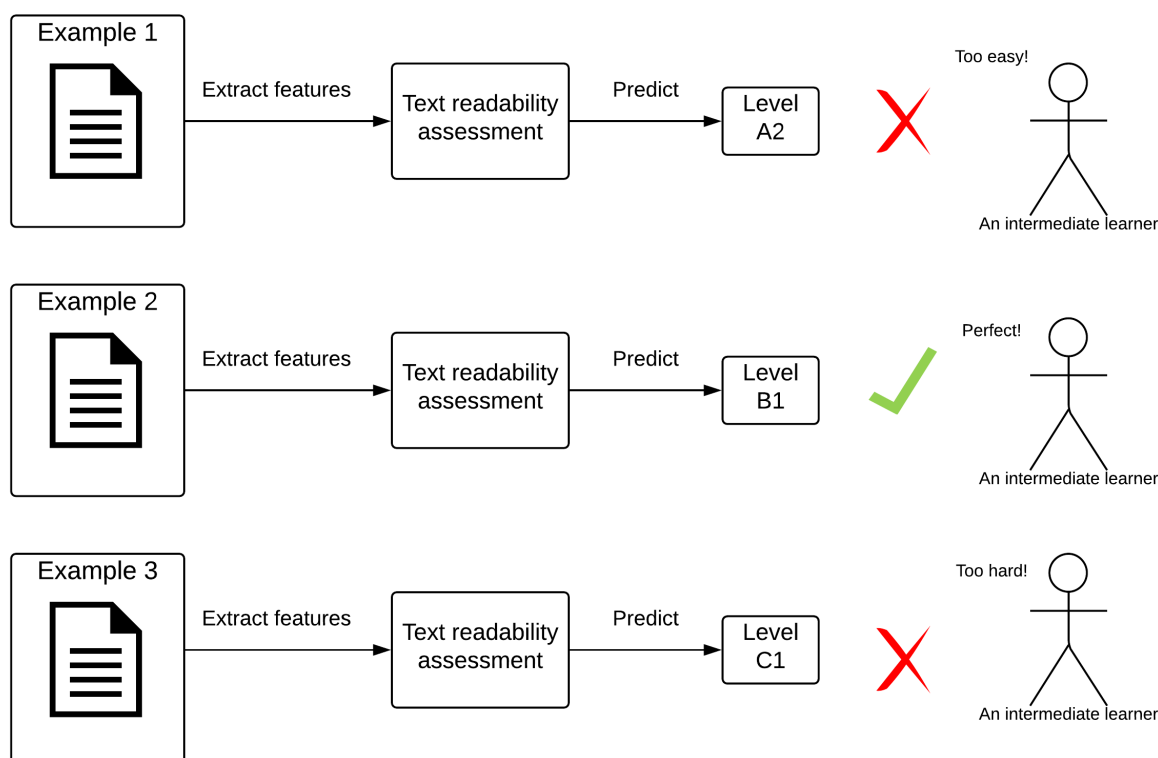


Figure 1.1: An illustration of using text readability assessment to select reading materials for L2 learners

the difference in the time and pace in language acquisition, the factors affecting readability for L2 learners often differ from those for native readers. For example, the grammatical aspects of readability usually contribute more to text comprehensibility for L2 learners than the conceptual cognition difficulty of the reading material (Callan and Eskenazi, 2007). Additionally, it is plausible to assume that the major purpose of reading for L2 learners lies in gaining language skills rather than information, entertainment, etc. Therefore, in this work, we focus on the influence of linguistic features for text reading difficulty rather than typography and literary style.

A system that is tailored towards learner's perception of reading difficulty can produce a more accurate estimation of text reading difficulty for non-native readers and thus better facilitate language learning.

In addition, one of the major challenges for a data-driven approach to text readability assessment for L2 learners is that there is not enough significantly sized, properly annotated data for this task. Most of the data available are annotated with reading levels for native readers. Therefore, it is crucial for the current work to obtain texts that are tailored for L2 learners' readability and explore methods that take advantage of existing native data to estimate readability for L2 learners. In reality, the learner's native language also influences their L2 reading proficiency (Bernhardt and Kamil, 1995). However, this is not explored in our work

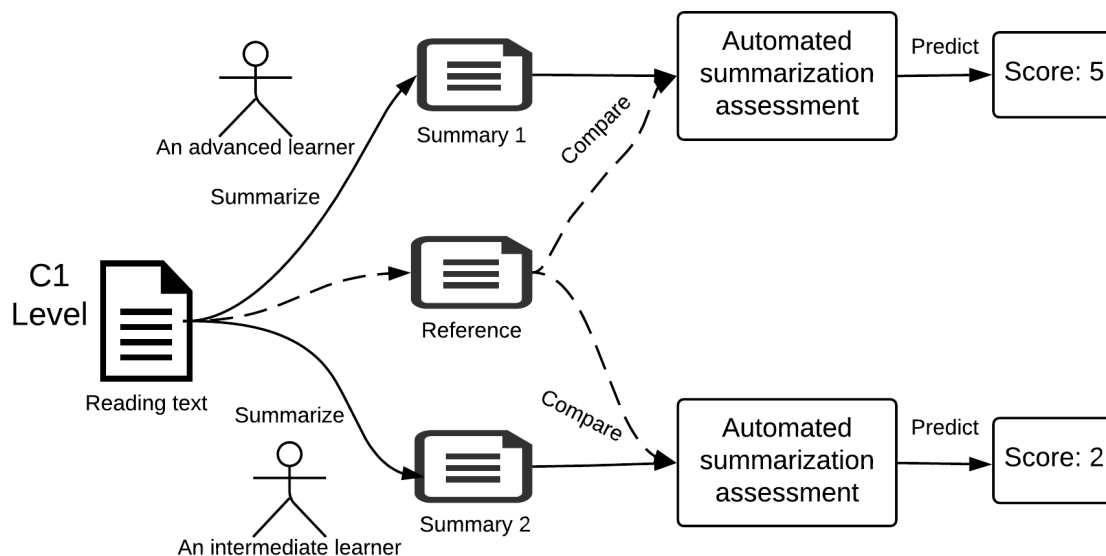


Figure 1.2: An illustration of the framework using automated learner summarisation assessment for evaluating reading comprehension

because it is difficult to measure the individual difference in reading proficiency in practice.

1.2 Assessing learner summarisation for reading comprehension

Readability assessment provides a technique for selecting reading materials for language learners at different proficiency levels. To better assist the reading ability development for the learners, we assess the degree of reading comprehension by the learners. That is to say, when the reading passages that are appropriate to a certain level of readers are selected and prepared, reading tasks can be generated to evaluate readers' comprehension, and thence adapt the selection of content according to the readers' performance in the tasks. In this work, we investigate summarisation as a method for assessing the learner's comprehension and build an automated system to assess the quality of a summary.

Many tasks have been designed to test the reading comprehension of learners, including reading aloud, cloze tests, and multiple choice questions. These tasks are often used to test different aspects of reading ability. Among them, summarisation is considered to be one of the most effective methods to test both the cognitive and the contextual dimensions of reading comprehension (Weir et al., 2013). Hence we describe a summarisation task for assessing the reading comprehension of the learner and discuss the criteria for scoring the summary. We then propose methods to automatically and accurately evaluate the quality of learner summaries.

Figure 1.2 illustrates our framework of using automated summarisation assessment to evaluate learners' reading comprehension. In this framework, learners are asked to write a summary for a reading passage. The system computes an internal representation of the original text as the "gold-standard" reference and compares the learner summary to the reference to assess the quality of the learner summary. Similar to text readability assessment, the task of automated learner summarisation assessment can be addressed with a machine learning approach to assign to each summary a numerical score that reflects the quality of the summary.

Automated assessment of learner summarisation is challenging. It requires the system to be able to establish representations of the original reading passage and the summary, which are then compared to check how well the summary properly summarizes the reading passage. Besides, it is important that the system can reproduce human judgements on assessing the summary. In addition, as automated summarisation assessment is a relatively unexplored field, there is very little publicly available data representing summaries written by L2 learners, let alone summaries of different quality. To deal with the lack of data, we collected summaries from learners at different English proficiency levels with summary qualities annotated by experienced markers.

1.3 Research questions

In this thesis, we aim to address the following research questions:

1. We aim to develop an automated text readability assessment system. *What linguistic features are useful for assessing text readability? Are there any differences in the predicative power of different types of linguistic features for L1 and L2 learners?*
2. One of the major challenges for a data-driven approach to predict text readability for L2 learners is the limited amount of data. We aim to gather a dataset with texts that are tailored for L2 learners' readability. However, at the same time, there is also a larger amount of data available from previous research for L1 readability. *Can we make use of the more abundant L1 data to improve L2 readability assessment?*
3. We want to build a system to automatically assess the learner summary by comparing the content of the summary to that of the original reading passage. Traditional methods in summarization assessment rely on metrics based on exact word match for assessing text similarity. However, we believe that exact word match is not sufficient for assessing text similarity. For example, when paraphrase and abstraction is used in a summary, the content of the summary can still be similar to that of the reading passage without having to use exactly the same words. *What other NLP methods can we exploit to better model text similarity?* We also believe that although our application is for assessing learner summary, the same methods can also be used to benchmark automated summarizers and to assess general text similarity.

4. Although there are datasets on automated summarization, there is very limited data of level-graded summaries. To address this, we would like to collect a dataset of summaries produced by learners at different proficiency levels and ask professional examiners to manually score the summaries. Before we collected the learner data, we also created a simulated dataset with summaries of either “good” or “bad” qualities produced by members of our university to develop our models.

1.4 Contributions

Our work makes three main contributions:

1. We develop a system that produces state-of-the-art estimation of text readability. One of the major challenges in this task is the lack of significantly-sized level-annotated data. We focus on readability for L2 learners of English and present a dataset with level labels for non-native readability analysis. We extract a range of readability measures, investigate their predictive power, and compare the importance of the features for L1 and L2 readability assessment. We found that the model performs the best with a combination of traditional, lexical, syntactic, language model and discourse features.

We demonstrate how to make use of the existing native corpora to produce better estimation of readability when there is not enough data aimed at L2 learners. Specifically, we apply a generalization method to adapt models trained on native data to estimate text readability for learners, and exploit domain adaptation and self-training techniques to improve system performance on the data aimed at L2 learners. To the best of our knowledge, these approaches have not been applied in readability experiments before.

Parts of the work on text readability assessment have been previously published in the following article:

Menglin Xia et al. (2016). “Text readability assessment for second language learners”. In: *Proceedings of the 11th Workshop on Building Educational Applications Using NLP*. Association for Computational Linguistics, pp. 12–22²

2. We present a summarisation task for evaluating non-native reading comprehension and develop an automated summarisation assessment system aimed at evaluating the quality of learner summary. To build and to evaluate our system, we collect summaries from proficient English users and L2 learners with different levels of English proficiency. We explore three novel machine learning based approaches to assess the summary and show that the best system produces estimation close to human performance. More specifically,

²The work in this paper was mainly carried out by myself while I was mentored by Dr Ekaterina Kochmar and Prof Ted Briscoe.

in the first approach, we examine using several NLP techniques for extracting features to evaluate the content similarity between the reading passage and the summary; in the second approach, we propose a similarity matrix to measure sentence-to-sentence similarity between the reading text and the summary and apply a CNN model to assess the summary quality from the similarity matrix treated as an image; in the third approach, we build an end-to-end summarisation assessment system using RNN. Further, we combine the three approaches into a single system using a simple parallel ensemble modelling technique. We show that our automated assessment system outperforms traditional approaches that rely on exact word match on the task. To our knowledge, none of the approaches have been used in previous work for automated summarisation assessment, and hence serve as benchmarks for future research.

3. We build two publicly available datasets: one for L2 text readability assessment and one for learner summarisation assessment.

1.5 Thesis outline

The remainder of the thesis is structured as follows:

Chapter 2 gives an overview of the related work on the two key components of this thesis: text readability assessment and summarisation assessment, and describes how our research relates to and builds upon the existing work.

Chapter 3 describes our approach to assessing text readability. We explore a range of linguistic features to measure text readability, including traditional features, lexico-semantic features, parse tree syntactic features, language modelling features and discourse-based features, and investigate their predictive power for estimating reading difficulty. We train and compare a classification model and a ranking model on the native data to build a general readability assessment system and examine which model is better for the task.

Chapter 4 focuses on text readability assessment for L2 learners. We collect a dataset of texts tailored for L2 readability and compare the importance of the readability features on L1 and L2 data. We apply a generalization method to adapt models trained on larger native corpora to estimate text readability for learners, and explore domain adaptation and self-learning techniques to make use of the native data to improve performance on the limited L2 data.

Chapter 5 presents our work on automated learner summarisation assessment. We describe a summarisation task to assess the reading comprehension of learners and propose several methods to automatically evaluate the quality of the learner summary. The first method makes use of distributed representations of various text units in combination of verbatim overlap metrics to measure the content similarity of the summary against key ideas of the reading text. The second method defines a similarity matrix that captures sentence-to-sentence similarity between the reading text and the summary and learns patterns from the similarity matrix to assess the summary.

The third method is an end-to-end summarisation assessment system based on RNNs with Long Short-Term Memory (LSTM) units. Additionally, we use an ensemble modelling technique to combine the three methods into a single system. We collect data from members of our university and real learners to train and evaluate our systems. We evaluate our model performance against human examiners and compare with the results reported in previous automated approaches for learner summarisation assessment.

Chapter 6 reviews the contributions of our work, presents our conclusions, and discusses directions for future work.

Chapter 2

Background

In this chapter, we review related work on automated text readability assessment and automated summarisation assessment and explain how our research relates to or builds upon existing methods. Section 2.1 reviews the large body of literature on text readability assessment. We summarize existing work on text readability assessment for both native and L2 readers, give an overview of the readability corpora used in previous studies, and show examples of how readability assessment can be applied in educational scenarios. Section 2.2 discusses the summarisation task for assessing reading comprehension of language learners, reviews the limited amount of existing work on automated summarisation assessment for both machine-generated summaries and learner summaries, and present how our research fits in the context of the latest work in the field.

2.1 Text readability assessment

2.1.1 Related work on readability assessment

Research on readability assessment has a history of at least seven decades (Chall, 1958). Earliest works on readability were often based on traditional readability formulae and metrics that represent shallow textual characteristics. Typical traditional reading metrics include number of sentences in a passage, average number of words per sentence, maximum number of words per sentence and average number of characters per word. Traditional measures approximate semantic and syntactic features of text with variables averaged over all words, as a result making the readability estimation easy to compute. However, they inevitably over-simplify the semantic and syntactic aspects of text as they typically ignore the structure and ordering of sentences and words and rely exclusively on superficial counting of text units. For example, the most widely used traditional formula, Flesch-Kincaid grade level score (Kincaid et al., 1975), takes the average number of words per sentence as the indication of syntactic difficulty and the average

number of syllables per word as the estimation of semantic (vocabulary) complexity.

$$\text{Flesch-Kincaid score} = 0.39 * \frac{\text{total \# words}}{\text{total \# sentences}} + 11.8 * \frac{\text{total \# syllables}}{\text{total \# words}} - 15.59 \quad (2.1)$$

Another popular formula is Coleman-Liau Index (Coleman and Liau, 1975), which measures word length by the number of characters instead of syllables.

$$\text{Coleman-Liau Index} = 0.0588 * L - 0.296 * S - 15.8 \quad (2.2)$$

where L is the average number of letters per 100 words and S is the average number of sentences per 100 words.

Some traditional formulae make use of lexical information from additional vocabulary resources. For example, the Revised Dale-Chall Readability Formula (Dale and Chall, 1948; Chall and Dale, 1995) measures word difficulty using the word occurrence in a reference list of 3000 words that are understood by 80% of American fourth-grade students. Words that do not appear in the list are considered difficult. The formula then estimates the level of text with a weighted sum of the percentage of difficult words and the average sentence length. The hard assignments of words into two categorical levels produce rather coarse measure of difficulty.

More than 50 readability formulae have been developed to measure text difficulty more accurately and efficiently over the past seven decades (Crossley et al., 2008). In spite of their computational simplicity, the traditional formulae has been validated in a number of studies that found high correlation between the scores produced by the formula and the observed difficulty by human participants (Chall and Dale, 1995; Crossley et al., 2008; Benjamin, 2012). Also, it is intuitive to estimate text difficulty with vocabulary and sentence complexity, because vocabulary difficulty and sentence structure are known to account for most of the variance in texts across levels of different reading difficulty (Chall, 1958; Collins-Thompson, 2014). As a result, traditional readability measures became popular and are widely used in modern applications such as Microsoft Word.

Despite the fact that these formulae are still widely used, there are several limitations of these methods. Since the formulae only take simplified surface features into consideration, the critical weakness of the traditional readability measures lies in their failure to take into account more complex factors of the text, such as the lexical semantics, the grammatical structure of sentences. Furthermore, such formulae put strict constraint on the text content. They typically assume that the text to be evaluated consists of well-formed sentences (Collins-Thompson, 2014; Davison and Kantor, 1982). As a result, applying these formulae directly to web texts, which are usually relatively noise-prone, is less reliable. Additionally, the validity of these formulae is greatly affected by the length of the text. It has been reported that they would become unreliable for passages with less than 100 words (Collins-Thompson, 2014; Kidwell et al., 2011). This is

because the formulae rely on the superficial statistics of the text. However, in general, the shorter the text is, the less representative the statistics are.

Being aware of such limitations, recent research in readability assessment has shifted to machine learning based methods that enable us to take advantage of a broader set of linguistic features. The machine learning based approaches learn from a training corpus a function that maps the representation of the text to a numerical value that indicates the text difficulty. In this framework, the text is usually represented with a set of features computed from the text that describe different aspects of readability. Depending on the application and machine learning model used in prediction, the output of the learned function can be grade levels of reading difficulty (using a classification model), a continuous-valued score (a regression model), or the ranking of the text compared to others (a ranking model). The most common machine learning frameworks used in the literature include Support Vector Machine (SVM) and logistic regression for classification, and Support Vector Regression (SVR) and linear regression for regression, to name just a few. While most of the work on readability assessment uses a classification or a regression model, there are some studies that use a pairwise ranking model to predict the relative reading difficulty of pairs of texts (Pitler and Nenkova, 2008; Tanaka-Ishii et al., 2010).

The advances in computational linguistics also made it possible to extract various features that can better model the text difficulty. Si and Callan (2001) and Collins-Thompson and Callan (2004a) were among the early works on statistical readability assessment. They applied unigram language models and naïve Bayes classification to estimate the grade level of a given text. Experiments showed that the language modelling approach yields better results in terms of accuracy than the Flesch-Kincaid readability formula. Schwarm and Ostendorf (2005) extended this method to multiple language models. They combined traditional reading metrics with statistical language models as well as some basic parse tree features, and then applied an SVM to classify the level of a given text. Callan and Eskenazi (2007) and Heilman et al. (2008a) expanded the feature set to include certain lexical and grammatical features extracted from parse trees while using a linear regression model to predict the grade level. Furthermore, more recent studies began to develop language features that can model aspects of readability that are associated with higher levels of reading. For example, Pitler and Nenkova (2008) and Feng et al. (2010) were the first to introduce discourse-based features, such as entity coherence, discourse relations, and lexical chain features, into the framework. Their experiments with discourse features demonstrated promising results in predicting the readability level of text for both classification and regression approaches. Coh-Matrix (Graesser et al., 2011) is a popular computational tool that provides discourse measures for text representation. It analyses texts with respect to over 50 types of cohesion relations that are designed to model referential cohesion, cohesion in causal events, and cohesion via cohesive devices between sentences, etc. (Graesser et al., 2011).

Compared to traditional readability formulae, machine learning based approaches provide more accurate and flexible assessment of readability by exploiting a richer feature representation.

François and Miltsakaki (2012) compared the “classic” (= traditional metrics and formulae) and “non-classic” readability features (= NLP-enabled non-traditional features) using linear regression and SVM. They discovered that the NLP-enabled non-traditional features, such as language modelling features and parse tree syntactic features, contribute significantly to the improvement of performance, and SVM outperformed linear regression in accuracy.

In understanding the reasons for improvements of machine learning methods over traditional formulae, Kate et al. (2010) looked at both the effect of the feature choice and the machine learning framework choice on performance. They compared models including Gaussian process regression, decision trees, support vector regression, linear regression, and bagged decision trees. They found that, although the choice of the learning algorithms can affect the results, the gain obtained by changing the framework is smaller than that from changing the features, thereby emphasizing the importance of feature extraction on top of existing hand-designed features.

In sum, machine learning based approaches have demonstrated better predicting power than the traditional formulae in readability assessment. It also allows the investigation of a wider range of features to describe more sophisticated aspects of reading difficulty.

2.1.2 Assessing L1 and L2 readability

In the early stages of research into L2 readability, researchers tried to apply traditional readability formulas (such as the Flesch-Kincaid score) to evaluating L2 reading difficulty directly (Carrell, 1987; Brown, 1997; Greenfield, 1999; Crossley et al., 2008). However, they were mostly dissatisfied by the limited power of the traditional readability formulas in indicating L2 readability. They argued that these scores designed for L1 readability failed to account for L2-reader characteristics. In particular, Brown (1997) argued that “first language readability indices are not very highly correlated to the EFL (English as a Foreign Language) difficulty”. He also argued that readability formulas for L2 readers needed to be more sensitive to the type, function and frequency of words and to word redundancy within the text. Therefore he designed a readability index for estimating L2 reading difficulty that takes the “L2 variables” into account, which includes the average number of syllables per sentence, the frequency of the cloze items in the text as a whole, the percentage of words in the text of more than seven letters, and the percentage of function words. He showed that the Brown’s EFL readability index has a higher degree of correlation with reader performance in a cloze test than the Flesch-Kincaid score. On the other hand, Hamsik (1985) claimed that the traditional readability measures “do measure readability for ESL students and that they can be used to select material appropriate to the reading level of ESL students”. However, Greenfield (1999) argued that Hamsik’s study was neither large enough nor sufficiently fine-grained to settle the question of predictive validity. By comparing the results of the previous studies (Brown, 1997; Greenfield, 1999), Crossley et al. (2008) discussed that although the traditional readability measures can discriminate relative difficulty reasonably well for L2 students, adjustments on these scores that reflect the demands of L2 readers can offer

improvement on assessing L2 readability.

Although reading in the native language shares many important elements with reading in a second language, previous studies (Singhal, 1998; Callan and Eskenazi, 2007) have found that the extent to which the linguistic factors can affect L2 readability may differ from that for native readers. Compared to the L1 learners, L2 learners usually learn from more controlled language input in less frequent time (Beinborn et al., 2014). Due to the difference in the pace and time in language acquisition, L1 readers often already master the basics of the language before they attempt to read texts, whereas L2 learners are regularly affected by unknown words and structures in a text (Beinborn et al., 2014). This may lead to the difference in the effect of linguistic factors on readability for L1 and L2 learners. For example, Callan and Eskenazi (2007) tested the effect of grammatical features extracted from a set of 22 grammatical patterns in parse trees for both L1 and L2 readers. They found that these grammatical features correlate better with L2 readability scores than L1 readability scores. Also, Zeeland and Schmitt (2012) examined the influence of lexical knowledge in L1 and L2 listening comprehension and compared to their influences for reading comprehension. They found that lexical knowledge contributes to both the comprehension in L1 and in L2; however, the level of lexical coverage had different effects on the comprehension for L1 and L2 learners.

Therefore, to build a system that best facilitates language acquisition for L2 learners, we focus on developing methods that assess text readability for L2 learners. In the meantime, we also want to explore how different linguistic features affect L1 and L2 readability differently.

In addition, the reader's first language also influences their perception of L2 reading difficulty (Lee and Schallert, 1997). However, due to reasons of practicality, this is not examined in our work.

2.1.3 Automated readability assessment for L2 learners

Most previous work on automated readability assessment is directed at predicting reading difficulty for native readers. Several efforts in developing automated readability assessment that take L2 learners into consideration have emerged since 2007. For example, Callan and Eskenazi (2007) compared the effect of grammatical features on L1 and L2 reading difficulty. Vajjala and Meurers (2012) combined measures from Second Language Acquisition research with traditional readability features and showed that the use of lexical and syntactic features for measuring language development of L2 learners has a substantial positive impact on readability classification. They observed that lexical features perform better than syntactic features, and that the traditional features have a good predictive power when used with other features. Shen et al. (2013) developed a language-independent approach to automatic text difficulty assessment for L2 learners. They treated the task of reading level assessment as a discriminative problem and applied a regression model using a set of features that are claimed to be language-independent. However, most of these studies (Feng et al., 2010; Vajjala and Meurers, 2012; Shen et al., 2013)

have used textual data annotated with the readability levels for native speakers of English rather than L2 learners who may judge difficulty of a text in a different way.

While the majority of work on automated readability assessment is for English, studies on L2 readability in other languages, including French (François and Fairon, 2012), Portuguese (Branco et al., 2014), and Swedish (Pilán et al., 2016a,b), are also emerging. These studies generally use textbook materials with readability levels assigned by publishers or language instructors.

Overall, study of automatic readability analysis for L2 learners is still in its early stages, mainly due to the lack of available well-labelled data annotated with the readability levels from the perspectives of L2 learners.

2.1.4 Overview of existing readability corpora

Within the data-driven approach, the availability of sufficient well-annotated data that can be used as the gold-standard is very important for training machine learning models for automatic readability analysis. This section gives an overview of the corpora used in previous studies on readability assessment.

Among the limited number of publicly available corpora for readability assessment, the Weekly Reader corpus (Schwarm and Ostendorf, 2005; Feng et al., 2010) is one of the relatively popular datasets for the task. Weekly Reader is an educational newspaper with versions targeted at different grade levels. The articles from the Weekly Reader cover a variety of non-fiction topics, including science, history, and current events. It was first used by Schwarm and Ostendorf (2005) in their statistical language modelling approach where a total number of 2,400 articles from the second, third, fourth and fifth grade editions of the newspaper were gathered. Feng et al. (2010) also reported on a classification task on the Weekly Reader data, but their articles are retrieved from newer editions of the magazine. They collected 1433 full articles ranging from grade 2 to 5.

More recently, Vajjala and Meurers (2012) created a five-level dataset named the WeeBit corpus. The WeeBit corpus contains articles from four grade levels of the Weekly Reader, and from the BBC-Bitesize, which is a website with articles classified into two distinct age groups. The two datasets are then merged and reorganized according to the age range of the targeted readers, resulting in a five-level corpus. The WeeBit corpus consists of 3,125 articles altogether, with 625 articles per reading level.

Parallel corpora between an original corpus and the simplified version of that corpus have been another popular resource for predicting binary reading difficulty. For example, the Simple English Wikipedia dataset (Coster and Kauchak, 2011) was originally created for text simplification research. Coster and Kauchak (2011) obtained 10,588 aligned articles from English Wikipedia¹ and Simple English Wikipedia². English Wikipedia is a web encyclopedia in English, whereas

¹<http://en.wikipedia.org>

²<http://simple.wikipedia.org>

Simple English Wikipedia contains texts of similar content to the English Wikipedia but with simpler vocabulary and grammar adapted for children, English language learners, and native speakers of low literacy. The Simple English Wikipedia dataset was used in a number of studies for readability assessment (Stajner et al., 2012; Vajjala and Meurers, 2014; Wagner et al., 2016) where they treated unsimplified text from English Wikipedia as of “normal” reading difficulty and the simplified version from Simple English Wikipedia as of “simple” reading difficulty. Similar parallel datasets include articles from Encyclopedia Britannica and Britannica Elementary and articles from CNN and abridged CNN (Schwarm and Ostendorf, 2005; Petersen and Ostendorf, 2009).

Many other studies collected texts of various sizes from educational webpages and used the targeted grade level as the gold-standard labels for text readability (Si and Callan, 2001; Collins-Thompson and Callan, 2004a, 2005; Qumsiyeh and Ng, 2011). For example, Newsela is an educational website with articles adapted to different levels of reading difficulty for native students.³ Similarly, Pitler and Nenkova (2008) and Qumsiyeh and Ng (2011) collected texts from different sources and annotated the data themselves. Denning et al. (2015) and Wagner et al. (2016) examined text readability for books and used the meta-information provided by the publishers to annotate the reading difficulty of the books. Additionally, Rodrigo Wilkens and Fairon (2018) used learner essays graded with CEFR⁴ levels as the training corpus for readability assessment.

Table 2.1 lists some of the corpora that have been used in the existing studies on English readability assessment.

Authors	Readability corpora
Si and Callan (2001)	91 science educational webpages that had been written for students at particular American grades levels (K–2, 3–5, and 6–8)
Collins-Thompson and Callan (2004a, 2005)	(a) 550 English documents across 12 American grade levels (b) a set of 228 levelled documents from Reading A-Z.com, spanning grade 1 through grade 6 (c) 17 stories from Diagnostic Reading Scales (DRS) spanning grades 1.4 through 5.5
Heilman et al. (2008a) and Callan and Eskenazi (2007)	(a) 362 Web page articles with reading difficulty level labels (1-12) (b) 200 textbook materials from a series of English as a Second Language reading courses across level 2-5

³<https://newsela.com/>

⁴The CEFR is an international standard for describing language ability. It describes language ability on a six-point scale, from A1 for beginners up to C2 for those who have mastered a language.

Pitler and Nenkova (2008)	30 articles from the Wall Street Journal corpus used in Penn Treebank, rated by three college students on a scale of 1-5
Schwarm and Ostendorf (2005) and Petersen and Ostendorf (2009)	(a) Weekly Reader corpus: 2,400 articles in 4 levels (b) 115 article pairs from Encyclopedia Britannica/Britannica Elementary (c) 111 article pairs from CNN/abridged CNN
Feng et al. (2010)	Weekly Reader corpus: 1,433 articles at 4 levels
Qumsiyeh and Ng (2011)	(a) T Data: a dataset of 5,000 documents extracted from a standardized English Language tests website (14 levels from kindergarten to college) (b) W Data: 5,000 web pages, which were downloaded from different school websites and have been manually assigned a grade level to each page by human experts.
Vajjala and Meurers (2012)	WeeBit corpus: 3,125 articles at 5 levels
Shen et al. (2013)	Data in four languages: Arabic, Dari, English, and Pashto (approximately 1,390 each) across 7 levels, labelled by two independent linguists expertly trained in ILR (Interagency Language Roundtable) level scoring
Vajjala and Meurers (2014)	(a) WeeBit corpus (b) Common Core Standards corpus: 168 English texts belonging to four genres that serve as exemplars for the Common Core Standards reading initiative of the U.S. education system (c) TASA corpus: about 37,000 texts annotated with their reading level in terms of DRP (Degrees of Reading Power) scale assigned by Touchstone Applied Science Associates Inc. (TASA). The score typically ranges from 30–80. (d) 120 Math web pages annotated with the reading level created by Zhao and Kan (2010) (e) Simple English Wikipedia corpus: collected by Zhu et al. (2010): approximately 100,000 article pairs (f) One Stop English corpus: 30 articles from onestopenglish.com that are parallel versions of ten articles at three reading levels.

Denning et al. (2015) * readability of books	BookRL corpus: data extracted from websites containing 18,127 books distributed among the K-12 grade levels with their ranges determined by their publishers.
Wagner et al. (2016)	(a) Wikibooks corpus: 35 virtual books for readers of four different proficiency levels (beginner, intermediary, advanced and professional) (b) Simple English Wikipedia corpus: 2,240 pairing articles (c) Britannica Biographies corpus: 2,385 biographies with versions in three different readability levels (elementary, medium and high)
Rodrigo Wilkens and Fairon (2018)	EF-Cambridge Open Language Database: 532 thousand texts written by English learners and graded with CEFR levels

Table 2.1: An overview of English corpora used in the previous readability research

The way the labelled passages are gathered and their grade level is determined in the training corpus can have an impact on the predictive power of the readability measures. If the articles are from an educational source, it's likely that traditional formulae have been used in the pre-determination of the grades of texts. which are used as the gold standard label in the machine learning framework. Therefore, new machine-learning measures developed using these labels are likely to replicate the same or similar measures that were used to calibrate the articles (Collins-Thompson, 2014).

The majority of existing readability corpora are created and labelled for L1 readers. When using L1 data for analysing L2 readability, the existing studies usually take reading materials for school graders to approximate the levels of reading difficulty for L2 learners. However, while there are certain similarities in lexical and grammatical acquisition for L1 and L2 learners, the difference in the process of acquiring their skills cannot be neglected.

In general, the lack of suitable and accurately labelled data for L2 learners is still the bottleneck of readability studies.

2.1.5 Readability assessment for educational applications

In recent years, several tools integrating readability assessment for educational scenarios have been developed. The REAP (REAders-specific Practice) Readability Tool is a computer-assisted educational learning system designed for vocabulary acquisition developed at the Language Technologies Institute of Carnegie Mellon University (Collins-Thompson and Callan, 2004b; Heilman et al., 2008b). The tool allows the users to search for textual passages retrieved from the web that contain specific vocabulary items. The system obtains documents by query-based

crawling from a commercial web search engine and stored the filtered pages in an off-line database. It then employs a regression model to determine the reading level of each document using a simple bag-of-word representation. A text classifier is used to categorise the candidate documents into 16 topic categories.⁵ The system retrieves documents according to user-specific indications on topical interest from the offline database with the Lemur Toolkit (Ogilvie and Callan, 2001). In several user analysis studies, Heilman et al. (Heilman et al., 2006, 2010) asked students to take cloze tests before and after practising with the system. They showed that practising with the tool and customizing the reading passages to match their topical interests led to consistent gains in student performance in vocabulary acquisition.

Read X is a system developed by Miltsakaki and Troutt (2007, 2008). Read X searches the web and analyses the readability of texts on the web in real time. The system performs the search using the Yahoo Web Service and retrieves the documents from search results. It then extracts the human-readable text from HTML and XML formatted documents using a scraper. Text readability is predicted with a set of traditional readability metrics and mapped to K-12 grade levels⁶. Texts are categorised into 8 super topical categories and 41 subcategories using a text classifier. Moreover, the system incorporates a tool that predicts unknown vocabulary for a specific reader and grade profiles specified by the reader. It checks the vocabulary frequency of the words for the chosen categories and highlights the ones that are less frequent than a pre-determined threshold. The system is designed to assist L1 readers with low reading ability to identify reading material that matches their interest and reading-level.

There are also tools with similar functions, such as Nettekrek,⁷ which is a commercially available search tool for finding digital learning resources for American grade levels. In a related work, Collins-Thompson et al. (2011) attempted to personalize web search result by user's reading level. They computed reading level predictions of snippets (the combined title and summary text that appears for the page in the search engine) and full-text documents extracted from the HTML of the underlying page using language models and traditional metrics. They also estimated users' reading proficiency from their search behaviour, such as the previous queries. Then a re-ranking model was learnt based on the difference between user and resulting reading level profiles. They then extended their work to labelling existing Web pages with metadata that contained readability estimates and showed that these readability metadata can be used in a variety of applications, such as personalized web search and modelling user and website expertise (Collins-Thompson, 2013, 2011).

⁵The topics include: Movies and Theater, Music, Visual Arts, Computers and Technology, Business, Math, Physics and Chemistry, Biology and Environment, Social Sciences, Health and Medicine, Fitness and Nutrition, Religion, Politics, Law and Crime, History, American Sports, and Outdoor Recreation.

⁶K-12 is a short form for the 12 publicly-supported school grades prior to college in the United States and Canada.

⁷<http://www.nettrekker.com/>

2.2 Assessing learner summarisation

2.2.1 Assessing reading

Many tasks have been designed in the domain of language testing to assess reading comprehension at different levels of understanding, ranging from local comprehension to global comprehension. Local comprehension refers to the understanding of lexical items and syntactic and semantic abilities to understand information presented at the sentence level. Global comprehension refers to the understanding of macro-propositions at the text level, including the main ideas of the text, the relations between the ideas, as well as the way in which micro-propositions in sentences elaborate upon the main ideas (Khalifa and Weir, 2009). Assessing reading in the global level examines the overall ability to construct an organized representation of the entire reading text, which is often built upon the ability to understand the local information and the ability to integrate the information to establish a macro-structure of the text. Global comprehension requires a range of reading skills, including lexical knowledge, syntactic ability to parse the sentences, and the ability to inference and establish meaning from discourse.

The most common tasks to assess reading skills include the cloze tests, gap filling tests, multiple choice questions, true/false items, matching, and short answer questions.

The cloze tests and gap filling tests are among the most typical tasks for assessing reading at the local level. Cloze tests are constructed by deleting every n-th word from the text and requires the reader to restore the word that has been deleted. Gap filling tests are similar to the cloze tests; however, the words to be removed in gap filling tests are usually carefully decided by the test constructors. A variant for cloze test and gap filling tests is to provide multiple choices for the learners to select from. What an individual cloze test or gap filling test measures usually depends on the words that have been removed. In general, both tests are particularly useful for testing learners' understanding of propositions in sentences and testing their grammar knowledge. However, since the removed words are usually not constrained by long-range discourse, the tests place more emphasis on local comprehension (Alderson, 2005; Khalifa and Weir, 2009).

Multiple choice questions ask learners to select from a set of choices to respond to particular questions about the reading passage. They have proven to be an appropriate device to test detailed understanding of the text (Khalifa and Weir, 2009). Multiple choice questions are favoured by examinations for their ease in marking. However, there has been concern on their usefulness for assessing global comprehension of the learners, as responding to multiple choice questions is more of a problem solving process relying heavily on verbal reasoning than a fluid process of integrating the macro-propositions to form a representation of the text (Ferne and Rupp, 2007). A true/false test can be seen as a variant of multiple choice questions and is often used in exams to test lower level reading. It asks learners to read a statement about factual information in the text and decide whether they are true or false. True/false tests are normally less reliable than multiple choice questions because it is easier to guess the right answer when there are only two choices.

Matching is another variant of multiple choice questions. In matching tasks, learners are usually asked to match two lists of texts. For example, a typical matching task is to match headings with paragraphs, which has been very useful for assessing the understanding of micro-propositions in the text.

An alternative to multiple choice tests is short answer question, in which learners are asked to write a brief response to a question. Short answer questions can be used to test either the local or global levels of reading comprehension, depending on the questions asked. The main disadvantage of short answer questions is that it requires the construction of a set of questions covering different aspects of reading ability to effectively assess the overall reading comprehension of the learners.

Among the tasks for assessing reading, summary writing has been used to test higher level reading skills and comprehension in many English examinations (Weir et al., 2013). In summary writing tasks, students are typically asked to read a text and then summarize the main ideas of either the whole text or a part of it. The summarisation task is often used to test the student's ability to understand the text and to separate relevant from irrelevant information. It is believed that summarisation is an effective approach to assess both cognitive and contextual dimensions of reading. Summarisation tasks help to test reader's ability to form a coherent mental representation of the text (Kintsch, 1998) which relies on understanding of the rhetorical text structures, grammatical structures and word meanings, ability to reason about conceptual content and social content, and ability to draw conclusions and infer meaning. Bensoussan and Kreindler (1990) argued that summary writing is measuring comprehension at the "whole-text, super macro level". Nuttall (2005) stated that summarisation is a better reading task compared to multiple choice questions and cloze tests as it demands a fuller understanding of the text. This is because multiple choice questions and cloze tests often require the reader to pay close attention to individual words or sentences that are related to the task, whereas summarisation requires a more global understanding of the text. Studies in cognitive psychology claimed that comprehension naturally involves summarisation (Kintsch and Van Dijk, 1978; Van Dijk et al., 1983). In order to write a good summary, students need to establish the main ideas of the text, extract them and rewrite them in a coherent manner in their own words. Yu (2008) provided empirical support that L2 learners with better comprehension (evaluated from post-task questionnaires and interviews) produced better summaries.

In addition, empirical studies have attested that summarisation is an effective strategy to help learners improve reading skills, as it both measures and encourages reading comprehension. According to (Wade-Stein and Kintsch, 2004), students demonstrate more detailed and thoughtful contributions in classroom discussions on topics that they have summarized instead of merely read. Not surprisingly, there has been a considerable amount of empirical evidence that supports this observation of better comprehension and better retention by students on materials they have summarized (Bean and Steenwyk, 1984; Rinehart et al., 1986; Hill, 1991; Friend, 2001; Thiede

and Anderson, 2003; Duke and Pearson, 2009). As a consequence, summary writing has also been used extensively in language training (Khalifa and Weir, 2009).

Nonetheless, concerns have also been expressed that students' writing ability may interfere with their performance on summary tasks (Weir, 2005; Urquhart and Weir, 2014). In addition to comprehension, knowledge transformation and content organization skills are also required in writing a good summary. Also, integrating reading with writing presents a challenge for examiners in making decisions about to what extent verbatim copying from the text is permissible and if the learner is capable of producing a summary without borrowing from the text (Khalifa and Weir, 2009). For these reasons, researchers argued that objective and reliable marking of summaries is necessary in judging the language skills from summarization tasks (Weir, 1993; Weir et al., 2013). Therefore, we focus on developing an automated learner summarisation assessment system that serves as an objective and scalable device for assessing reading.

In general, summarisation is a task that effectively tests the higher level reading skills of constructing a text-level representation (Weir et al., 2013) and has multiple unique advantages compared with other known tasks. It is also a useful exercise that can help learners to develop better reading comprehension.

However, unlike multiple choice questions which can be scored easily and efficiently with a machine, scoring summaries in traditional English examinations is a time consuming and costly process that requires human examiners to manually evaluate the quality of the summary. Automating the process of evaluating learner summary can provide a cost-efficient and effective solution to assessing reading with summarisation, and lead to fully automated educational applications that utilise summarisation to enhance reading comprehension.

2.2.2 Automated evaluation of machine-generated summaries

Most of the previous studies on summary assessment are for evaluating automated summarisation systems. A number of methods to evaluate machine generated summaries have been used. Depending on whether human assessors are involved in the evaluation process, there are two types of evaluation approaches: manual evaluation and automatic evaluation. The manual evaluation approach usually requires human assessors to rate summaries on a scale and the scale is usually task-specific. Because of the cost and inefficiency of manual evaluation, researchers began to develop automatic methods to evaluate summaries. The automatic evaluation methods usually compare the machine generated summary against gold-standard summaries produced by a human using a set of automatic measures.

Among the automatic evaluation methods, ROUGE (Lin, 2004) is the most commonly used metric to evaluate the quality of a summary (Nenkova and McKeown, 2011). Inspired by the success of BLEU (Papineni et al., 2002) in machine translation, similar measures of N-gram overlap statistics are used in summary evaluation. It is assumed that the closer a candidate summary is to the gold-standard summary, the better it is. ROUGE measures the overlap between

a candidate summary and a set of reference summaries. There are several variants of the ROUGE metric, including ROUGE-N, ROUGE-L, ROUGE-W, ROUGE-S, and ROUGE-SU. Among them, ROUGE-N is defined as the N-gram recall between a candidate summary and a set of reference summaries. The definition of ROUGE-N can be formally written as:

$$ROUGE-N = \frac{\sum_{R \in S} \sum_{n\text{-grams} \in R} count_{match}(n\text{-gram})}{\sum_{R \in S} \sum_{n\text{-grams} \in R} count(n\text{-gram})} \quad (2.3)$$

where S is a set of reference summaries, R is one reference summary, and $count_{match}(n\text{-gram})$ is the maximum number of N-grams co-occurring in the candidate summary and the reference summary.

ROUGE-L extends the measurement of N-gram co-occurrences to the count of the length of the longest common subsequence (LCS) between the candidate and the reference summary. A common subsequence of two word sequences $X = [x_1, x_2, \dots, x_m]$ and $Y = [y_1, y_2, \dots, y_n]$ is a sequence of words that appear in both X and Y with increasing indices in both sequences. The intuition of counting the LCS is that the longer the LCS of two summaries is, the more similar the two summaries are. The LCS can be measured at both the sentence level and the summary text level. However, the standard LCS does not take the spatial relations within the sequence into account. This is because non-consecutive common subsequence also count as LCS. However, intuitively, consecutive common sequence should be preferred in a good summary. ROUGE-W improves upon ROUGE-L by assigning larger weights to consecutive matches in the LCS.

ROUGE-S measures the skip-bigram overlap between the candidate summary and the reference summary. Skip-bigrams are pairs of words in a word sequence arranged in the same order as they appear in the sequence, allowing gaps between the words. For example, for the word sequence $X = [x_1, x_2, x_3, x_4]$, the skip-bigrams in X are (x_1, x_2) , (x_1, x_3) , (x_1, x_4) , (x_2, x_3) , (x_2, x_4) , (x_3, x_4) . ROUGE-S calculates the co-occurrence of the skip-bigrams in the candidate summary and the reference summary. ROUGE-SU is almost identical to ROUGE-S. However, it adds a begin-of-sentence marker at the beginning of each summary to handle the cases where there is no skip-bigram overlap in the candidate summary and reference summary thus allowing for unigram co-occurrences.

ROUGE also comes with different settings, including word stemming, stop word removal and the length of N-gram. Different variants of ROUGE work best for different summarisation tasks, and overall ROUGE scores have been proven to correlate well with human judgements on Document Understanding Conference (DUC) 2001, 2002, and 2003 evaluation data (Lin, 2004). For example, the correlation of ROUGE-1 and human judgements ranges between 0.76 and 0.99 on the DUC 2001, 2002 and 2003 data. More details on correlation scores between different types of ROUGE measures and human judgements can be found in Lin (2004). Since then, it has been the dominant evaluation method in automatic summarisation.

ROUGE has been preferred for summarisation to BLEU because it is a recall-oriented

measure, whereas BLEU emphasizes precision. BLEU was first proposed by Papineni et al. (2002) and is the most popular evaluation method for machine translation. Lin and Hovy (2003) used BLEU to evaluate a DUC 2001 summarisation model, and found that the Spearman rank order correlation coefficient between the BLEU score and the human assessment is 0.66 on the single document summarisation task. It shows that BLEU is also a useful score for summarisation evaluation.

BLEU uses N-gram precision (p_n) to compare the candidate summary against a set of reference summaries:

$$p_n = \frac{\sum_{n\text{-gram} \in C} \text{count}_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in C} \text{count}(n\text{-gram})} \quad (2.4)$$

where C is a candidate summary, $\text{count}_{clip}(n\text{-gram})$ is the maximum number of N-grams co-occurring in a candidate summary and a reference summary clipped by the largest count observed in any single reference for that word, and $\text{count}(n\text{-gram})$ is the number of N-grams in the candidate summary. To prevent very short candidates from receiving high scores, a brevity penalty BP is added to BLEU formula:

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-\frac{r}{c})} & \text{if } c \leq r \end{cases} \quad (2.5)$$

where c is the length of the candidate summary and r is the length of the reference summary.

BLEU is then defined as:

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^N w_n \log p_n\right) \quad (2.6)$$

where N is the highest order N-gram in consideration (usually $N = 4$), w_n is a weighting factor usually set at $w_n = \frac{1}{N}$.

Both ROUGE and BLEU focus on verbatim overlap between the candidate summary and the gold-standard reference to evaluate the quality of the candidate summary. However, sentences can express similar meanings without verbatim borrowing. The pyramid method (Nenkova and Passonneau, 2004; Nenkova et al., 2007) is a manual evaluation approach for automated summarisation that focuses on the content similarity. The pyramid method relies on multiple reference summaries to evaluate a candidate summary. It is assumed that although the wording of the reference summaries can vary, the reference summaries are likely to have content in common and have content that is unique to each summary. A pyramid is created from the set of references by identifying Summary Content Units (SCUs) in the reference summaries that express approximately the same meaning. SCUs can vary in length, because the same information can be conveyed in different words. The SCUs are then weighted by the frequency with which they appear in the set of reference summaries to form a ‘‘pyramid’’. The resulting

pyramid tends to have very few SCUs with high weights and a large number of SCUs with lower weights. When a candidate summary is evaluated against the pyramid, it is matched against the SCUs and scored by the sum of weights of the matched SCUs. The pyramid requires manual processing in two steps: the creation of a pyramid and the annotation of the candidate summaries against the pyramid. Harnly et al. (2005) automated the latter step of scoring the summary against the pyramid. In their experiment, they enumerated all the continuous phrases (candidate contributors) in the candidate summary and matched against the pyramid to find the most similar SCUs. They used cosine similarity, cosine similarity with TF-IDF weighting, unigram overlap, bi-gram overlap, and word-wise edit distance to measure the similarity between the candidate contributors and the SCUs. Then they selected a set of disjoint candidate contributors that have the maximum overall similarity with the pyramid and calculated the pyramid score for the candidate summary using the chosen contributors. Harnly et al. (2005) showed that automated pyramid scoring yields higher correlation to human scores than ROUGE in automated evaluation of summary. Yet their system still requires manual processing to extract the SCUs as well as multiple gold standard summaries and therefore is not fully automated.

While most methods on automated evaluation of machine-generated summaries rely on using one or multiple human-generated references to assess the candidate summary, Louis and Nenkova (2013) proposed a method to assess machine summary content without a gold standard. They used several metrics including Kullback Leibler (KL) divergence (Kullback and Leibler, 1951), Jensen Shannon (JS) divergence (Fuglede and Topsoe, 2004) and cosine similarity based on TF*IDF vector representations to evaluate the similarity of the machine summary and the input text. They calculated the correlation between their scores and the score from the manual pyramid method on Text Analysis Conference (TAC) 2008 data⁸ and found that the JS divergence score correlates well (0.88) with the pyramid score. The similarity metrics they used went beyond simple exact word match and are based on the similarity of word distributions in the candidate summary and the original input text. However, similar to the shortcomings of exact word match, similar word distribution also does not guarantee content similarity. In addition, they proposed a method that requires fewer gold-standard summaries. In this method, they first evaluated the candidate summaries using a small number of gold-standard reference summaries available, and then they added the top ranking candidate summaries to the set of reference summaries and re-evaluated the rest of the summaries. They found that adding the “pseudo gold-standard” summaries improved the correlation of ROUGE scores of the candidate summaries with human judgements.

⁸<https://tac.nist.gov//2008/summarization/op.summ.08.guidelines.html>

2.2.3 Related work on automated evaluation of learner summaries

However, in contrast to the work on automated evaluation of automated summarisation systems, our goal is to assess human-generated summaries rather than machine-generated summaries. The work that is most similar to ours is the automatic scoring of a summarisation task by Madnani et al. (2013). They designed a task to measure the reading comprehension skills of native learners from grades 6 to 9 in the K-12 levels. In their task, students were asked to write a four-sentence summary to a reading passage with three paragraphs, where the first sentence is a summary on the whole passage and the following three sentences are on each paragraph. They collected 2,695 summaries written by students from the 6th, 7th and 9th grades on two passages. All summaries were scored by an experienced human rater on a 5-point scale. To build an automated system to score the summaries, they randomly selected a student summary with the highest score as the reference to calculate BLEU and ROUGE scores, and derived 8 NLP features in total to train a logistic regression classifier to predict the summary score.

Specifically, the features they used to evaluate the summaries include:

- BLEU
- ROUGE
- percentage of N-grams (with $N \geq 3$) that are copied from the passage in the summary
- percentage of N-grams (with $N \geq 3$) that are copied from the passage in the summary, compared to the total number of N-grams in the passage
- length of the longest word sequence in the summary copied from the passage
- number of sentences in the passage that the first sentence of the summary borrows N-grams (with $N \geq 2$) from
- number of sentences in the summary
- token counts of commonly used discourse connector words in the summary

They trained a separate classifier for each of the two passages, and obtained 65% and 52% accuracy on predicting the summary scores on the two passages. Their results outperformed a most-frequent-score baseline that yields 51% and 32% in accuracy. In their task, students were not explicitly told to refrain from verbatim copying, and there is no limit to the length of the summary by words.

In addition to the overall performance, they also examined how the features contribute to the prediction of summaries scores. They found that all the features correlate positively with the summary score. Among them, BLEU and ROUGE were the most useful features.

There are several limitations to their work. First, we want to build a generic system that can evaluate the quality of a summary for any reading passage without having to train a separate

model for each passage. However, their system is trained for particular passages. Second, they used a student summary with the highest score as the reference to evaluate the candidate summary, whereas we want to build a fully-automated system that doesn't require a manually prepared reference to assess the candidate summary. Third, most of the features used in their work capture information about verbatim overlap between the candidate and the reference summary. Although verbatim metrics prove to be effective in various tasks, they cannot capture the content similarity when paraphrasing and higher level of abstraction are used in the summary. Therefore, we will explore other features to capture the semantic similarity between the candidate summary and the reference.

For example, Summary Street (Wade-Stein and Kintsch, 2004) is an educational software designed for children to develop summarisation skills. It asks students to write a summary to a reading passage and scores the summary using Latent Semantic Analysis (LSA). It uses LSA to construct semantic representations of the summary text and a set of manually determined key sentences in the reading text. A score is assigned to a summary by calculating the cosine distance between the representation of the summary and the representation of the key sentences in the semantic space. The system then encourages the students to edit the summary until they get a higher score. Unlike ROUGE and BLEU, the system based on LSA evaluates similarity beyond verbatim overlap and captures topical and textual similarity without relying on the presence or absence of special key words or terms. Their system uses the cosine similarity based on LSA as the sole indicator of content similarity. However, in LSA, the text is represented as a bag of words and therefore it fails to capture the dependency of words in the text. Also, LSA may capture topical similarity instead of content similarity. In addition, LSA is usually outperformed when compared to newer approaches for building distributional representations (Mikolov et al., 2013b).

In sum, there has not been much research into automated evaluation of learner summaries for reading comprehension, and the related studies often use verbatim similarity to measure the quality of a summary. Verbatim measures are simple to compute and prove to be useful in identifying similarity between a candidate summary and its reference. However, measuring verbatim similarity is not enough for evaluating the content of a summary. For example, verbatim measures will fail to recognize a good summary when paraphrase and abstraction is present, especially in the case of abstractive summarisation by the learner. The early work on the LSA-based summary evaluation system demonstrates summary evaluation beyond verbatim borrowing. Overall, to develop an efficient and accurate evaluation method to assess learner summaries is a challenging task and it is yet to be explored.

Chapter 3

Assessing text readability

3.1 Introduction

The reading difficulty of a text is affected by many variables, such as the lexical variation, syntactical complexity, and structure and organization of the text, etc. In this work, we approach text readability as a supervised machine learning problem and learn a function from the annotated data that maps features extracted from a reading text into a numerical value indicating the reading difficulty.

In this chapter, we present our approach to assessing text readability. We introduce the data aimed at native English readers we use to develop the text readability assessment system and describe the range of linguistic features explored. We then introduce the machine learning framework applied to the native data that constitutes a general readability assessment system. The set of features used in our experiments is an extension of those used in previous work (Vajjala and Meurers, 2012, 2014; Feng et al., 2009; Pitler and Nenkova, 2008), and their predictive power for reading difficulty assessment is investigated in our experiments. Specifically, we have extended the feature set with the English Vocabulary Profile (EVP) based lexical features, grammatical relation (GR) based syntactic complexity measures extracted with the Robust Accurate Statistical Parsing (RASP) system (Briscoe et al., 2006), and the combination of language modelling features that have not been applied to readability assessment before.

3.2 Native data: the WeeBit corpus

As a data-driven approach, developing a machine learning based readability assessment system requires a large amount of training data. Among the existing publicly available corpora, the WeeBit corpus created by Vajjala and Meurers (2012) is one of the largest datasets for readability analysis. The WeeBit corpus contains a large number of texts aimed at native learners of different grade levels, and the texts in the WeeBit corpus cover a wide range of topics. Therefore, we use the WeeBit corpus to train our models and to examine how much different types of readability

	Level1	Level2	Level3	Level4	Level5
age group	7-8	8-9	9-10	10-14	14-16
original corpus	629	801	814	1969	3500
modified corpus	529	767	801	1288	845

Table 3.1: Number of documents in the original and modified WeeBit corpus

measures contribute to the estimation of text readability.

The WeeBit corpus is composed of articles targeted at native readers from two sources: the Weekly Reader magazine and the BBC-Bitesize website. Within the dataset, the Weekly Reader data consists of texts covering age-appropriate non-fictional content for four grade levels, corresponding to children of ages between 7-8, 8-9, 9-10 and 10-12 years old. The BBC-Bitesize website data is targeted at two grade levels, for ages between 11-14 and 14-16. The two datasets are merged to form the WeeBit corpus, with the targeted ages used to assign readability levels.

A copy of the original WeeBit corpus was obtained from the authors (Vajjala and Meurers, 2012). The texts are webpage documents stored in raw HTML format. We have identified that some texts contain broken sentences or extraneous content from the webpages, such as copyright declaration and links, that correlate with the target labels in a way which is likely to artificially boost performance on the task and would not generalize well to other datasets. To avoid that, we re-extracted texts from the raw HTML and discarded text documents that do not contain proper reading passages. We manually examined a sample set and used Perl script to filter out the improper HTMLs. Examples of texts from the original WeeBit corpus and texts from modified WeeBit corpus are shown in Appendix A and Appendix B. Table 3.1 shows the distribution of texts in the modified dataset.

3.3 Readability measures

This section describes the readability measures used for assessing text reading difficulty. A range of linguistic features are explored and their predictive power for readability assessment are investigated. The set of features includes traditional readability features, lexico-semantic features, parse tree syntactic features, language modelling based features, and discourse features.

The set of all features used in our experiments can be summarized as follows:

- *Traditional readability features* (see 3.3.1):
 - number of sentences in the text
 - average number of words per sentence
 - maximum number of words per sentence
 - average number of characters per word

- average number of syllables per word
- Flesch-Kincaid score
- Coleman-Liau Index
- *Lexico-semantic features* (see 3.3.2):
 - type-token ratio (TTR), Root TTR, Corrected TTR
 - lexical variation of nouns, adjectives, verbs, adverbs and prepositions, Root lexical variation, Corrected lexical variation
 - lexical density of nouns, adjectives, verbs, adverbs and prepositions, content words combined and function words
 - lexical complexity based on the Academic Word List
 - lexical complexity based on the English Vocabulary Profile
- *Parse tree syntactic features* (see 3.3.3):
 - average parse tree depth
 - average number of NPs, VPs, PPs and APs per sentence
 - average number of clauses per sentence
 - grammatical relation-based measures (GR measures)
 - other parsing complexity measures from the RASP system output
- *Language modelling features* (see 3.3.4):
 - word token N-gram models with $N \in [1, 5]$ trained on the British National Corpus (BNC)
 - Part of speech N-gram models with $N \in [1, 5]$ trained on five grade levels of texts in WeeBit
- *Discourse-based features* (see 3.3.5):
 - entity density features
 - lexical chain features
 - entity grid features

3.3.1 Traditional features

The traditional features are easy-to-compute representations of superficial characteristics of text. Although traditional readability metrics can only capture shallow aspects of the text, they have proven to be highly informative in most readability classification tasks. Their computational simplicity is also one of the major reasons why they remain widely used in readability assessment. The metrics that are considered in our experiments include: the number of sentences per text, average and maximum number of words per sentence, average number of characters per word, and average number of syllables per word. Two popular readability formulae are also included: the Flesch-Kincaid score (see formula 2.1 in Section 2.1.1) and the Coleman-Liau readability formula (see formula 2.2 in Section 2.1.1).

3.3.2 Lexico-semantic features

Vocabulary knowledge is one of the most important aspects of reading comprehension (Collins-Thompson, 2014). Lexico-semantic features provide information about the difficulty or familiarity of vocabulary in the text.

A widely used lexical measure is the *type-token ratio* (TTR), which is the ratio of the number of unique word tokens (referred to as types) to the total number of word tokens in a text. However, the conventional TTR is influenced by the length of the text: a shorter text normally has a higher TTR than a longer one. *Root TTR* (see formula 3.1) and *Corrected TTR* (see formula 3.2), which take the square root of the text length instead of the direct word count as denominator, can produce a more unbiased representation and are included in the experiment (Durán et al., 2004).

$$\text{Root TTR} = \frac{\text{total number of types}}{\sqrt{\text{total number of words}}} \quad (3.1)$$

$$\text{Corrected TTR} = \frac{\text{total number of types}}{\sqrt{2 * \text{total number of words}}} \quad (3.2)$$

Part-of-speech (POS) based lexical variation and lexical density measures (Lu, 2011) are also examined. *Lexical variation* is defined as the type-token ratio of lexical items including nouns, adjectives, verbs, adverbs and prepositions. For example, verb variation is the ratio of unique verbs to the total number of verbs. Root lexical variation (based on Root TTR) and Corrected lexical variation (based on Corrected TTR) are also computed for each of the lexical items. *Lexical density* is defined as the proportion of the five classes of lexical items in all word tokens. The percentage of content words (nouns, verbs, adjectives and adverbs) and function words (all the remaining POS types) out of all word tokens are two other indicators of lexical density.

Vajjala and Meurers (2012, 2014) reported in their readability classification experiment that the proportion of words in the text that are found in the Academic Word List is one of the

most predictive measures among all the lexical features they considered. The Academic Word List (Coxhead, 2000) is comprised of words that frequently occur across all topic ranges in an academic text corpus that covers such subjects as art, commerce, law and science. The list excludes the most frequent 2000 words of English, proper nouns and words that have a narrow range in terms of subject areas, i.e. technical terms and argot. It was intended to be used for preparing learners for academic readings and discussions. The list contains 570 word families and approximately 3000 words altogether. The percentage of academic vocabulary in the text can be viewed as a measure of lexical complexity of the text.

A similar but more refined approach to estimate lexical complexity is based on the use of the *English Vocabulary Profile* (EVP).¹ The EVP is an online vocabulary resource that contains information about which words and phrases are acquired by learners at each CEFR level. It is collected from the Cambridge Learner Corpus (CLC), a collection of examination scripts written by learners from all over the world (Capel, 2012). It provides a more fine-grained lexical complexity measure that captures the relative difficulty of each word by assigning the word difficulty to one of the six CEFR levels. Additionally, the EVP indicates the word difficulty for L2 learners rather than native speakers, which makes it more informative in non-native readability analysis.

The total number of entries in the current version of EVP is 15,665, with 11,059 word senses and 4,606 phrases. For a given word, the lowest level across all the senses is chosen as its lexical difficulty level. If a word has several POS tags assigned, these are added as separate entries with the appropriate CEFR levels. For example, the word *access* comes with the following three word sense entries in EVP:

entry	POS	sense	lexical difficulty level
<i>access</i>	noun	RIGHT/OPPORTUNITY	B1
<i>access</i>	noun	METHOD	B1
<i>access</i>	verb	-	B2

Therefore the chosen lexical difficulty for the word *access* is as follow:

entry	POS	lexical difficulty level
<i>access</i>	noun	B1
<i>access</i>	verb	B2

Taking into account different word senses and POS, the total number of unique entries in EVP is reduced to 7,361. In the experiments, the proportion of words at each CEFR level is calculated and incorporated in the feature set.

¹<http://www.englishprofile.org/>

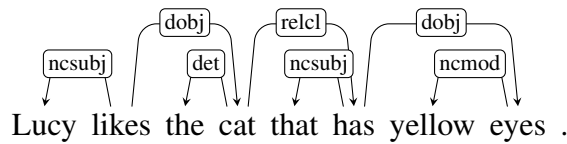


Figure 3.1: An example of GRs in a sentence

3.3.3 Parse tree syntactic features

Syntactic complexity is another key factor that influences text readability. Study showed that more complex syntactic structures usually take more time to process and is generally more difficult to understand (Collins-Thompson, 2014). Syntactic complexity measures are typically derived from the outputs of a grammatical parser.

In our experiments, a number of *syntactic measures* based on the RASP parser output (Briscoe et al., 2006) are used to describe the grammatical complexity of text, including the average parse tree depth, the average number of noun, verb, adjective, adverb, prepositional phrases per sentence (decided by their POS tags), and the average number of clauses (from the RASP output) per sentence.

Grammatical relations (GR) between words may also affect the judgement of syntactic difficulty. A GR describes syntactic dependencies between constituents in a sentence. Figure 3.1 demonstrates GRs identified by RASP in a sentence. For example, ‘*nsubj*’ represents binary relations between non-clausal subjects and their verbal heads, as in ‘*Lucy*’ and ‘*likes*’. Depending on the direction of the relation, i.e. the position of the head compared to the dependent, there are positive dependencies and negative dependencies. For example, ‘*Lucy*’ and ‘*likes*’ have a positive dependency as the position of the head is after the dependent, whereas ‘*like*’ and ‘*cat*’ have a negative dependency as the head precedes the dependent.

Yannakoudakis (2013) applied 24 GR-based complexity measures in essay scoring and found that they perform well on the task. These complexity measures capture the grammatical sophistication of the text through the representation of the distance between the sentence constituents. For instance, these measures calculate the longest/average distance in the GR sets generated by the parser and the average/maximum number of GRs per sentence. In total, a set of 24 GR-based measures are generated by RASP for each sentence. We take the average of these measures across the text to incorporate the GR-related aspect of its syntactic difficulty.

In particular, we extract the following GR-based measures, which are grouped for positive and negative dependencies (Yannakoudakis, 2013):

1. GR-LONGEST-TOP-P/N: longest distance in word tokens between a head and dependent over the top ranked derivation for positive and negative dependencies (P/N) separately.
2. GR-TOTAL-TOP-P/N: sum of the distances between a head and dependent over the top ranked derivation for P/N dependencies separately.

3. GR-MEAN-TOP-P/N: the means for P/N dependencies calculated by dividing GRTOTAL-TOP-P/N by the number of GRs in the set for the top parse only.
4. GR-LONGEST-NBEST-P/N: longest distance for P/N over the top 100 parses.
5. GR-TOTAL-NBEST-P/N: sum of the distances for all GR sets over the top 100 parses for P/N separately.
6. GR-MEAN-NBEST-P/N: the means for P/N dependencies calculated by dividing GR-TOTAL-NBEST-P/N by the number of GRs in the top 100 parses.
7. NBEST-MED-GR-TOTAL-P/N: median for GR-TOTAL-NBEST-P/N (calculated over the top 100 parses).
8. NBEST-STD-GR-TOTAL-P/N: standard deviation for GR-TOTAL-NBEST-P/N.
9. NBEST-AVG-GR-TOTAL-P/N: average for GR-TOTAL-NBEST-P/N.
10. NBEST-MED-GR-LONGEST-P/N: median for GR-LONGEST-NBEST-P/N.
11. NBEST-STD-GR-LONGEST-P/N: standard deviation for GR-LONGEST-NBEST-P/N.
12. NBEST-AVG-GR-LONGEST-P/N: average for GR-LONGEST-NBEST-P/N.

Other types of complexity measures that are derived from the parser output include: *cost metric*, which is the total number of parsing actions performed for generating the parse tree; *ambiguity of the parse*, and so on. A total number of 114 non-GR based complexity measures that describe different costs in parsing are extracted. These complexity measures are averaged across the text and used to model finer details of the syntactic difficulty of the text.

3.3.4 Language modelling features

Statistical language modelling provides information about distribution of word usage in the text and is in fact another way to describe the lexical dimension of readability. Different ways of using language modelling for readability prediction are explored.

A language model (LM) is a function that assigns probabilities to sequences of words (Jurafsky and Martin, 2008). Among language modelling, the N-gram model is a simple model for estimating the probability, where the term N-gram refers to a sequence of N words. To estimate the probability $P(w_1, w_2, \dots, w_m)$ of observing a sequence w_1, w_2, \dots, w_m , it can be decomposed using the chain rule of probability:

$$P(w_1, w_2, \dots, w_m) = \prod_{i=1}^m P(w_i | w_1, \dots, w_{i-1}) \quad (3.3)$$

In the N-gram model, it is assumed that the probability of the word w_i given the context of the preceding $i - 1$ words can be approximated with the probability of the word in a shorter context of the most recent $N - 1$ words.

Therefore the probability $P(w_1, w_2, \dots, w_m)$ is approximated as:

$$P(w_1, w_2, \dots, w_m) \approx \prod_{i=1}^m P(w_i | w_{i-N+1}, \dots, w_{i-1}) \quad (3.4)$$

The probability of a word w_i given the context of previous $N - 1$ words is approximated by counting the number of times the sequence occurs in a corpus and normalized by the count of all N-grams sharing the same context of the previous $N - 1$ words:

$$P(w_i | w_{i-N+1}^{i-1}) = \frac{\text{count}(w_{i-N+1}^i)}{\sum_i \text{count}(w_{i-N+1}^i)} \quad (3.5)$$

where w_1^n denotes $w_1 \dots w_n$ and $\text{count}(w_1^n)$ is the number of occurrences of the word sequence in the corpus.

When scoring the probability of the text with the LMs, frequent word sequences in the training corpus will get high probability and infrequent word sequences will get low probability. Therefore LMs is useful for capturing patterns of word usage across the readability levels.

Larger N-grams capture information of longer context, however, it also requires sufficient training data. For higher order N-grams, the model is usually more susceptible to the problem of data sparsity. As the corpus is limited, some perfectly acceptable N-grams are bound to be missing from the training data. The N-gram model assigns zero probability to these unseen N-grams, and this will lead to zero probability for the entire word sequence.

To keep an LM from assigning zero probability to these unseen sequences, *smoothing* and *backoff* techniques are often used. *Smoothing* is used to assign non-zero probability to unseen N-grams by adjusting the probability mass for these unseen patterns. Typical smoothing methods include add-one smoothing (Jurafsky and Martin, 2008), Good-Turing smoothing (Good, 1953), and Kneser-Ney smoothing (Kneser and Ney, 1995). *Backoff* (BO) is a type of smoothing method that approximates the probability of an unseen N-gram with the probabilities of lower order N-grams, which usually have richer statistics. In order for a backoff model to give a correct probability distribution, a discounting function is usually used to save some probability mass to distribute to the lower order N-grams:

$$P_{BO}(w_i | w_{i-N+1}^{i-1}) = \begin{cases} P^*(w_i | w_{i-N+1}^{i-1}) & \text{if } \text{count}(w_{i-N+1}^i) > 0 \\ \alpha(w_{i-N+1}^{i-1}) P_{BO}(w_i | w_{i-N+2}^{i-1}) & \text{otherwise} \end{cases} \quad (3.6)$$

where α is a discounting factor and P^* is the discounted probability.

Kneser-Ney smoothing is one of the most commonly used and best performing smoothing

methods (Jurafsky and Martin, 2008). Kneser-Ney smoothing is a method derived from *absolute discounting* (Ney et al., 1994) with the backoff model. *Absolute discounting* (AS) reduces the probability mass of observed N-grams by subtracting a fixed value d from each count.

$$P_{AS}(w_i|w_{i-N+1}^{i-1}) = \frac{\text{count}(w_{i-N+1}^i) - d}{\sum_i \text{count}(w_{i-N+1}^i)} \quad (3.7)$$

Kneser-Ney smoothing improves absolute discounting with a more sophisticated approach to estimate the probabilities of lower order N-grams. When backing-off to lower order N-gram probabilities, the standard backoff model (see equation 3.6) does not distinguish words that are frequent but only occur in a small number of contexts from words that are less frequent but occur in a wider variety of contexts. Kneser-Ney smoothing modifies the count of lower order N-grams by taking account of the number of different contexts the pattern has appeared in. Instead of counting the number of occurrences of the N-gram, it counts the number of times the N-gram appears in a novel context.

The number of times an N-gram appears in a novel context is the cardinality of the set of the context where the last word in the N-gram appears in:

$$KNcount(w_{i-N+1}^i) = |w_{i-N+1}^{i-1} : \text{count}(w_{i-N+1}^i) > 0| \quad (3.8)$$

Therefore in Kneser-Ney smoothing, the probability of the lower order N-gram is computed as:

$$P_{KN}(w_i|w_{i-N+2}^{i-1}) = \frac{KNcount(w_{i-N+2}^i)}{\sum_i KNcount(w_{i-N+2}^i)} \quad (3.9)$$

In sum, Kneser-Ney smoothing can be defined as:

$$P_{KN}(w_i|w_{i-N+1}^{i-1}) = \begin{cases} \frac{\text{count}(w_{i-N+1}^i) - d}{\sum_i \text{count}(w_{i-N+1}^i)} & \text{if } \text{count}(w_{i-N+1}^i) > 0 \\ \alpha(w_{i-N+1}^{i-1})P_{KN}(w_i|w_{i-N+2}^{i-1}) & \text{otherwise} \end{cases} \quad (3.10)$$

Chen and Goodman (1999) modified Kneser-Ney smoothing by using interpolation instead of backoff, and using three discount parameters instead of just one. The modified Kneser-Ney smoothing is the most widely used and best-performing version of Kneser-Ney smoothing.

In implementation, we use the SRILM toolkit (Stolcke, 2002) to learn the language models. Stopwords² and low frequency words (with a threshold of at least 4 occurrences) are removed from the text. Modified Kneser-Ney smoothing (Chen and Goodman, 1999) is used.

A combination of LMs trained on the British National Corpus (BNC) and POS LMs trained on the WeeBit corpus is used in our experiments to model the lexical usage and syntactic patterns in the text. The BNC is a 100 million word corpus of texts from a wide range of sources. We

²<http://www.ranks.nl/stopwords>

used the BNC as our background corpus because it is a large corpus covering a broad range of topics. Also, the date of the texts as well as the topics present in the BNC is similar to that of the WeeBit Corpus.

We train the LMs on BNC to learn the likelihood of different words and phrases in English texts. In a preliminary test, a unigram language model was trained on the BNC (Burnard, 1995) and then used to derive a high dimensional vector representation of the frequency/presence of specific words in the text. A simple information gain based feature selection metric was applied and the feature values were normalized by text length. However, the language model applied in this manner leads to very poor performance due to its high dimensionality. Therefore, in our experiments, the language modelling approach is only used to score each text by both its probability and perplexity. Perplexity is the inverse probability of the text normalized by the text length, which compensates for the bias towards shorter articles in language modelling. Word token N-gram models with $N \in [1, 5]$ trained on BNC are applied in this manner, resulting in 10 ($= 5 * 2$) numerical features.

The other way to build the language models is to train them on the WeeBit corpus itself. Language models can be trained on each grade level of the corpus and then used to compute the perplexity scores of the texts. However, language models trained directly from the WeeBit word tokens can overgeneralise to the WeeBit data. Instead, language models trained on POS sequences can better represent the lexical usage across the grade levels, are not topic dependent, and thus are more suitable for this research. In addition, higher order POS LMs can capture information about syntactic patterns in the texts across the grade levels. We generate a total of 50 features ($= 5 * 5 * 2$) by applying POS N-grams trained on 5 levels of text with n ranging from 1 to 5.

3.3.5 Discourse-based features

Discourse features measure the cohesion and coherence of the text. Cohesion means the connection of sentences at lexical and grammatical levels, whereas coherence represents the connection in the development of the ideas and arguments. Both cohesion and coherence can affect reading comprehension. Pitler and Nenkova (2008) were one of the first to explore the use of the features related to cohesion and coherence in readability assessment experiments. They demonstrated that discourse-based features are highly correlated with perceived text readability. Feng et al. (2010) used a few discourse features and evaluated their impact on the prediction of the grade level of the reading material for primary school students. Although they found that discourse features were not very useful in their task, they also pointed out that the texts in the corpus for primary school students are on the average of relatively low complexity and the discourse features may exhibit better discriminative power for texts at higher levels.

Three types of discourse-based features are used. These include entity density features, lexical chain features and entity grid features.

(1) Entity density features

Previous work by Feng et al. (2010, 2009) has shown that entity density is strongly associated with text comprehension. They believe that entities form basic components of concepts and propositions and they are the basis of higher level discourse processing; therefore the density of entities is important for comprehension. To extract the entity density features, an entity set is created as a union of named entities and general nouns (including nouns and proper nouns) contained in the text. Then named entities are extracted from the text using the Stanford Named Entity Recognizer (NER) (Finkel et al., 2005) and general nouns are detected from the output of RASP. The general nouns that are also a part of named entities (referred to as overlapping nouns) are removed from this set. Based on this, 10 entity density features are calculated:

- *total number of entity mentions per document*³
- *total number of unique entity mentions per document**
- *average number of entity mentions per sentence**
- *average number of unique entity mentions per sentence**
- *percentage of named entities out of all tokens per document*
- *percentage of unique entities out of all tokens per document*
- *percentage of named entities out of all tokens per sentence*
- *percentage of overlapping nouns removed out of all tokens*
- *percentage of named entities in all entities*
- *percentage of unique named entities in all unique entities**

(2) Lexical chain features

Lexical chains model the semantic relations among entities throughout the text. The lexical chaining algorithm developed by Galley and McKeown (2003) is implemented. The semantically related words for the nouns in the text, including synonyms, hypernyms, and hyponyms, are extracted from the WordNet (Miller, 1995). Then for each pair of the nouns in the text, we check whether they are semantically related. Finally lexical chains are built by linking semantically related nouns in text. The following lexical chain based features are computed. The length of a chain is the number of entities contained in the chain, the span of the chain is the distance between the index of the first and the last entity in a chain.

- *total number of lexical chains per document*

³We mark the novel features used in our experiments with asterisks.

- *total number of lexical chains normalized with text length**
- *average lexical chain length*
- *maximum lexical chain length*
- *average lexical chain span*
- *maximum lexical chain span*
- *the number of lexical chains that span longer than half of the document*

(3) Entity grid features

Another entity-based approach to measure text coherence is the entity grid model introduced by Barzilay and Lapata (2008). They represented each text by an entity grid, which is a two-dimensional array that captures the distribution of discourse entities across text sentences. Each grid cell contains the grammatical role of a particular entity in the specified sentence: whether it is a subject (S), an object (O), neither a subject nor an object (X), or absent from the sentence (-). A local entity transition is defined as the transition of the grammatical role of an entity from one sentence to the following sentence. In our experiment, we used the Brown Coreference Toolkit v1.0 (Eisner and Charniak, 2011) to generate the entity grid for the documents. The probabilities of the 16 types of local entity transition patterns within the text are calculated to represent the coherence of the text.

3.4 Experiments

3.4.1 Method

Supervised text readability assessment can be defined as the task of learning from the training data a function that maps the set of features describing different aspects of the input text to a numerical value indicating the difficulty of the input text. Depending on the requirements of the applications, readability assessment can be treated as a classification task, a regression problem or a ranking problem (Collins-Thompson, 2014). Most of the previous studies on reading difficulty explored the classification and regression methods with various statistical models. In our experiments, readability assessment is addressed as a supervised machine learning problem and a ranking approach is adopted and compared with a classification method. Intuitively, the reading difficulty of text is a continuous rather than a discrete variable. Text difficulty within a grade level can also vary. Instead of assigning an absolute grade level to the text, treating readability assessment as a ranking problem allows predicting the relative difficulty of pairs of documents, which captures the gradual nature of readability better. In addition, a ranking framework would be more useful in the future application for the retrieval of suitable

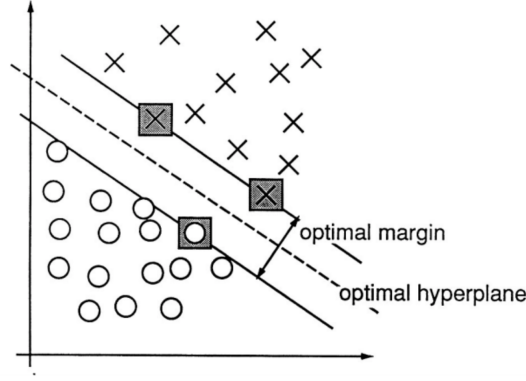


Figure 3.2: An example of a separable problem in a two dimensional space for SVM (Cortes and Vapnik, 1995)

reading material for L2 learners as the relative confidence score of readability prediction can be subsequently applied to rank the retrieved documents.

3.4.1.1 Classification model

Support vector machines (SVM) (Cortes and Vapnik, 1995) have been used in the past for readability assessment by many past researchers and have consistently yielded better results when compared to other statistical models for the task, including logistic regression and naive Bayes classifier (Kate et al., 2010; Feng et al., 2010). Most importantly, the SVMs usually perform quite well when the dataset is small (Forman and Cohen, 2004). Therefore they are used in our experiments.

An SVM classifier constructs a hyperplane in the feature space that separates the data points and maximizes the distance from the closest data points to the hyperplane (also called the margin), see Figure 3.2. Equivalently, it derives a decision function $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$, where x is the input and w and w_0 are the weight and bias term for the decision function, and maximizes the distance $r = \frac{f(\mathbf{x})}{\|\mathbf{w}\|}$. In addition, we want the data points $(\mathbf{x}_i, y_i)_{i=1, \dots, N}$ to fall on the correct side of the boundary, so we want $f(\mathbf{x}_i)y_i > 0$. Therefore we need to optimize,

$$\max_{\mathbf{w}, w_0} \min_{i=1}^N \frac{y_i(\mathbf{w}\mathbf{x}_i + w_0)}{\|\mathbf{w}\|} \quad (3.11)$$

Because rescaling the parameters \mathbf{w} and w_0 won't change the distance of any data points to the boundary, we can set the scale factor to $\min y_i f(\mathbf{x}_i) = 1$. Therefore the objective becomes,

$$\begin{aligned} \min_{\mathbf{w}, w_0} \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{subject to } y_i(\mathbf{w}^T \mathbf{x}_i + w_0) \geq 1 \\ \text{for } i = 1, \dots, N \end{aligned} \quad (3.12)$$

Here we minimize $\|\mathbf{w}\|^2$ instead of $\|\mathbf{w}\|$ for computational convenience and the $\frac{1}{2}$ is also added for convenience.

When the data points are not linearly separable, there is no feasible solution for $y_i f(\mathbf{x}_i) \geq 1$. In this case, slack variables $\xi_i \geq 0$ are introduced, so that we can still have $y_i f(\mathbf{x}_i) \geq 1 - \xi_i$ and we try to force ξ_i to be small. The objective is now modified to

$$\begin{aligned} \min_{\mathbf{w}, w_0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i \\ \text{subject to} \quad & \begin{cases} \xi_i \geq 0 \\ y_i(\mathbf{w}^T \mathbf{x}_i + w_0) \geq 1 - \xi_i \end{cases} \\ & \text{for } i = 1, \dots, N \end{aligned} \tag{3.13}$$

where C is a regularization parameter that controls the trade-off between maximizing the margin and tolerating misclassifications.

The SVM algorithm can be easily extended to nonlinear functions by applying the kernel trick. The resulting algorithm is similar in formulation, except that the original dot product is replaced by a nonlinear kernel function. This allows the algorithm to map the input feature vector x into an extended feature space F by $\Phi : \mathbf{x} \rightarrow \mathbf{k}$, where \mathbf{k} is the feature vector in the extended features space. Then the standard SVM algorithm can be applied.

In the case of multi-class classification with m classes, we can either use the *one-vs-all* approach to train m binary classifiers for which the data from the target class is treated as positive, and the data from all the other classes is treated as negative; or use the *one-vs-one* approach to train $\frac{m(m-1)}{2}$ binary classifiers to discriminate all pairs of classes and classify a point into the class which has the highest number of votes. The *one-vs-all* approach is likely to suffer from data imbalance and existing study (Hsu and Lin, 2002) has shown that the *one-vs-one* approach usually yields better results. Therefore the *one-vs-one* approach is used in our experiments.

3.4.1.2 Ranking model

In assessing readability, we compare a pairwise ranking model against the multi-class classification model.

Ranking algorithms are typically categorised into three groups by how they model the process of ranking: the pointwise approach, the pairwise approach, and the listwise approach (Liu, 2009). The pointwise approach assigns a score to each instance, the pairwise approach predicts which instance is better in a given pair of instances, and the listwise approach directly optimizes and predicts a ranked list. We use a pairwise ranking algorithm rather than a pointwise or listwise algorithm because it can better exploit the labelling information from the gold-standard grade

levels and it can model the relative difficulty of pairs of texts.

Among the pairwise ranking algorithms, ranking SVM (rankSVM) is one of the most commonly used methods (Lee and Lin, 2014). RankSVM is extended from the standard SVM algorithm by turning the ranking problem into a binary classification problem. It constructs a set of preference pairs $P = \{(i, j) | y_j > y_i\}$ from all the data points (\mathbf{x}_i, y_i) and tries to optimize:

$$\begin{aligned} \min_{\mathbf{w}, w_0} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i, \\ \text{subject to } \begin{cases} \xi_i \geq 0 \\ \mathbf{w}^T(\mathbf{x}_i - \mathbf{x}_j) \geq 1 - \xi_i \end{cases} \\ \text{for } (i, j) \in P \end{aligned} \quad (3.14)$$

which is equivalent to learning a standard SVM classifier that classifies the pairwise difference vectors $(\mathbf{x}_i - \mathbf{x}_j)$.

In prediction, the instances \mathbf{x} are ranked by the value of $\mathbf{w}^T \mathbf{x}$, where a larger $\mathbf{w}^T \mathbf{x}$ implies that the instance should be ranked higher (Chapelle and Keerthi, 2010).

3.4.1.3 Implementation

In our experiments, we use the LIBSVM toolkit (Chang and Lin, 2011) for training the multi-class classifier and the LIBSVM-rankSVM toolkit (Blondel et al., 2015) for learning the ranking model. In a preliminary experiment, we examined three different kernel functions (including linear, polynomial and Gaussian radial basis kernels) and found that the linear kernel outperforms other kernels and therefore it is used throughout the experiments. All the other internal parameters are kept as default when applying the toolkits. The features are scaled to the range $[-1, +1]$ attribute-wise before training and testing.

3.4.2 Evaluation

Five-fold cross validation is used for evaluation. The data for each of the grade levels is evenly divided into five validation folds and each of the validation folds is held out as the testing data in one of the cross validation runs, while the other folds are used as the training data in the same run. The final results are averaged across the cross validation runs.

We use two performance metrics for evaluation, accuracy (ACC) and Pearson Correlation Coefficient (PCC).

Accuracy is intuitively the most indicative measure of performance and most work on readability assessment used accuracy for evaluation. For the classification model, accuracy is

computed as the percentage of the documents that get correctly classified by the model.

$$ACC = \frac{\text{number of documents that are correctly classified}}{\text{total number of documents}} \quad (3.15)$$

For the ranking model, accuracy is measured pairwise. Pairwise accuracy (pairwise ACC) is defined as the percentage of instance pairs that the model ranked correctly. Assume we have a set of document instances (x_i, y_i) , where $x_i \in \mathbb{R}^n$ is the feature vector and $y_i \in \mathbb{R}$ is the target value.

Suppose the ranking model maps the input feature vector to an output rank score $f(x_i)$, then by definition pairwise accuracy is:

$$\text{pairwise ACC} = \frac{\{(i, j) \in P : f(x_i) > f(x_j)\}}{p} \quad (3.16)$$

where $P = (i, j) | y_i > y_j$ is the set of all possible instance pairs.

However, it should be noted that accuracy and pairwise accuracy are not directly comparable. Thus, PCC is introduced to compare the results of the classification and the ranking models.

PCC is a frequently used measure of the linear correlation between two variables. PCC ranges between -1 and 1, with 1 indicating total positive correlation, 0 no correlation, and -1 total negative correlation. PCC is invariant to linear transformations in location and scale in the two variables. The PCC between the model prediction Y_{pred} and the ground-truth label Y_{true} is defined as:

$$PCC = \frac{\text{cov}(Y_{pred}, Y_{true})}{\sigma_{Y_{pred}} \sigma_{Y_{true}}} \quad (3.17)$$

where cov is the covariance, $\sigma_{Y_{pred}}$ is the standard deviation of Y_{pred} , $\sigma_{Y_{true}}$ is the standard deviation of Y_{true} . When evaluating the ranking model, we compute the Pearson correlation between the ranking scores and the ground truth labels instead.

Yannakoudakis and Cummins (2015) compared the usefulness of several evaluation measures for automated text assessment. They found that although ranking correlation coefficients (such as Spearman's correlation and Kendall's τ_b) facilitate error analysis and system interpretation, their reliability decreases as the number of ties increases. As a result, PCC is preferred to a ranking correlation coefficient because of the large number of ties in the evaluation corpus.

3.4.3 Results and analysis

Feature set	dim	Classification		Ranking	
		ACC	PCC	pairwise ACC	PCC
traditional	7	0.586	0.770	0.862	0.704
lexical	32	0.578	0.726	0.863	0.743
syntactic	145	0.599	0.731	0.824	0.692
syntactic - GR subset	29	0.575	0.708	0.810	0.626
LM	60	0.714	0.848	0.872	0.769
discourse	33	0.563	0.688	0.848	0.659
all combined	277	0.803	0.900	0.922	0.824

Table 3.2: Classification and ranking results with feature sets grouped by their type

Table 3.2 shows how well each feature set performed and the result of applying a combination of all feature sets when using the classification and ranking models as well as the dimensionality of each feature set. Note that in Table 3.2 “syntactic - GR subset” refers to a subset of syntactic features that only contains GR complexity measures and parse tree based measures.

In predicting the text reading difficulty on the native data, the best result is achieved with a combination of all features and a classification model, with $ACC=0.803$ and $PCC=0.900$.⁴

Among all the feature sets, the language modelling features outperform all the other ones for both classification and ranking. It shows that the language models can capture a lot of information. It can be assumed that a combination of LMs trained on BNC and POS LMs on WeeBit can model both lexical and syntactic phenomena, and thus produce the best result. The discourse features are less predictive compared to the others in this task. This is not surprising, because although the documents are of five different levels, they are generally targeted at children of young age and tend to display less discourse complexity. Due to their relative simplicity in discourse, this set of features can not distinguish the documents well. The traditional measures, by contrast, have yielded relatively good results in readability estimation. Despite the fact that they are easy to compute and are of low dimensionality, their predictive power is relatively high. This explains why they have been very popular in early readability studies.

Comparing the classification and the ranking models, we can notice that the results of the two models vary across feature sets and none of the two models is consistently better than the other. When all features are combined, the classification model outperforms the ranking one. It suggests that a ranking model is not necessarily the best model in predicting readability overall when trained and tested on the same dataset.

Ablation tests are carried out to investigate how each individual feature set contributes to the overall model performance and detect possible redundancies in the feature set. In the ablation

⁴The best result is judged by PCC, as ACC and pairwise ACC is not directly comparable.

tests, we remove a single type of features each time and retrain the models on the remaining feature set. Table 3.3 shows the results of these experiments.

feature set	Classification		Ranking	
	ACC	PCC	pairwise ACC	PCC
all combined	0.803	0.900	0.922	0.824
- traditional	0.799	0.895	0.919	0.853
- lexical	0.801	0.892	0.915	0.849
- syntactic	0.794	0.896	0.919	0.832
- LM	0.706	0.846	0.899	0.799
- discourse	0.797	0.891	0.919	0.857
all with syntactic GR subset	0.804	0.900	0.924	0.848

Table 3.3: Results of the ablation tests

As Table 3.3 shows, the removal of each feature set leads to a decrease in model performance. And running significance tests using t -test for ACC and Williams’ test (Williams and Williams, 1959) for PCC reveals that the decrease is statistically significant in all cases. It shows that all the feature sets contribute to the overall model performance. The results of the ablation study are consistent with the individual feature set study. Both show that the LM features are the most useful feature set for the task. However, when replacing the original syntactic set with its subset that contains only GR-based complexity measures and parse tree based features, the results are improved for both classification and ranking models. This suggests that many syntactic features are redundant and the removal of the redundancy can lead to better performance. As a result, the combination of traditional, lexical, syntactic (GR-only), LM and discourses features is taken to be the optimal feature set for readability estimation on the WeeBit corpus.

Although there have been readability assessment studies on similar datasets, the results obtained in our experiments are not directly comparable to those. One of the major reasons is the modifications that we have made to the corpus (as discussed in Section 3.2). Vajjala and Meurers (2012) reported that a multilayer perceptron classifier using three traditional metrics alone yielded an accuracy of 70.3% on their version of the WeeBit corpus. Their final system achieved a classification accuracy of 93.3% on the five-class corpus. Nonetheless, the best system in our experiments yields results competitive to most existing studies. For reference, Feng et al. (2010) reported an accuracy of 74.01% using a combination of discourse, lexical and syntactic features for readability classification on their Weekly Reader Corpus and an accuracy of 63.18% when using all feature sets described in Schwarm and Ostendorf (2005).

3.5 Summary

In this chapter, we compared a classification model (SVM) and a ranking model (rankSVM) for automated readability assessment on the WeeBit corpus and investigated the effects of a range of readability measures on the task. The best performing system is the classification model, which yields an ACC of 0.804 and PCC of 0.900 with a combination of traditional, lexical, syntactic (GR-only), LM and discourses features. Although the classification model outperformed the ranking model with the optimal features set, the performance of the two models varies across feature combinations and none of the two models is consistently better than the other.

Chapter 4

Readability assessment on L2 data

4.1 Introduction

So far we have investigated the effect of various readability measures on the task of readability assessment and used two different types of models to predict text difficulty. However, the WeeBit corpus consists of texts aimed at native speakers of different ages rather than at L2 readers. Although there are certain similarities concerning reading comprehension between these two groups, the perceived difficulty of texts can be very different due to the difference in the pace and stages of language acquisition. Since the goal of our research is to automatically detect readability levels for language learners, it would be more helpful to work with data that are directly annotated with reading difficulty for L2 learners.

In this chapter, we focus on readability for L2 learners of English and present a level-graded dataset for non-native readability analysis. We explore methods that make use of the existing corpora aimed at native speakers to produce better estimation of readability when there is not enough data aimed at L2 learners. We then compare the importance of different types of readability features on native and L2 data and examine the optimal set of features for assessing L2 readability.

4.2 L2 Data: the Cambridge Exams dataset

Most work on readability assessment has been done on native corpora with age-specific reading levels (Schwarm and Ostendorf, 2005; Feng et al., 2010). Such texts are not aimed at L2 learners but rather at native-speaking children of different ages. Therefore, the level annotation in such texts is arrived at using criteria different from those that are relevant for L2 readers. The lack of significantly sized L2 level-annotated data raises a problem for readability analysis aimed at L2 readers. To tackle this, we created a dataset with texts tailored for L2 learners' readability specifically. The dataset is made publicly available.

The dataset is composed of reading passages from the five main suite Cambridge English

Exams, which include Key English Test (KET), Preliminary English Test (PET), First Certificate in English (FCE), Cambridge Advanced Certificate in English (CAE), and Cambridge Certificate of Proficiency in English (CPE).¹ These five exams are targeted at learners at A2–C2 levels of the Common European Framework of Reference for Languages (CEFR) (Verhelst et al., 2009).

The CEFR defines foreign language proficiency at six levels in increasing order: A1, A2, B1, B2, C1 and C2. Table 4.1 explains the overall requirements for reading ability by the six levels of CEFR (Verhelst et al., 2009).

Levels		Description
Proficient User	C2	Can understand and interpret critically virtually all forms of the written language including abstract, structurally complex, or highly colloquial literary and non-literary writings. Can understand a wide range of long and complex texts, appreciating subtle distinctions of style and implicit as well as explicit meaning.
	C1	Can understand in detail lengthy, complex texts, whether or not they relate to his/her own area of speciality, provided he/she can reread difficult sections.
Independent User	B2	Can understand the main ideas of complex text on both concrete and abstract topics, including technical discussions in his/her field of specialisation. Can read with a large degree of independence, adapting style and speed of reading to different texts and purposes, and using appropriate reference sources selectively. Has a broad active reading vocabulary, but may experience some difficulty with low-frequency idioms.
	B1	Can understand the main points of clear standard input on familiar matters regularly encountered in work, school, leisure, etc. Can read straightforward factual texts on subjects related to his/her field and interest with a satisfactory level of comprehension.
Basic User	A2	Can understand short, simple texts on familiar matters of a concrete type which consist of high frequency everyday or job-related language. Can understand sentences and frequently used expressions related to areas of most immediate relevance.
	A1	Can understand very short, simple texts a single phrase at a time, picking up familiar names, words and basic phrases and rereading as required.

Table 4.1: Overall requirements for reading comprehension by CEFR levels

The reading passages are harvested from the reading tasks in the past papers for each of the exams. The associated tasks, such as gap filling, factorial question answering, and multiple choice questions, are removed and only the reading texts are kept in the dataset. We use the level of the exams to describe the reading difficulty of the passages. The Cambridge English Exams are designed for L2 learners specifically and the test materials are constructed to ensure coverage of general topics at the appropriate levels of difficulty. Therefore the A2-C2 levels assigned to

¹<http://www.cambridgeenglish.org>

each reading paper can be considered as the level of difficulty of the reading passages for the L2 learners. However, it should be noted that there are different types of reading tasks such as multiple choice questions and gap filling questions in the exams. The type of the task may also have an effect on the reading difficulty of the texts, but this is beyond the scope of this work.

Table 4.2 shows the number of passages at each CEFR level across the dataset and the average length of text at each level measured by the number of sentences in the text.

Experimenting on the language testing data annotated with the L2 learner readability levels is one of the contributions of this research. Most previous work on readability assessment for English have relied on the data annotated with readability levels aimed at native speakers. In this work, we use language testing data with the levels assigned based on L2 learner levels, and it is believed that this level annotation is more appropriate for text readability assessment for L2 learners than using texts with the level annotation aimed at native speakers.

Exams	KET	PET	FCE	CAE	CPE
targeted level	A2	B1	B2	C1	C2
# of docs	64	60	71	67	69
avg. len. of text	14.75	19.48	38.07	45.76	39.97

Table 4.2: Statistics for the Cambridge English Exams data

Ideally, it would be good to train a model directly on text annotated with L2 levels and then use this model to estimate readability for the new texts. However, we think that the Cambridge Exams data we have compiled is relatively small, and the model trained on it will likely not generalize well. Therefore, we examine several approaches to make use of the native data for readability assessment on the L2 data. Nonetheless, training directly on the L2 data can serve as an upper bound for evaluating other models.

4.3 Experiments

4.3.1 Generalization of classification and ranking models

First, we test the generalization ability of the classification and ranking models trained on the WeeBit corpus on the Cambridge Exams data to see if it is possible to directly apply the models trained on native data to L2 data.

A classification model (SVM) and a ranking model (rankSVM) are trained using the same implementation as in previous experiments (see Section 3.4.1.3). We use the best performing feature combination to train both models, which includes 7 traditional features, 32 lexical features, 29 syntactic features (the GR subset), 60 LM features trained on WeeBit corpus and 33 discourse features, amounting to 161 features. The models are trained on the entire native data and then tested on the L2 data.

	Classification		Ranking	
	ACC	PCC	pairwise ACC	PCC
native data → L2 data	0.233	0.730	0.913	0.880
native data → native data	0.803	0.900	0.924	0.848

Table 4.3: Generalization results of the classification and ranking models trained on native data applied L2 data

Table 4.3 reports the results of the generalization experiment. Classification and ranking results on the native data (see Section 3.4.3) are provided here for comparison.

In the case of the multi-class classification model, the accuracy dropped greatly when the model is applied to the L2 dataset, while the correlation remained relatively high. This is expected because the Cambridge Exam levels and the WeeBit corpus levels are created with different standards and the reading difficulty of the levels do not match across the datasets. Looking at the confusion matrix of the classifier’s predictions on the L2 data (see Table 4.4), we can notice that most of the passages in the L2 data are assigned with the higher levels according to the WeeBit model. This is because, on average, the Cambridge Exams texts are more difficult than the WeeBit corpus ones, which are generally targeted at children of young ages. Thus, the mismatch between the targeted levels has led to poor generalization of the classification model.

Levels	1	2	3	4	5
A2	4	0	55	4	1
B1	0	0	24	6	30
B2	0	1	1	4	65
C1	0	0	0	3	64
C2	0	0	0	0	69

Table 4.4: Confusion matrix of the classification model on the L2 data, where the rows represent the gold-standard CEFR-based levels of the texts and the columns represent the WeeBit-based levels they got classified into.

In contrast, for the ranking model, both evaluation measures are relatively unharmed when the model is applied to the L2 data. It shows that, when generalizing to an unseen dataset, the estimation produced by the ranking model is able to maintain a high pairwise accuracy and correlation with the ground truth. This is because the ranking model does not try to band the texts into one of the levels on a different basis of difficulty annotation. Instead, pairwise ranking captures the relative reading difficulty of the texts, and therefore the resulting ranked positions of the passages are closer to the ground truth compared to the classification model.

The generalization experiment shows that, although the ranking model and the classification model perform similarly on the same dataset (see Section 3.4.3), the ranking model generalizes better when it is applied to an unseen dataset compared to the classification model. Further,

it suggests that we can make use of the native data for assessing L2 readability by training a ranking model on the native data and applying it to the L2 data directly, while preserving the accuracy on ranking L2 reading difficulty.

4.3.2 Mapping ranking scores to the CEFR levels

From the generalization experiment we can conclude that ranking is more accurate in predicting the reading difficulty of unseen learner texts than classification. Therefore, to estimate the CEFR level of the learner texts, it is more appropriate to make use of the more informative ranking scores produced by the ranking model. To do that, we use the model trained on the native data to rank the readability of the texts in the L2 data and learn a mapping function that bands the ranking scores into CEFR levels to predict the CEFR-based reading difficulty of the text.

In learning the mapping function, we adopt a five-fold cross-validation approach. We split the Cambridge Exams dataset into five cross validation folds, with approximately equal number of passages at each level in each fold. A mapping function that converts ranking scores into CEFR levels is learnt from the training folds and then tested on the validation fold in each run. The final results are averaged across the runs.

We compare three approaches to learn the mapping function: a regression-based approach, learning the cut-off boundary, and a classification-based approach.

4.3.2.1 Regression-based approach

Regression is a typical approach for modelling the relationship among variables. A regression function is learnt from the ranking scores and the ground truth labels on the training part of the L2 data and then applied to the validation part. The mapped CEFR prediction is then rounded to its closest integer and clamped to the range $[1, 5]$. Both linear regression and polynomial regression models are considered. Linear regression fits a linear function between two variables and polynomial regression fits a k th polynomial function between the variables. The intuition behind using polynomial functions instead of a simple linear function for mapping is that the correlation of ranking scores and CEFR levels is not necessarily linear. Therefore we explore using both linear and non-linear functions for learning the mapping. However, higher order polynomials can overfit the data and the resulting model may not generalize well.

There are different techniques for parameter estimation in regression. Among them, the Ordinary Least Square (OLS) estimator is the most common estimator for regression. It minimizes the sum of squared differences between the true value and the value predicted by the model. In addition, we apply the Theil-Sen estimator (Theil, 1992) to the regression models, and compare the results against the OLS estimator. The Theil-Sen estimator is a robust estimator, which fits the data by choosing the median of the slopes and intercepts of all lines through pairs of points. Compared to the OLS regression, the Theil-Sen estimator is insensitive to the outliers and noise

Regression functions	OLS		Theil-Sen	
	ACC	PCC	ACC	PCC
Linear regression	0.541	0.857	0.544	0.855
Polynomial (2nd)	0.544	0.858	0.574	0.865
Polynomial (3rd)	0.583	0.872	0.565	0.869
Polynomial (4th)	0.586	0.873	0.574	0.868
Polynomial (5th)	0.577	0.868	0.547	0.852

Table 4.5: Results of mapping ranking scores to the CEFR levels by regression, with ordinary least square (OLS) estimator and the Theil-Sen estimator

(Theil, 1992). Since the ranking scores are only estimates on the L2 reading passages, it could be beneficial to allow for some noise when learning the mapping function.

Table 4.5 lists the regression functions that were used to learn the mapping from the ranking scores to the CEFR levels and their results on this task. Linear regression and polynomial regression up to the 5th order were considered. The results show that the best regression function for the task is a 4th order polynomial function. This is because the lower order functions fails to model the relation between ranking scores and CEFR levels well due to their over-simplicity, while the 5th order polynomial function may have overfitted the data.

In addition, the results show that the Theil-Sen estimator is helpful for improving the result for lower order regression functions, where it is easier to estimate the noise. However, when it is applied to higher order functions, it fails to capture the outliers and results in poorer prediction.

4.3.2.2 Learning the cut-off boundary

To convert the ranking scores to the CEFR levels, we can also learn a separation boundary between the levels that bands the ranking scores to levels by maximizing the accuracy of such separation. For instance, we consider the ranked texts as a list with descending readability, with their ranking scores following the same order. If we could find a suitable cut-off boundary between each two adjacent levels in the list, then every text above the boundary would fall into the higher level, and all texts below the boundary into the lower level. In this way, the ranked texts are banded into five levels with four separation boundaries learnt.

The cut-off boundaries are learnt from the training part of the data and then applied to the validation part.

In the implementation, we experiment with two ways to use the data points for learning the cut-off boundaries: one is to use data points across all levels for learning a boundary, the other is to use only the data points in the adjacent reading levels for learning the cut-off boundary for the two levels. For example, when finding the appropriate cut-off boundary between level 3 and level 4, the first approach considers all data points in level 4 and level 5 as points above the boundary and all the data points in levels 1-3 as points below the boundary, while the second approach only considers data points in level 4 as above the boundary and data points in level 3 as

below the boundary.

The results of the two implementations for learning the separation boundary to predict the reading levels of texts are presented in Table 4.6.

Learning cut-off boundary	ACC	PCC
with all data points	0.562	0.872
with data points in adjacent levels	0.534	0.854

Table 4.6: Results of mapping ranking scores to CEFR levels by learning the cut-off boundary

The table shows that, between the two implementations, using all data points to learn the separation boundaries yields a better result than using only the points in adjacent levels for the task. This is because an optimal separation boundary should be able to classify the data points from higher and lower levels in addition to points from the adjacent levels, so that data points from other levels can also contribute to determining the position of the boundary. Also, there are more data to train the cut-off boundary on by making use of all the data points. Therefore this implementation makes better decisions on the optimal position of the boundary by taking all the data points into consideration.

4.3.2.3 Classification-based approach

The task of learning a mapping function from the ranking scores to predict the CEFR levels can also be addressed as a classification problem. The ranking scores can be considered as a single dimensional feature and the CEFR levels as the target value. Here, two approaches are adopted and compared, logistic regression and a linear SVM. As a matter of fact, the SVM approach can be considered as a variation of learning a separation boundary, as it tries to find an optimal decision boundary between the classes. The results are shown in Table 4.7.

Classifiers	ACC	PCC
logistic regression	0.610	0.862
linear SVM	0.622	0.864

Table 4.7: Results of mapping ranking scores to CEFR levels by classification

4.3.2.4 Results and analysis

Table 4.8 compares the best results of the three mapping methods. We also include the generalization result of the classification model from the native data to the L2 data for comparison.

Among the three approaches for mapping ranking scores to CEFR levels (regression-based, separation boundary-based, and classification-based), the classification ones showed better results than the others in terms of accuracy. Though not as high in accuracy as the SVM, a polynomial mapping function also yielded very good results in terms of PCC. Compared to the other two

Mapping functions	ACC	PCC
4th order polynomial regression	0.586	0.873
cut-off boundary	0.562	0.872
linear SVM	0.622	0.864
naive generalization (in Section 4.3.1)	0.233	0.730

Table 4.8: Results of mapping ranking scores to CEFR levels

methods, the separation boundary-based approach performs better than a linear regression function but fails to match the polynomial regression and classification-based methods.

When looking at the confusion matrix produced by the linear SVM approach for learning the mapping (see Table 4.9), we can see that the mapped prediction makes more mistakes at the higher levels. This is because as the ranking model trained on the WeeBit corpus has only learned about lower readability levels (the texts targeted at young children), therefore it cannot distinguish the higher level reading passages well.

	A2	B1	B2	C1	C2
A2	56	8	0	0	0
B1	8	42	9	0	1
B2	1	8	46	9	7
C1	0	1	24	13	29
C2	0	1	6	15	47

Table 4.9: The confusion matrix produced by SVM for mapping ranking scores to CEFR levels, where rows represent the gold standard CEFR levels, and columns represent the mapped levels

Nonetheless, all three approaches considerably outperform the naive generalization of the classification model from the WeeBit corpus to the Cambridge Exams data. These improvements are statistically significant at $p < 0.05$ level using t -test for ACC and Williams’ test (Williams and Williams, 1959) for PCC.

4.3.3 Domain adaptation from native to L2 data

Another way to make use of the native data is to treat the task as a domain adaptation problem, where the native data is taken as the source domain, and the L2 data as the target domain. The idea behind this is to use the out-of-domain training data to boost the performance on the limited in-domain data.

Domain adaptation techniques aim to learn from a resource-rich source domain an algorithm that performs well on a different but related resource-poor target domain. Depending on the availability of annotation on the target corpus, domain adaptation can be divided into supervised or semi-supervised cases. In our case, the source domain is the larger WeeBit corpus and

the target domain is the Cambridge Exams data, and we want to learn a model that exploits information from the native data and make it useful for predicting the reading difficulty of L2 data. To do this, a supervised domain adaptation method is adopted for this task.

EasyAdapt (Daumé III, 2007) is a simple yet effective domain adaptation algorithm. It has been used in various NLP tasks and showed good results, including machine translation (Green et al., 2014), short answer scoring (Heilman and Madnani, 2013), and essay scoring (Phandi et al., 2015).

EasyAdapt augments the original feature space of the data to encode general and target-specific features. More formally, suppose we have a feature vector x in the original feature space \mathbb{R}^F . In a two domain case, EasyAdapt expands the input feature space from \mathbb{R}^F to \mathbb{R}^{3F} , and then applies two mapping functions $\Phi^S(\mathbf{x})$ and $\Phi^T(\mathbf{x})$ on source domain data and target domain data input vectors respectively. Among the augmented feature dimensions, the first dimensions encode information from the general domain, the second dimensions encode the source domain, and the third dimensions encode the target domain.

$$\Phi^S(\mathbf{x}) = \langle \mathbf{x}, \mathbf{x}, \mathbf{0} \rangle, \Phi^T(\mathbf{x}) = \langle \mathbf{x}, \mathbf{0}, \mathbf{x} \rangle \quad (4.1)$$

Here, $\mathbf{0} = \langle 0, \dots, 0 \rangle \in \mathbb{R}^F$ is the zero vector.

In this manner, the instance feature vectors from the WeeBit corpus and Cambridge Exams datasets are augmented to three times their original dimensionality. The augmented feature space captures both general and domain specific information and is thus capable of generalizing source domain knowledge to facilitate estimation on the target domain.

A ranking model (rankSVM) is trained on the augmented feature space and a five-fold cross-validation is used as in the previous experiments. Since there is a mismatch in reading difficulty between the levels on native and L2 data, the pairwise ranking algorithm needs to be adapted to ensure that the preference pairs are only created from the same domain.

Table 4.10 shows the results of applying EasyAdapt with the ranking model. For comparison, the table also presents the results obtained when we apply the model trained on the native data to the L2 data directly, and the results obtained when we train the ranking model on the L2 data only. We can see that ranking with EasyAdapt outperforms the naive generalization approach significantly ($p < 0.05$), but it does not beat the results obtained when training a model on the L2 data directly.

	pairwise ACC	PCC
EasyAdapt	0.933	0.905
native data \rightarrow L2 data	0.913	0.880
L2 data \rightarrow L2 data	0.943	0.913

Table 4.10: Results of domain adaptation from native to L2 data with the ranking model

After applying the ranking model with EasyAdapt, the ranking scores can be converted to

CEFR levels using the same methods as described in Section 4.3.2. The best mapped CEFR estimation is achieved with a linear SVM classifier on the ranking score, reaching an ACC of 0.707 and PCC of 0.899.

4.3.4 Using self-training to enhance the classification model

In addition to the domain adaptation, we explore using self-training to boost the performance on the limited L2 data with the native data.

Self-training is a commonly used semi-supervised machine learning algorithm that aims to use a large amount of unlabelled data to help build a better classifier on a small amount of labelled data (Zhu, 2005). In self-training, a supervised model is first trained on the small amount of labelled data and then used to classify the unlabelled data. The most confident unlabelled instances are selected and added to the training set. Then the model is re-trained and the procedure is repeated. Self-training can be considered as a wrapping algorithm and applied to many different supervised machine learning algorithms (Zhu, 2005). Unlike the supervised domain adaptation algorithm where the model is usually trained on the larger and labelled source domain and adapted to the target domain, self-training makes use of a large amount of unlabelled data by augmenting the data in the training set. Self-training is also a popular technique in NLP tasks where there is usually only a small amount of annotated data and plenty of unannotated data, such as text classification (Nigam et al., 2000) and parsing (McClosky et al., 2006).

When using native data to boost model performance on L2 data with self-training, the L2 data is regarded as labelled instances, and the native data as unlabelled ones. A classification model (SVM) is trained on the L2 data and then used to score the native data. The most confident K instances as well as their labels are added to the training set and then the model is re-trained. The confidence of the SVM classifier is obtained from mapping the the distance of the data point from the hyperplane to a probability score via a logistic function. A five-fold cross-validation is used in evaluation as before.

In a preliminary test, we experimented with a grid search on K 's and the number of iterations, and found that whatever the choice of the parameters is, the model performance degrades with self-training when the unlabelled instances are added blindly to all levels of the L2 dataset. Taking into account the mismatch in the difficulty levels between the native and L2 texts and the lack of difficult texts in the WeeBit corpus, the algorithm is adapted to add the unlabelled data only to the lower three levels of the L2 dataset.

Following this approach, the best result is achieved with $K=10$ and 9 iterations, with 270 texts added in total, reaching $ACC = 0.797$ and $PCC = 0.938$. However, the effect of self-training degrades when too much unlabelled data are selected and the process makes the annotation of the training set unreliable and unbalanced.

It seems reasonable to compare the results of this approach to those obtained with a classifi-

Type	ACC	PCC
L2 data only	0.785	0.924
self-training	0.797	0.938

Table 4.11: Results of applying the self-training method to enhance the classification model

classification model that is trained directly on the L2 data. Hence, we include the results of this model in Table 4.11 for comparison. The results show that self-training can significantly ($p < 0.05$) help estimating readability for L2 texts by including a certain amount of unlabelled data (in this case, the native data) in training. However, the range of the reading difficulty covered by the unlabelled data may also influence the model performance.

4.3.5 Comparing readability features on native and L2 data

To compare the predictive power of different types of readability features on the native data and the L2 data, a set of experiments for individual feature sets and an ablation study are carried out on the Cambridge Exams data, with the same settings as in the experiments performed on the native data (see Section 3.4.3). The same five fold cross validation approach as in previous sections is used. A set of classification models (SVM) and a set of ranking models (rankSVM) are trained on the training folds and tested on the validation folds.

The results of these experiments are given in Table 4.12 and Table 4.13.

Feature set	Classification		Ranking	
	ACC	PCC	pairwise ACC	PCC
traditional	0.680	0.843	0.903	0.842
lexical	0.737	0.892	0.921	0.901
syntactic	0.450	0.588	0.790	0.653
syntactic - GR subset	0.459	0.633	0.781	0.626
LM	0.634	0.797	0.885	0.817
discourse	0.562	0.773	0.882	0.794
all combined	0.741	0.895	0.934	0.905

Table 4.12: Classification and ranking results with the individual feature sets on the L2 data

Comparing the feature sets performance on the L2 data to the results on the native data (see Section 3.4.3), we can see that the lexical and discourse features play a more important role for the L2 data than for the WeeBit corpus. This is due to the fact that the variation in lexical usage and sophistication in the discourse structures in the Cambridge Exams data are greater than those in the WeeBit corpus. The traditional features also correlate better with the readability levels of the L2 data. The LM features are less predictive on the Cambridge Exams data, nonetheless they are still producing relatively good results. This is not surprising because the POS LMs trained on the WeeBit data can no longer capture the lexical usage or the grammatical aspects of the

Feature set	Classification		Ranking	
	ACC	PCC	pairwise ACC	PCC
all combined	0.741	0.895	0.934	0.905
- traditional	0.725	0.884	0.933	0.906
- lexical	0.692	0.859	0.922	0.882
- syntactic	0.749	0.889	0.944	0.911
- LM	0.716	0.881	0.924	0.892
- discourse	0.689	0.870	0.925	0.897
all with syntactic GR subset	0.785	0.924	0.943	0.913

Table 4.13: Results of the ablation study on the L2 data

sentences in the learner’s data as well as on the WeeBit data itself, because of the mismatch in the training and test data for the LMs. Nonetheless this type of feature can generalize relatively well because the POS LMs from WeeBit and LMs trained from BNC are relatively domain independent. We can also observe that the syntactic feature set, due to many redundant features in it, has actually harmed the performance. When it is replaced with the GR-based subset, the results improve.

The best performance on the L2 data is achieved using a combination of traditional, lexical, GR-based syntactic, LM and discourse features, with $ACC = 0.785$, $PCC = 0.924$ for the classification model and pairwise $ACC = 0.943$, $PCC = 0.913$ for the ranking model.

4.4 Summary

One of the major challenges in assessing text readability for the L2 readers is the lack of significantly sized level-annotated data. To address this, we collected a dataset of CEFR-graded texts tailored for learners of English as an L2 and investigated text readability assessment for both native and L2 learners. We applied a generalization method to adapt models trained on larger native corpora to estimate text readability for learners, and found that the ranking model generalizes better than the classification model when it is applied to the unseen L2 data. We explored domain adaptation and self-learning techniques to make use of the native data to improve system performance on the limited L2 data. We compared the linguistic features on predicting reading difficulty on texts aimed at native readers and texts aimed at L2 readers and found that the traditional, lexical and discourse features play a more important role for the L2 readability estimation. In our experiments, the best performing model for readability estimation on learner data achieves an accuracy of 0.797 and PCC of 0.938 using the classification model and self-training.

Chapter 5

Automated evaluation of learner summarisation for reading comprehension

5.1 Introduction

Summarisation is a well-established method in traditional English examination to measure reading proficiency. It is considered an effective approach to test both cognitive and contextual dimensions of reading (Weir et al., 2013). However, due to the inefficiency in manual assessment of summaries, modern English exams usually replace the summarisation task with multiple choice or short answer questions that are easier to score (Alderson, 2005). Automating the assessment of learner summarisation provides an efficient evaluation method for the quality of the learner summary and can lead to effective educational applications to enhance reading comprehension.

In this chapter, we describe the framework of a summarisation task for assessing reading comprehension and propose three approaches to assess the quality of a summary. In the first approach, we extract features from different levels of summary representations to evaluate the semantic similarity and relevance of the summary with respect to the original reading passage. In the second approach, we define a similarity matrix and learn patterns from the matrix to assess the summary. In the third approach, we train neural network based models in an end-to-end fashion for summary assessment. In addition, we use an ensemble modelling method to combine models trained independently using the three approaches into a single system. We evaluate our proposed methods on two datasets we collected and compare the models with the results from the related work.

5.2 Summarisation task

Before we talk about the automated approach to evaluating learner summaries, we describe the summarisation task used in our experiments in this section. The summarisation task is planned to

be a part of the “Read and Improve” framework designed to help learners improve their reading skills.

Read & Improve

TextID: B1, 4

[Return to Instructions](#)

HONEY

Honey is a sweet liquid made by bees. It consists of water and sugars. Bees may travel as far as seventy-five thousand kilometres and visit over two million flowers to produce just half a kilo of honey. The colour and flavour of honey depend on the type of flower visited. In fact, there are more than three hundred varieties of honey.

The lighter-coloured ones are generally milder in flavour than darker honey.

In ancient times, honey was the main sweet food, as sugar was very rare. Honey was of great value to the ancient Egyptians, who used it as payment.

Today, honey is produced and eaten in every part of the world. Research suggests that it prevents tiredness and improves athletic performance. However, honey is not just food - it can be taken for sore throats and is used in many skin and hair-care products.

Summary

Instructions: Please write a summary of approximately **50 words** for the text above.

[Save](#)

[Next](#)

Figure 5.1: An illustration of the summarisation task in the “Read and Improve” framework

In this framework, learners, regardless of their ability level, are asked to complete three summarisation tasks:

1. Task 1: read a passage at B1 level and write a 50-word summary based on the comprehension of the passage.
2. Task 2: read a passage at B2 level and write a 100-word summary of the passage.
3. Task 3: read a passage at C1 level and write a 120-word summary of the passage.

As the reading level of the original passage increases over the three summarisation tasks, the text also gets longer. As a result, the word limits on the summarisation tasks are set to keep a relatively constant compression ratio between the summary and the original passage. No specific time limit for writing the summary was specified at the time of this experiment.

An example summarisation task can be seen in Figure 5.1.

The following instructions are given to the learners before they are asked to write the summaries:

There are three passages in this task. For each passage please write a summary that is at approximately the corresponding word limit in your own words. Please read each passage carefully before writing its summary.

A summary should be informative of the original text. Please write complete and coherent sentences and do not enumerate pieces of information in bulleted lists.

The original texts for these summaries are reading passages from the Cambridge English Exams (Xia et al., 2016). In total, 18 (= 6 passages * 3 levels) reading passages are used in the framework. The same set of passages are presented to learners at different levels, including B1, B2 and C1. The three reading passages presented to each learner are randomly selected from the set of reading passages.

Each summary written by the learner was graded on a 6-point scale. The criteria for the assessment scale are defined as:

Band 5: *The summary demonstrates excellent understanding of the original passage: Content covers all the main points of the passage. All content included is accurate, with no irrelevant details or repetitions. Target reader is fully informed.*

Band 4: *Performance shares features of Bands 3 and 5.*

Band 3: *The summary demonstrates acceptable understanding of the passage: Most of the main points are included. Most of the content is relevant and paraphrased, with some irrelevant details, repetitions or inaccuracy of content. Target reader is on the whole informed.*

Band 2: *Performance shares features of Bands 1 and 3.*

Band 1: *The summary demonstrates very little understanding of the passage: Most of the content is of limited relevance, with repetitions or verbatim borrowing from the original text. In some paraphrased parts of the text, inaccuracy of content or omissions of main points are evident. Target reader is minimally informed.*

Band 0: *No understanding of the passage is demonstrated. The content is totally irrelevant to the original passage. Target reader is not informed.*

In general, a good summary is considered to be a concise text that accounts for all the main points in the original passage. The content of a good summary should be relevant and accurate, so that a reader can be put in possession of the gist of the original passage without reading it. The summarisation task also asks the learner to express the content of the summary in their

own words. While verbatim borrowing from the reading passage is not entirely forbidden in writing the summary, it is penalized if the summary does not demonstrate understanding of the passage. A summary made up of a series of copied sentences from the passage is marked down accordingly. Additionally, summaries that are too short are likely to lead the learner to over-abstract and omission of important information, and long summaries exceeding the word limit are likely to be diffused or irrelevant, and both are marked down accordingly. It is expected that learners at higher proficiency levels produce better summaries especially when the reading difficulty of the passages increases.

As we want to assess the reading comprehension of the learner, the criteria for the assessment scale emphasizes the content and comprehension aspects of the summary rather than the writing, in order to minimize the influence of the learner's writing skills on assessing their summary. Specifically, inaccuracy in syntax and vocabulary is not punished during the grading.

The assessment scale is not presented to the learners before they write the summary.

5.3 Data

5.3.1 Simulated learner data

Before collecting summaries produced by real learners, we use simulated learner summaries to develop the automated summary evaluation system. For that, we asked 50 members of our university to write a "good summary" and a "bad summary" for the 3 reading passages of different levels of difficulty. The participants were invited through email and they were given a webpage link with contents similar to Figure 5.1 to complete the three summarization tasks. In total, 300 summaries were collected (with 150 good summaries and 150 bad summaries). The simulated learner data is then used to train binary classification systems to assess whether a summary has properly summarized the original passage or not.

The same instructions as in the summarisation task (see section 5.2) are given to the participants for writing the good summary. Since the participants are all proficient users of English, it is assumed that the good summaries written by them for all three levels of reading passages are of the appropriate quality.

No specific instructions on how to write a bad summary were given to the participants. However, they were asked to produce grammatically correct sentences in the bad summary and to write a bad summary with content that mimics what a learner who does not fully understand the original passage would produce. The participants were allowed to take different strategies as they find proper to write the bad summary.

When examining the summaries after the data was collected, we found that there are a few typical strategies used by the participants to write a bad summary:

- (i) The summary is made up of a series of copied sentences from the original passage and the

copied content does not cover the main points of the passage.

- (ii) The summary is off-topic to the original passage. However, the summary is usually phrased with words that have appeared in the passage.
- (iii) The summary does not fully cover the main points of the passage, with noticeable omissions.
- (iv) The summary focuses on details in the passage, rather than the main ideas.
- (v) The summary contains invented or inaccurate information.

In many cases, the participants used a combination of these strategies to produce bad summaries. Despite the fact that the bad summaries are only emulations of what language learners would produce, we believe that the same problems are expected to be present in actual learners' summaries. Examples of good and bad summaries are given in Appendix D.

5.3.2 Real learner data

Learner levels	Count
B1	40
B2	40
C1-C2	57
Total	137

Table 5.1: The distribution of the proficiency levels of the learners

Summaries were collected from learners at B1, B2 and C1-C2 levels of proficiency using the framework described in Section 5.2. The learners were recruited by researchers from Cambridge Assessment. They were provided with a webpage link to complete the summarization tasks. The proficiency levels of learners are self-identified. They are usually based on placement exams by language schools or language certificates the learners have obtained. All learners were asked to complete three summarisation tasks for the three levels of reading passages. In total, 411 summaries from 137 learners are collected. The distribution of the proficiency levels of the learners is shown in Table 5.1. Examples of learner summaries are given in Appendix E.

Pair of annotators	PCC
Annotator A - Annotator B	0.693
Annotator B - Annotator C	0.690
Annotator B - Annotator C	0.794
Average	0.726

Table 5.2: Inter-annotator agreement on the learner data

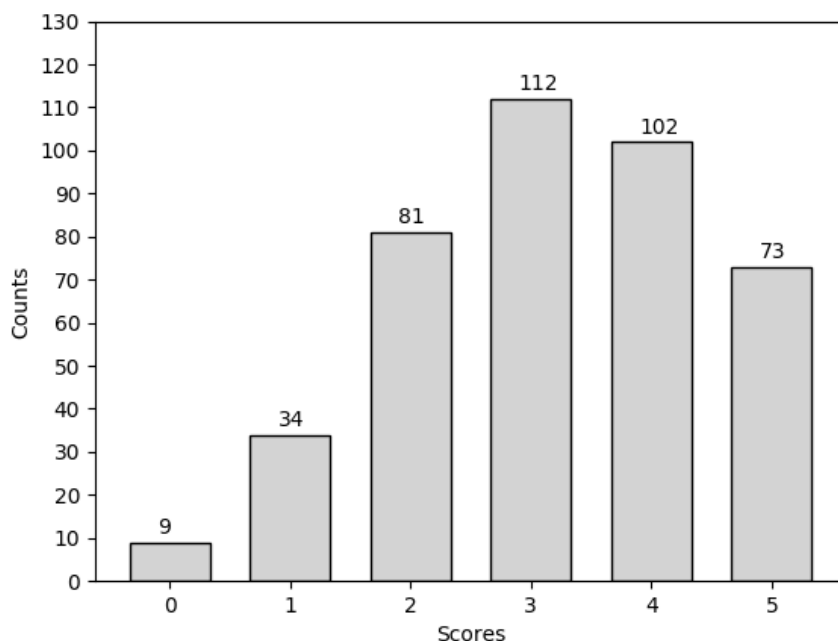


Figure 5.2: A histogram illustrating the score distribution in the learner summaries

The summaries are then scored by three annotators on the 6-point scale described previously. The Pearson correlation for the scores by each pair of annotators is shown in Table 5.2. The average correlation among the annotators is 0.726. We take the mean of the scores by the three examiners as the final annotation for the summaries. Figure 5.2 shows the distribution of the scores for the learner summaries.

Instead of treating the summary scores as categorical classes, we train regression models on the real learner data to predict the score of the summary, in order to make better use of the continuity of the assessment scale.

In the next three sections, we introduce three different approaches to automatically evaluate the learner summary.

5.4 Method 1: Measures for summary assessment

In the first approach, we extract a number of features to describe the similarity of the summary and the reading text and apply a machine learning model to learn and predict the summary quality.

The summarisation task for reading comprehension examines the content relevance and the ability to convey the main ideas of the text in the summary. To automatically assess the learner summary, the candidate summary is compared against a reference to assess the quality of its content.

We experiment with two types of references to evaluate the candidate summary: one is to compare the candidate summary against the original passage directly, the other is to extract key sentences from the original text with an automated extractive summarizer and compare the candidate summary to the key sentences. Ideally, an extractive summarizer identifies the salient text units in the original passage and extracts a subset of sentences from the passage that are representative of the original text. Although the extracted key sentences are not necessarily coherent among themselves, they provide a representation of the main ideas of the text. Comparing the candidate summary against the key sentences allows us to examine the content relevance and the coverage of the main ideas in the candidate summary.

We compare two popular extractive summarizers in selecting the key sentences for reference: **TextRank** (Mihalcea and Tarau, 2004) and **MEAD** (Radev et al., 2004a). We also compare the extractive summarizers against the baseline of using a random selection of sentences as the reference.

TextRank is a graph-based ranking model for text processing that has been successfully applied to keyword/sentence extraction adapted from PageRank (Page et al., 1999). The model converts a text into a highly connected graph, where text units of various sizes (e.g. words, phrases, sentences) are added as the vertices in the graph. Relations between each two lexical units (e.g. similarity of sentences and co-occurrence of words) form the edges between such vertices. The edges are weighted by the strength of connection between the two lexical units. The vertices are then scored by the vertices connected to them and the weights of the connected edges.

Formally, let $G = (V, E)$ be an undirected graph with a set of vertices V and a set of edges E . For a given vertex V_i , let $Adj(V_i)$ be the set of vertices connected to V_i , and w_{ij} be the weight on the edge between two vertices V_i and V_j . The score of a vertex V_i is defined as:

$$S(V_i) = (1 - d) + d * \sum_{V_j \in Adj(V_i)} \frac{w_{ij}}{\sum_{V_k \in Adj(V_i)} w_{ik}} S(V_j) \quad (5.1)$$

where d is a damping factor between 0 and 1, which represents the probability of jumping from a given vertex to another random vertex in the graph. The factor d is set to 0.85 in the work by Mihalcea and Tarau (2004) and also in our implementation. The graph is initialized with arbitrary scores on the vertices and the computation is iterated until the scores converge.

After the ranking algorithm is run on the graph, a score is assigned to each vertex. Vertices with higher scores are considered more important and are extracted as salient units to represent the text.

When TextRank is used for key sentence extraction, a vertex is added to the graph for each sentence in the text and edges are added between the vertices with weights indicating the strength of the connections between each pair of sentences. The weight between two sentences can be determined simply as the number of common tokens between the lexical representations of the

two sentences. To avoid assigning high weights to long sentences, a normalizing factor is applied on the weights. Formally, given two sentences S_i and S_j , the weight w_{ij} assigned to the edge between them is defined as:

$$w_{ij} = \frac{|\{t|t \in S_i \text{ and } t \in S_j\}|}{\log(|S_i|) + \log(|S_j|)} \quad (5.2)$$

, where $|S|$ is the number of words in the sentence.

Sentences that are related to many other sentences are likely to be central and would have higher weights. After running the algorithm, the top-ranked sentences are selected as the reference.

MEAD is a publicly available platform for extractive summarisation and it is known to perform generally well on a range of texts (Radev et al., 2004a). MEAD implements a number of summarisation algorithms and extracts a range of features on a sentence-by-sentence basis. The features implemented in MEAD include: cosine overlap with the centroid vector of the document (Radev et al., 2004b), cosine overlap with the first sentence in the document, the length of the sentence in words, whether the length of the sentence exceeds a defined threshold, the position of the sentence in the document, keyword match, and the sentence score calculated using the LexRank algorithm (Erkan and Radev, 2004). The centroid vector is computed using the frequencies of occurrences of topic words in the document, where the topic words are decided by thresholding the TF*IDF weightings of all words in the document. LexRank is a graph-based method for computing the relative importance of text units in the document. When applied to sentences, it computes a centrality score for each sentence to represent its importance. To do that, it first build a similarity graph, where sentences in the document form the nodes of the graph, and cosine similarities between each pair of sentences form the edges. Similar to TextRank, it applies a PageRank-like algorithm to compute the centrality score for each node using the scores of its adjacent nodes, weighted by the cosine similarity between the nodes. Having extracted the features, MEAD uses a linear combination of all the features to rank the sentences and extracts a set of top-ranked sentences that satisfy the length requirement to form the reference.

After obtaining the reference, we derive a range of linguistic features to evaluate the quality of the learner summary for reading comprehension.

Four types of features are used to assess the summary: verbatim features, semantic similarity features, features based on distributed vector representations of the summary, and features to describe discourse and other textual characteristics of the summary. The first three types of features are derived from comparing the candidate summary against the reference, while the last type of feature is extracted from the candidate summary itself. When comparing the candidate summary with the reference, distributed representations of text units are constructed at the word, sentence and document levels and used along with verbatim overlap measures to capture semantic

similarity and content relevance of the learner summary with respect to the reference. On the other hand, discourse-based features and other textual features are extracted from the learner summary itself to provide an assessment of the coherence and textual quality of the summary.

In the following sections, we describe the range of features we use to evaluate the learner summaries in more detail.

5.4.1 Verbatim features

Verbatim features evaluate the lexical overlap of the text units between the candidate summary and the reference. Verbatim similarity is the most straightforward measure that indicates content similarity. If the candidate summary and the reference are using exactly the same words and phrases in the sentences, the longer the lexical overlap is, usually the more similar the meanings of the two texts are. Also, verbatim similarity is easy to compute by counting the words and phrases the two texts have in common. However, while verbatim overlap suggests similarity between the texts, extensive use of verbatim borrowing in the summary can be a sign of poor comprehension, and a summary made up of copied sentences from the reading passage might still be partial or logically inconsistent. Therefore, a high verbatim similarity is not necessarily an indicator of a good summary.

We use the following metrics to measure verbatim similarity:

ROUGE (Lin, 2004) is a set of metrics commonly used in evaluating automated summarisation systems. It is a recall-based metric that measures the N-gram overlap between the candidate summary and the reference. In addition to verbatim similarity, a high value of ROUGE score usually indicates that the content of the reference is mostly covered by the candidate summary. ROUGE was discussed in Section 2.2. We make use the ROUGE package¹ to extract the ROUGE-N (with N ranging from 1-4), ROUGE-L, ROUGE-W, ROUGE-S, and ROUGE-SU scores.

BLEU (Papineni et al., 2002) is extensively used in the evaluation of machine translation. It has also been applied to paraphrase identification (Madnani et al., 2012) and summarisation evaluation (Madnani et al., 2013) and showed good results in these tasks. BLEU measures the precision-based N-gram overlap between the candidate and the reference summary. As a precision-based metric, it reflects the extent to which words and phrases in the candidate summary are borrowed from the reading passage. BLEU was also discussed in Section 2.2. We use the *NLTK.translate* package² to calculate BLEU scores from unigram to 4-gram.

METEOR (Denkowski and Lavie, 2011) is another word overlap based metric originally used for machine translation evaluation. However, it extends the exact word overlap with word stem matches extracted with the Porter stemmer (Porter, 1980) and synonym matches extracted

¹The original webpage is no longer maintained. A copy of the original ROUGE package can be found at <https://github.com/kylehg/summarizer/blob/master/rouge/ROUGE-1.5.5.pl>

²https://www.nltk.org/_modules/nltk/translate/bleu_score.html

from the WordNet (Miller, 1995), which allows for better coverage in measuring lexical overlap. To do that, METEOR creates a word alignment between the pair of texts it compares, such that every word in each text maps to at most one word in the other text. The alignment is incrementally produced by identifying all possible matches for exact words, word stems and synonyms between the pair of texts and selecting the largest subset of word mappings that constitutes an alignment. If more than one alignment is found, METEOR selects the alignment that better preserves the word order in both texts. Once an alignment is produced, METEOR computes the unigram precision, recall and F-score (weighted harmonic mean between precision and recall) from the word mapping. Formally, let m be the number of mapped unigrams found between the two texts, n_c be the total number of unigrams in the candidate summary and n_r be the total number of unigrams in the reference, the F-score of the alignment is:

$$F_{score} = \frac{P \cdot R}{\alpha \cdot P + (1 - \alpha) \cdot R} \quad (5.3)$$

where $P = m/n_c$, $R = m/n_r$, and α is a parameter.

Additionally, METEOR computes a penalty for the alignment to measure the extent to which the word order is preserved in the alignment. To calculate the penalty, the matched unigrams are divided into the fewest possible number of “chunks” such that the matched unigrams in each chunk are adjacent in both texts and have an identical word order. Then a fragmentation fraction is computed between the number of chunks (ch) and the number of matched unigrams (m): $frag = ch/m$. The penalty is then computed as:

$$Pen = \gamma \cdot frag^\beta \quad (5.4)$$

Finally, the METEOR score for the candidate summary is defined as:

$$METEOR = (1 - Pen) \cdot F_{score} \quad (5.5)$$

The three parameters are set to be $\alpha = 0.9$, $\beta = 3.0$, $\gamma = 0.5$, which is the default setting in METEOR.

Compared to ROUGE and BLEU, METEOR only measures unigram overlap in the texts. However, it takes into account the word order in measuring verbatim similarity by penalizing crossing word order in the alignment. Also, METEOR’s flexible word matching addresses the variability in expression, allowing for morphological variants and synonyms to be evaluated. We make use of the original METEOR package³ to compute the score.

³<http://www.cs.cmu.edu/~alavie/METEOR/>

5.4.2 Semantic similarity features

Although verbatim overlap metrics prove to be effective in various tasks, they fail to capture the content similarity when paraphrasing and higher levels of abstraction are used in the summary. To compensate for this, word embeddings and sentence embeddings are used to model text semantic similarity at the word and the sentence level. We measure the semantic similarity between words and sentences in the texts and combine the scores into a measure of document-level semantic similarity.

An embedding is the representation of a word (or a sentence) as a vector. Vector models of word meaning have been used in NLP for over 50 years (Jurafsky and Martin, 2008). They are commonly used as features to represent word meaning in a range of applications such as indexing, textual entailment, and information retrieval. Vector models are also the most common approach to compute semantic similarity between words, sentences, and documents.

The traditional way to represent words in the vector space is based on counting the word co-occurrences in a training corpus. Recently, neural network based vector models have been shown to outperform traditional count-based models on various linguistic tasks (Baroni et al., 2014; Levy et al., 2015). The neural network based models learn embeddings for words from predicting the context of a target word. The intuition is that words with similar meanings often appear in similar context. Therefore, the neural models learn a word embedding by starting from a random vector and iteratively shifting the embedding to be more like the embeddings of the words that occur in the similar contexts. Embeddings pre-trained with neural models on large corpora can give good performance out-of-the-box on many tasks. In particular, they prove to be very effective on standard semantic similarity datasets (Mikolov et al., 2013a; Pennington et al., 2014; Hill et al., 2015).

We consider two state-of-the-art neural network based embedding models to measure word similarity and sentence similarity between the candidate summary and the reference.

1. Word similarity

Word2vec (Mikolov et al., 2013a) is a neural network based model for learning vector representation of words from a large corpus of texts. There are two variants of word2vec architectures: skip-gram and CBOW (continuous bag of words). The skip-gram variant learns word embeddings from predicting the context of a given word, whereas the CBOW variant is trained to predict the target word given some context words. We make use of the CBOW variant in our experiments. The CBOW model architecture is shown in Figure 5.3.

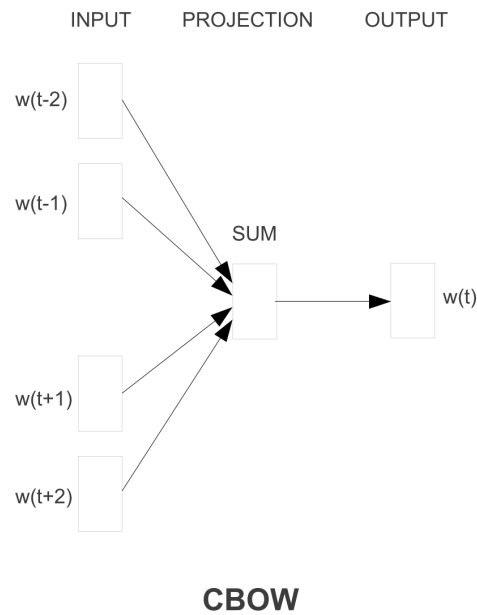


Figure 5.3: The CBOW model architecture (Mikolov et al., 2013a)

The CBOW model maps each word to a dense vector of a fixed dimensionality. The dimensionality of the vector is a parameter of the system, and it is usually set to a few hundreds. The model consists of input, projection and output layers. At the input layer, context words from both the previous and the following context are encoded with one-hot encoding and projected onto a projection layer using a shared projection matrix. The projected vectors are summed up and normalized to predict the target word at the output layer. The output layer corresponds to a probability distribution of all words in the vocabulary and the word with the highest probability is considered to be the prediction. The CBOW model is called a bag-of-words model because the order of the words does not influence the prediction. The projection layer and the output layer are connected with a word embedding matrix, where each row of the matrix corresponds to the embedding of a word in the one-hot encoding. The projection matrix and the word embedding matrix are randomly initialized. As the model is trained to maximize the classification of the target word, the word embeddings gradually become more similar to the embeddings of words in similar contexts, and less similar to the embeddings of words that do not appear in the same context.

In the experiments, we use embeddings pretrained on Wikipedia to compute word-to-word similarity between the candidate summary and the reference. The embeddings are learnt from 3.5 billion words of encyclopedic text and contain 300-dimensional vectors for approximately 750,000 unique words and phrases.

We use the cosine similarity between the word vectors to measure the semantic similarity between two words. For example, given two word vectors w_1 and w_2 , the cosine similarity

between the word vectors is defined as:

$$\text{cosine similarity}(w_1, w_2) = \frac{w_1 \cdot w_2}{|w_1||w_2|} \quad (5.6)$$

, where $w_1 \cdot w_2$ is the dot product of w_1 and w_2 , $|w|$ is the length of the vector.

Cosine similarity measures the cosine of the angle between two vectors, and it is the most popular metric to measure the similarity of two word vectors. Many other popular vector distance metrics have been proposed to measure similarity, such as Euclidean distance, Jaccard distance, and Kullback-Leibler divergence. However, the cosine similarity metric is usually preferred in NLP tasks involving word similarity, as previous studies showed that cosine similarity outperformed other similarity metrics on these tasks (Bullinaria and Levy, 2007; Huang, 2008). In addition, cosine similarity is irrelevant to the length of the word embeddings. Although not as apparent as in count-based models, the length of word embeddings learnt with neural models is also affected by the frequency of the words (Schakel and Wilson, 2015). Therefore, using cosine similarity can help ameliorating the influence of the word frequency in measuring the similarity of two words.

2. Sentence similarity

Skip-thought (Kiros et al., 2015) is a model for learning vector representations of sentences. The skip-thought model implements an encoder-decoder structure. An encoder maps a sentence to a vector, and a decoder conditions on the vector to generate surrounding sentences. Figure 5.4 shows the architecture of the skip-thought model.

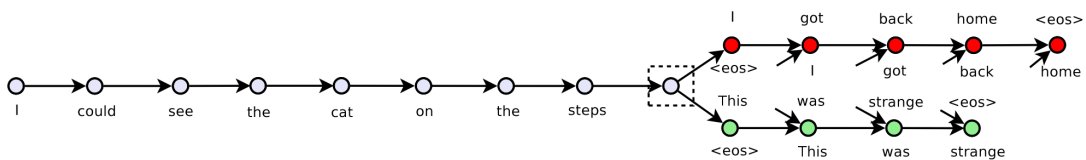


Figure 5.4: The skip-thought model architecture (Kiros et al., 2015)

In the skip-thought model, a sentence is an input to a RNN encoder with Gated Recurrent Unit (GRU) (Cho et al., 2014) units to compose a sentence vector, where words in the input sentence are represented with pretrained word2vec embeddings and are fed sequentially to the encoder. The resulting sentence vector is then passed on to a RNN-decoder to predict the previous sentence in the context and passed on to a second decoder to predict the next sentence. The decoder predicts one word at a time from the probability distribution of all words in the vocabulary. The model is trained as a single network to optimize the sum of the log probabilities of the words in the surrounding sentences conditioned on the encoder representation and previously predicted words. Depending on the encoder, there are two

versions of skip-thought models: a uni-skip model using uni-directional RNN-encoder and a bi-skip model using bi-directional RNN-encoder with the forward and backward encoders.

Once trained, the encoder of the skip-thought model can be used as a generic feature extractor to generate sentence embeddings for unseen sequences. Kiros et al. (2015) showed that the sentence vectors extracted with pretrained skip-thought model are robust and effective on many NLP tasks from semantic relatedness to paraphrase detection to image caption ranking.

We use the model pretrained on the BookCorpus (Zhu et al., 2015) to generate sentence vectors for the candidate summary and the reference.⁴ The BookCorpus is a dataset containing 985 million words of unpublished novels covering a wide range of genre and style. The resulting sentence vectors are 4,800 dimensional, with the first 2,400 dimensions obtained from the uni-skip model and the last 2,400 from the bi-skip model. We use the combined skip vectors because they are the best performing sentence embeddings as reported in Kiros et al. (2015).

Average word embeddings: In addition to the skip-thought model, we experiment with composing the sentence vectors from averaging the word2vec embeddings. Taking the average of word vectors is a frequently used approach to represent a word sequence as a vector. Despite its simplicity, it has proven to be very useful in various applications (Mitchell and Lapata, 2010; Zanzotto et al., 2010). In particular, recent studies suggest that averaging neural network based word embeddings is an effective approach to representing sentence meaning and it has been shown to be a strong baseline in many semantic related tasks (Faruqui et al., 2014; Yu et al., 2014; Gershman and Tenenbaum, 2015; Kenter and De Rijke, 2015; Kenter et al., 2016). Therefore, we use the average of the word2vec embeddings derived previously as one of the approaches in our experiments to obtain sentence representations for the candidate summary and the reference.

Similar to word similarity, we use cosine similarity between the sentence vectors to capture sentence-to-sentence similarity between the candidate summary and the reference.

Once the word-to-word (or sentence-to-sentence) similarity scores are obtained, we combine the similarity scores of the text segments into a semantic similarity measure to compare the content of the candidate summary and the reference at the text level.

Taking the word similarity scores as an example, we explore three scoring functions to construct the text-level semantic similarity measures from the word-to-word similarity scores. Formally, let w_c^1, \dots, w_c^m be the words in the candidate summary C and w_r^1, \dots, w_r^n be the words in the reference R , where m is the length of the candidate summary and n is the length of the

⁴<https://github.com/ryankiros/skip-thoughts>

reference. We compute the following scoring functions to measure the semantic similarity of the two texts $sim(C, R)$:

- (i) *average word similarity*: We calculate the cosine similarity between every word pair $sim(w_i, w_j)$ in the candidate summary and the reference, and take the average of word similarity scores to measure the text similarity.

$$sim(C, R) = \frac{\sum_{w_i \in C} \sum_{w_j \in R} sim(w_i, w_j)}{m \cdot n} \quad (5.7)$$

- (ii) *greedy matching*: In this approach, we use a greedy method to pair up the words that are found to be the most similar to each other and take the average similarity of the word pairs. We use a scoring function similar to the one introduced by Mihalcea et al. (2006) to compute the text similarity score from the word-to-word similarities between the greedily paired up words. In their work, they defined a directional similarity score $sim'(T_1, T_2)$ to describe the semantic similarity of a text T_1 with respect to a text T_2 . For every word in T_1 , they identify a word in T_2 that has the highest semantic similarity. The directional similarity score is then determined as:

$$sim'(T_1, T_2) = \frac{\sum_{w_i \in T_1} \max_{w_j \in T_2} sim(w_i, w_j)}{len(T_1)} \quad (5.8)$$

where the similarity scores between the greedily obtained word pairs are added up and normalized by the length of T_1 . The directional similarity score is asymmetric: $sim'(T_1, T_2) \neq sim'(T_2, T_1)$.

Following this, we obtain the directional similarity scores for $sim'(C, R)$ and $sim'(R, C)$:

$$sim'(C, R) = \frac{\sum_{w_i \in C} \max_{w_j \in R} sim(w_i, w_j)}{m} \quad (5.9)$$

$$sim'(R, C) = \frac{\sum_{w_j \in R} \max_{w_i \in C} sim(w_i, w_j)}{n} \quad (5.10)$$

The scores from both directions are combined into a bidirectional similarity score using a simple average function:

$$sim(C, R) = \frac{sim'(C, R) + sim'(R, C)}{2} \quad (5.11)$$

- (iii) *optimal matching*: Rus and Lintean (2012) designed an optimal matching approach to assess the similarity of two texts based on word-to-word similarity scores. The optimal matching approach creates an alignment between the words in a pair of texts where every word in each text is matched to at most one word in the other text, and it finds an optimal alignment such that the sum of the word similarities over the alignment is maximized.

For example, Figure 5.5 illustrates the word-to-word similarity scores between two text segments “motion acceleration” and “speed motion”. A greedy matching approach will match both “motion” and “acceleration” in T_1 to “motion” in T_2 when measuring the similarity of T_1 with respect to T_2 , and match “speed” and “motion” in T_2 to “motion” in T_1 when comparing T_2 to T_1 . The optimal matching approach, on the other hand, would pair “motion” in T_1 with “speed” in T_2 and “acceleration” in T_1 with “motion” in T_2 as this alignment yields an overall similarity of 1.70, whereas the greedy approach gives an estimate of 1.85.

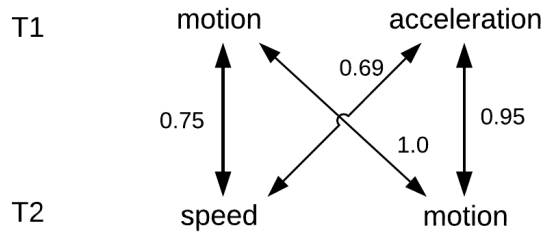


Figure 5.5: Example of two text segments with word-to-word similarity scores between each word pair

We formulate the optimal alignment of words as a linear assignment problem. The linear assignment problem is one of the fundamental combinatorial optimization problems of finding a maximum weight matching in a weighted bipartite graph. Formally, given two sets A and B of equal size, together with a weight function $W : A \times B \rightarrow R$, the problem consists of finding a bijection function (i.e. the optimal matching) $f : A \rightarrow B$ such that the objective function $\sum_{a \in A} W(a, f(a))$ is maximized. The linear assignment problem can be solved with the Hungarian algorithm (Kuhn, 1955) in polynomial time.

To convert the task of aligning the words in two texts into a linear assignment problem, we add dummy word tokens to the shorter text to make sure that the lengths of the two texts are equal. The similarity between an actual word and the dummy token is set to 0. Words in the candidate summary and the reference are regarded as elements of the two sets, and the similarity scores between the words form the weight function.

After obtaining the optimal alignment of words between the candidate summary and the reference, the sum of word similarities over the alignment is computed and normalized with the length of the candidate summary. The final score is used to represent the text-level similarity between the candidate summary and the reference:

$$sim(C, R) = \frac{\sum_{w_i \in C} sim(w_i, optimal-align(w_i))}{m} \quad (5.12)$$

Similarly, the three scoring functions are applied to sentence embeddings to combine the sentence-to-sentence similarity scores into a text-level semantic similarity measure between the candidate summary and the reference.

5.4.3 Distributed vector representations of the summary

In addition to the word and sentence similarities, we investigate methods to model the content similarity between the candidate summary and the reference directly at the document level. Vector representations of the candidate summary and the reference are created and compared to assess the similarity of the two texts. Similar to word similarity and sentence similarity, we take the cosine similarity between the document vectors to estimate the similarity between the candidate summary and the reference.

In our experiments, we use the Simple English Wikipedia corpus,⁵ which contains 30 million words from more than 130k documents covering a diverse range of topics, as our background resource to learn the document representations. The Simple English Wikipedia data is used to train the models because the documents in this corpus are rendered simple for English learners. Therefore the lexical usage and syntactic structure in Simple English Wikipedia is more similar to the summaries written by learners.

Specifically, we use the following five approaches to construct vector representations of the learner summaries:

TF-IDF is a standard weighting scheme in information retrieval to construct document representations using term-document matrices. In a term-document matrix, each row represents a word in the vocabulary and each column represents a document from a collection of documents. Each cell in this matrix represents the number of times a particular word occurs in a particular document, a.k.a term frequency (TF). In this model, each document is represented with a column vector in the term-document matrix. Beside term frequency, Sparck Jones (1972) observed that terms that are limited to a few documents are more useful than terms that occur frequently across the entire collection for discriminating those documents from the rest of the collection. Therefore, inverse document frequency (IDF) is introduced to assign higher weights to words that occur only in a few documents. We use the following formula to calculate the IDF for word i :

$$idf_i = \log\left(\frac{N}{df_i}\right) \quad (5.13)$$

where N is the total number of documents, df_i is the number of documents in which word i occurs. Combining term frequency tf_{ij} with IDF results in the TF-IDF weighting scheme for word i in document j , w_{ij} :

$$w_{ij} = tf_{ij} \cdot idf_i \quad (5.14)$$

⁵<https://simple.wikipedia.org>

TF-IDF weighted document vectors are frequently used for measuring query-document similarity in information retrieval. In our task, we use the TF-IDF weighted document vectors to measure the similarity of the candidate summary and the reference. The IDF weighting is learnt from the background corpus.

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) applies singular value decomposition (SVD) on the term-document matrix to obtain dense vector space representation of the documents. LSA has been used in document classification, information retrieval and word relatedness tasks and showed good results (Landauer, 2006).

$$\begin{array}{ccccccc}
 & & & & & & \text{vector representation for document } j \\
 & & & & & & \begin{array}{c} 1, 2, \dots, j, \dots, n \\ \hline \end{array} \\
 \left(\begin{array}{c} M \\ \end{array} \right) & \approx & \left(\begin{array}{c} U^* \\ \end{array} \right) & \left(\begin{array}{c} \Sigma^* \\ \end{array} \right) & \left(\begin{array}{c} V^* \\ \end{array} \right) \\
 m \times n & & m \times k & k \times k & k \times n
 \end{array}$$

Figure 5.6: An illustration of LSA

SVD factorizes the term-document matrix M of size $m \times n$ into the product of three matrices: U , Σ , and V , where m is the size of the vocabulary and n is the total number of documents. U is a $m \times r$ matrix, where each row represents a word and each column represents one of r dimensions in a latent space. The number of the dimensions r is the rank of the term-document matrix M . Σ is a diagonal $r \times r$ matrix, with singular values along the diagonal indicating the importance of each latent dimension. V is a $r \times n$ matrix, with each row representing one of the latent dimensions and each column representing a document. Dimensionality reduction can be applied to the latent space by using only the first k dimensions out of all r dimensions, resulting in a truncation in the matrices U^* , Σ^* , and V^* . The product of the truncated U^* , Σ^* , and V^* becomes a least-squares approximation to the original matrix M . Each column of the truncated matrix V^* becomes a dense k dimensional vector representing a document. Figure 5.6 is an illustration of LSA.

We set the dimensionality of the LSA document vectors to 100, 200, and 400 and train our models on Simple English Wikipedia with TF-IDF weightings.

Latent Dirichlet Allocation (LDA) (Blei et al., 2003) represents the documents as mixtures of topics. It can be used to measure the content similarity and topical relevance of the documents.

LDA is a generative probabilistic model. It assumes that documents are represented as mixtures over latent topics, where each topic is characterized by a distribution over words. The graphical representation of LDA is show in Figure 5.7. In this model, a document is a sequence of N words denoted by $\mathbf{d} = (w_1, w_2, \dots, w_N)$, and a corpus is a collection of M documents

Algorithm 1: The generative process of LDA

- 1: Choose θ_i from $Dir(\alpha)$, where $i \in \{1, \dots, M\}$, θ_i is the topic distribution for document d_i , $Dir(\alpha)$ is a Dirichlet distribution with a parameter α
 - 2: For each topic k , choose ϕ_k from $Dir(\beta)$, where $k \in \{1, \dots, K\}$, ϕ_k is the word distribution for topic k
 - 3: For each word w_n in the document i :
 - (a) Choose a topic $z_{i,n}$ from $Multinomial(\theta_i)$ (a multinomial distribution)
 - (b) Choose a word $w_{i,n}$ from $Multinomial(\phi_{z_{i,n}})$
-

denoted by $D = \{d_1, d_2, \dots, d_M\}$. Let K denote the number of topics. The generative process of LDA is described in Algorithm 1.

Learning the various distributions (the set of topics, their associated word probabilities, the topic for each word, and the topic mixture for each document) is a problem of Bayesian inference. Details of computing the posterior distribution can be found in Deerwester et al. (1990) and are not discussed in this thesis.

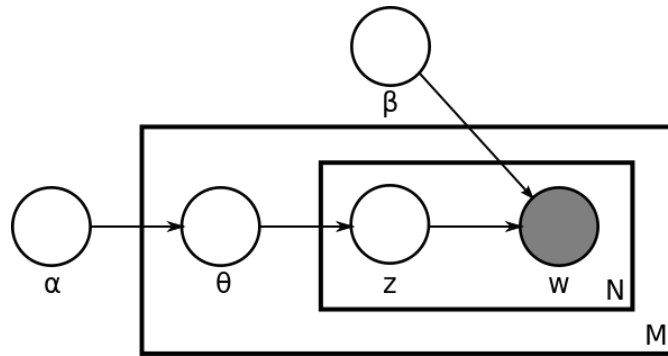


Figure 5.7: Graphical model representation of LDA (Blei et al., 2003)

We set the topic number to 100, 200 and 400 and use LDA to learn the topic representation of the summary.

Doc2Vec (Le and Mikolov, 2014) is a neural network model derived from the word2vec model for learning vector representation of documents. Similar to the word2vec model, there are two variants of doc2vec: the distributed memory model and the distributed bag of words (DBOW) model. Similar to the skip-gram model, DBOW is trained by predicting context words given a document vector, and similar to CBOW, the distributed memory model is trained to predict a target word in a document given the document vector and its contexts words. We make use of the distributed memory model to construct our vector representation of the summary, as it performs better than the DBOW model according to (Le and Mikolov, 2014). Figure 5.8 shows the architecture of the distributed memory model.

The distributed memory model maps every document to a unique vector. A fixed length

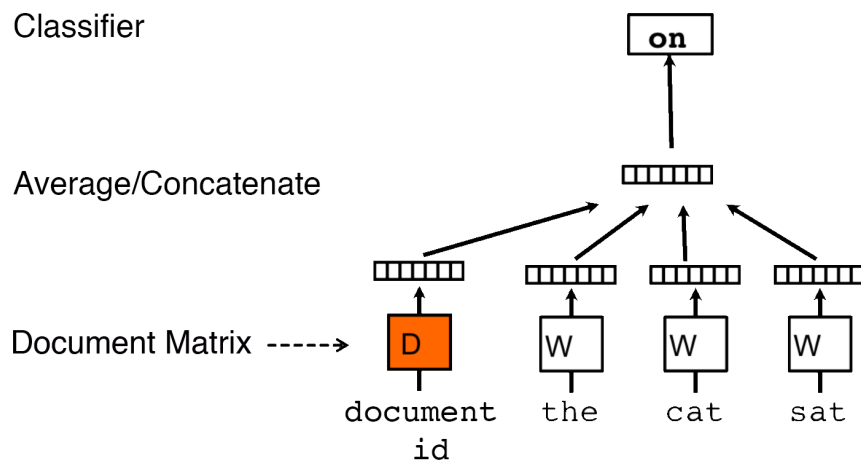


Figure 5.8: The distributed memory model architecture (Le and Mikolov, 2014)

context is sampled from the document. The document vector and word vectors are averaged or concatenated to predict the next word in the context. Similar to the CBOW model, the output is a softmax layer for classification. The document vector and the word vectors are trained using stochastic gradient descent to maximize the classification of the target word. The document vectors for texts that have not been seen before are also obtained via gradient descent. In this “inference” process, parameters for the rest of the model (e.g. the word vectors and the softmax weights) are fixed, and the model is trained to predict words in the new document given the document and the previous contexts. After the model is trained, the document vector can be used as a feature to represent the text in other tasks.

We use pretrained word2vec embeddings obtained previously to initialize the word vectors in this model. We train the model on Simple English Wikipedia and obtain document vectors of 100, 200 and 400 dimensions.

Average word embeddings: We also make use of the average word2vec embeddings to encode the summary. To create a vector for the document, each word in the document is mapped to a vector, and we take the average of the word vectors of all words to represent that document.

We make use of the Python *gensim* package⁶ to compute the TF-IDF, LSA, LDA and Doc2Vec document vectors. Having obtained the vector representations of the candidate summary and the reference, we take the cosine similarity between the vectors to estimate the similarity of the texts.

5.4.4 Discourse and other textual features

Apart from the content-based measures of the summary, the textual quality of the summary is also important for its overall quality estimation. For instance, good summaries tend to be more coherent and logically consistent. We extract lexical chain based discourse measures to assess

⁶<https://radimrehurek.com/gensim/>

the coherence of the text.

Lexical chains model the semantic relations among entities throughout the text. We implement the lexical chaining algorithm developed by Galley and McKeown (Galley and McKeown, 2003) and extract 7 lexical chain-based features to model the entity-based coherence of the summary. We use the same features as introduced in Section 3.3.5.

We also measure the following superficial textual features:

- **Length:** Number of words in the summary.
- **Compression ratio:** The ratio of the number of words in the summary to the number of words in the reading passage.
- **Type-token ratio:** The ratio of the number of unique words to the total number of words in the summary.
- **Text readability:** The reading difficulty of the passage to be summarized in the CEFR level.

We believe that the length and the compression ratio of the summary are also important for assessing the summary quality, as short summaries can be excessively abstract or lack important information, and long summaries are likely to be diffuse and contain irrelevant information. Although we have set a word limit for the summarisation task, learners sometimes fail to adhere to the word limit in writing the summary. Therefore the length-related features can be used to identify summaries deviating from the length limit. Type-token ratio is a simple indicator of lexical variation in the summary. It is useful in recognizing repeated use of words and sentences in a summary. The reading difficulty of the passage to be summarized also affects the summary quality. A text that is more readable is usually easier to summarize.

5.4.5 Models for assessing summary

To automatically assess the summary quality, machine learning models are used to combine and map the features to a score indicating the quality of the summary. Depending on the annotation information we have on the data, different machine learning frameworks are implemented. We train a binary classification model on the simulated learner data to predict whether the summary is good or bad, and a regression model on the real learner data to predict the summary score on the scale of 0 to 5, because the simulated data is a binary dataset with summaries of two types of quality written by proficient English users, and the real learner data contain real learner summaries of different quality.

In a preliminary experiment, we compared several popular machine learning models for the classification task and the regression task on the summary data using a combination of all features. We found that the SVM classifier performs the best for the classification task, compared

to Multilayer Perceptron (Gardner and Dorling, 1998), Logistic Regression, and Random Forest (Liaw and Wiener, 2002) classifiers. The best model for the regression task is the Kernel Ridge Regression model, which outperforms linear regression, Support Vector Regression (Drucker et al., 1997), Multilayer Perceptron Regression (Gardner and Dorling, 1998), Gaussian Process (Rasmussen, 2004) and Random Forest Regression (Liaw and Wiener, 2002) models. As a consequence, the SVM classifier and the Kernel Ridge Regression model are selected for our experiments. The SVM classifier was introduced in Section 3.4.1.1.

Kernel Ridge Regression (KRR) is a special case of Support Vector Regression (SVR). Given some training data (\mathbf{x}_i, y_i) , where \mathbf{x}_i denotes the feature vector and scalar y_i denotes the target value, the goal of the usual support vector regression model is to find a function $f(\mathbf{x})$ that has at most ϵ deviation from the actually obtained target y_i for all the training data, and at the same time is as flat as possible. Taking a linear function for an example, the linear function f takes the form:

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b \quad (5.15)$$

, where w and b are the weights and bias term.

Similar the SVM objective, the optimization problem for SVR can be written as:

$$\begin{aligned} & \text{minimize} \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N (\xi_i + \xi_i^*), \\ & \text{subject to} \quad \begin{cases} y_i - \mathbf{w} \cdot \mathbf{x}_i - b \leq \epsilon + \xi_i \\ \mathbf{w} \cdot \mathbf{x}_i + b - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \\ & \text{for} \quad i = 1, \dots, N \end{aligned} \quad (5.16)$$

where ξ_i, ξ_i^* are the slack variables and N is the size of the training set.

KRR simplifies the problem by ignoring the bias term b , setting $\epsilon = 0$, and using squared error loss. The optimization problem becomes:

$$\text{minimize} \quad a \|\mathbf{w}\|^2 + \sum_{i=1}^N (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2 \quad (5.17)$$

where $a = 1/C$. The kernel trick replaces \mathbf{x} with $F(\mathbf{x})$, where the kernel function $F : \mathbf{x} \rightarrow k$ maps the original feature vector to a transformed feature space.

In contrast to SVR, KRR has closed-form solution and is typically faster than SVR on smaller

datasets. The prediction of a new data point \mathbf{x} can be written in a vector form as:

$$\hat{y} = \hat{\mathbf{w}} \cdot \mathbf{x} = \mathbf{Y}(\mathbf{K} + a\mathbf{I})^{-1}\mathbf{k} \quad (5.18)$$

where $\mathbf{Y} = [y_1, \dots, y_N]$, \mathbf{K} is the $N \times N$ Gram matrix in which $\mathbf{K}_{i,j} = \mathbf{x}_i \cdot \mathbf{x}_j$, $\mathbf{k} = [\mathbf{x}_1 \cdot \mathbf{x}, \dots, \mathbf{x}_N \cdot \mathbf{x}]^T$, and \mathbf{I} is the identity matrix. When the kernel trick is applied, the element of \mathbf{K} becomes $\mathbf{K}_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, where κ is the kernel $\kappa(\mathbf{x}, \mathbf{x}') = F(\mathbf{x}) \cdot F(\mathbf{x}')$.

We compared using the linear function, the polynomial function and the radial basis function as the kernel function, and found that both the classification model and the regression model perform the best with the linear kernel. Hence, we use linear kernel SVM to assess the binary quality of the summary on the simulated data and linear kernel KRR to predict the summary score on the learner data. We make use of the Python *scikit-learn* package⁷ to implement both models.

5.5 Method 2: Assessing summaries with a similarity matrix

In the second approach, we construct a sentence similarity matrix between the candidate summary and the original reading passage and apply a Convolutional Neural Network (CNN) model on the similarity matrix to predict the quality of the summary.

5.5.1 Motivation and method

The motivation behind this approach originates from our definition of a good summary. One of the criteria by which we judge a summary is as follows: a good summary is supposed to comprise a comprehensive coverage of the information presented in the original text, whereas a bad summary might be partial or less relevant to the original text. But how does this criterion translate to a computational model?

Lemaire et al. (2005) proposed several computational cognitive models for assessing extractive summarisation. In their experiments, they prepared two reading passages and asked 278 native American school students (from grade 8 to grade 11) to underline three to five sentences that they think are the most important in each of the two texts. The underlined sentences were then compared against the set of all the sentences from the original passage using cosine similarity of vector representations of the sentences constructed with LSA. They observed that the important sentences selected by the students are highly connected to the rest of the sentences in the text, where the connection is defined by the semantic similarity of the sentences.

From their observation, we hypothesize that sentences in a good summary should have a well-distributed connection with as many sentences as possible in the original text, because a good summary is supposed to cover all the important information in the text. In contrast,

⁷<https://scikit-learn.org/>

sentences in a bad summary may fail to form a well-distributed connection with sentences in the original text. For example, if a bad summary only captures a few of the main points in the original text, then the sentences in such bad summary would be connected only to the sentences where these points are mentioned in the original text, while lacking the connections to the rest of the text. If a bad summary is generally irrelevant to the original text, sentences in such bad summary would be minimally connected to most of the sentences in the original text. Beside these extreme cases of summary quality, summaries of intermediate quality may display patterns of connection to the original passage that share the characteristics of the good summary and the bad summary to various degrees.

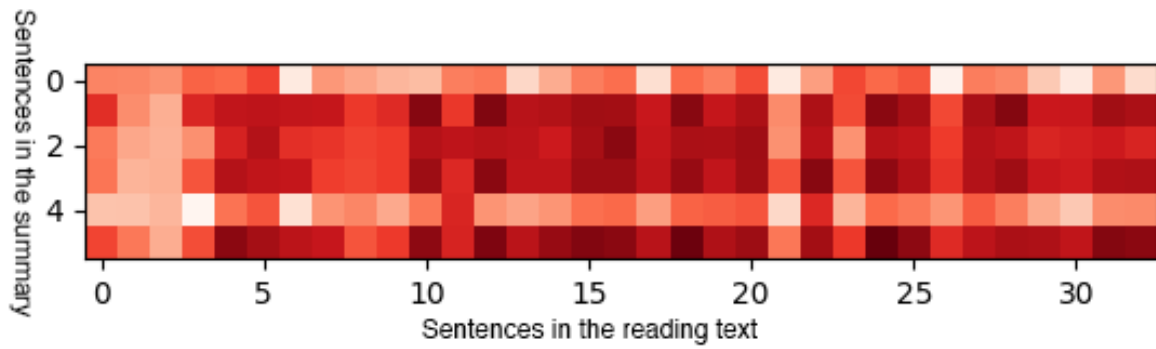
Following this idea, we construct a sentence similarity matrix between the candidate summary and the original text. Each element of the matrix is a cosine similarity score between the vector representations of a sentence from the summary and a sentence from the original text. We use the two sentence similarity models described in section 5.4.2, skip-thought and average word embeddings, to build the sentence vectors.

From our hypothesis, the quality of the summary corresponds to different patterns in the similarity matrix. For example, a good summary has higher sentence-to-sentence similarity scores on the average and its matrix is more evenly distributed as the summary sentences connect to more sentences in the original text. A bad summary might have high similarity scores with several specific sentences in the original text, while the scores might be low for all other sentences.

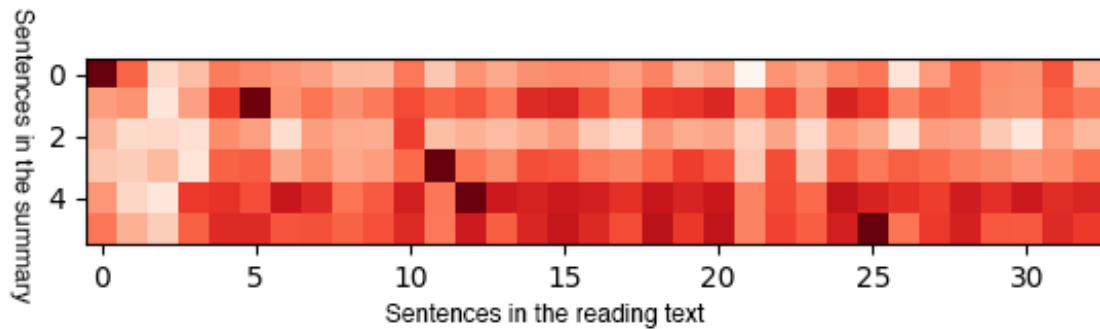
The similarity matrix can be viewed as a heat map “image” from which we can learn patterns to detect the quality of the summary. Figure 5.9 demonstrates the similarity matrices of two summaries for the same reading passage from the simulated data.⁸ The rows of the matrix represent sentences in the summary and the columns of the matrix represent sentences in the reading passage. The shade of the coloured map indicates the degree of similarity between two sentences: the darker the shade is, the more similar the sentences are. In this example, Summary A is an example of a good summary, and Summary B is an example of a bad summary. Both summaries are made up of six sentences. We can see that sentences in Summary A are similar to a number of sentences in the original text, resulting in a well-distributed heat map. By contrast, sentences in Summary B are similar to five particular sentences in the text and are less similar to other sentences, which is reflected by the isolated dark patches in the image. On the whole, Summary A has higher similarity scores than Summary B, which makes its heat map darker. These two examples illustrate that different patterns can be observed in the heat map of the summaries of different quality.

To learn these patterns automatically, we apply a CNN model on the similarity matrix to predict the quality of the summary. CNNs are widely used in the computer vision field to learn patterns from two dimensional image matrices and they have been proved to be very useful in

⁸The two summaries and the original reading text are given in Appendix D and Appendix C.



(a) The similarity matrix of Summary A



(b) The similarity matrix of Summary B

Figure 5.9: Similarity matrices of two summaries for the same reading passage from the simulated data. Summary A is a good summary and Summary B is a bad summary.

tasks such as object classification (Krizhevsky et al., 2012; Szegedy et al., 2015), face recognition (Lawrence et al., 1997; Schroff et al., 2015), etc. Similarly, we use a CNN to learn the patterns in the similarity matrix for assessing the quality of the summary.

5.5.2 The convolutional neural network model

A CNN is a neural network model made up of convolutional layers, pooling layers and fully connected layers. The structure of a typical CNN model for image processing is shown in Figure 5.10.

The convolutional layer consist of a set of learnable filters that perform convolution on the input image matrix. Each filter is a small matrix that slides across the whole input image and computes dot products between the elements of the filter and the input at any position. During the training process, the CNN automatically learns the values of the filters. Intuitively, the network is trained to learn filters that activate when they see specific patterns in the image. A convolutional layer is usually made up of a number of filters, where different filters can capture different patterns in the image. By convolving the filter with the input image, each filter maps the local patches of lower-level features in the input into a higher-level representation. The network combines the representations from all filters for prediction. Convolutional layers can be stacked

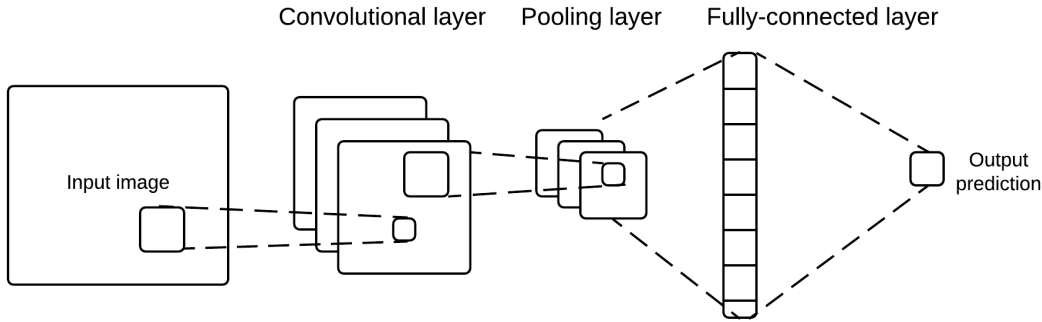


Figure 5.10: The Convolutional Neural Network model for image processing

to learn higher-level features from the output of previous convolutional layers. A non-linear activation function is usually applied element-wise on the output of the convolutional layer. The most commonly used activation function for CNNs is the ReLu function: $f(x) = \max(0, x)$.

Pooling layers are frequently used in convolutional neural networks to summarize the activities of local patches in the convolutional layer, so as to reduce the amount of features and the computational cost of the model and also prevent the model from over-fitting (Hinton et al., 2012a). There are several variants of pooling layers, including max-pooling and average-pooling. Max-pooling selects the largest element from a pre-defined window size in the feature map. Average-pooling takes the average of the values in the local patch. Both max-pooling and average-pooling may perform well depending on the data and the task (Boureau et al., 2010).

Dropout is a technique frequently used in training neural network models. The term “dropout” refers to randomly removing units temporarily from the neural network representation. Dropout can prevent the neural networks from over-fitting and it gives improvements on many benchmark tasks (Srivastava et al., 2014; Hinton et al., 2012a).

A fully connected layer connects every neuron in one layer to every neuron in another layer. It is in principle the same as a regular neural network layer. When the model is used for classification, the output from the fully connected layer is usually fed to a Softmax function, defined as follow:

$$\sigma(\mathbf{x})_j = \frac{e^{x_j}}{\sum_{k=1}^K e^{x_k}} \quad \text{for } j = 1, \dots, K \quad (5.19)$$

The output of the Softmax function represents the categorical probability distribution for the target classes, and the class with the highest probability is predicted by the model. The model is usually trained to minimize the cross-entropy loss between the target classes and predicted classes. Cross-entropy is a function between two probability distributions p and q that measures the number of bits needed to encode data from the distribution p using the distribution q in

information theory:

$$H(p, q) = - \sum_x p(x) \log q(x) \quad (5.20)$$

The cross entropy loss function for the model is computed by taking the average of all cross entropies in the training sample. In binary classification, the cross entropy loss function can be calculated as:

$$L(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{N} \sum_{n=1}^N [y_n \log \hat{y}_n + (1 - y_n) \log (1 - \hat{y}_n)] \quad (5.21)$$

where N is the number of samples, y_n and \hat{y}_n are the gold-standard label and the prediction for sample n respectively, $\mathbf{y} = [y_1, \dots, y_n]$, and $\hat{\mathbf{y}} = [\hat{y}_1, \dots, \hat{y}_n]$.

When the model is used for regression, the output of from the fully connected layer is set to a single value as the final prediction. The square loss function is commonly used for training regression models. Given N training samples with label set \mathbf{y} and prediction set $\hat{\mathbf{y}}$, the square loss is written as:

$$L(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{n=1}^N (y_n - \hat{y}_n)^2 \quad (5.22)$$

5.5.3 CNN model for assessing the summary quality

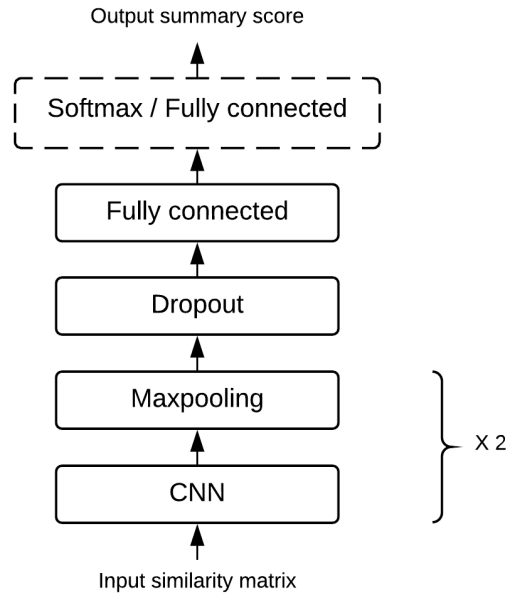


Figure 5.11: The model architecture for assessing summary with a similarity matrix

To assess the learner summary using the similarity matrices we constructed, we build a small network with CNN components to predict the summary score. We train a classification model to predict the summary quality on the simulated data and a regression model to predict the summary score on the real learner data. Figure 5.11 shows the model architecture. We make use of the Python *Keras*⁹ library to implement the CNN model.

The similarity matrices are fed into two layers of convolutional networks (with the Relu activation function) followed by a fully connected layer. We use zero-padding to resize the similarity matrices into the same shape. Zero-padding refers to adding zeros to the matrices to increase their size. Max-pooling is applied after each convolutional layer and dropout is applied before passing the vectors to the fully connected layer. The last layer is a Softmax layer for the classification model, and a fully connected layer with a single output value for the regression model. In the training phase, we optimize cross-entropy for classification and optimize mean squared error for regression. The model is trained with gradient descent to learn the patterns in the similarity matrix.

However, it should be noted that CNNs usually work the best when a large amount of training data is available, but the summary data we have collected represents a relatively small dataset. We compare the results of the CNN model against the feature extraction approach to investigate how well the model can learn from the limited amount of data.

5.6 Method 3: LSTM-based models for summary assessment

In this section, we describe several LSTM-based neural network models for assessing the summary quality. The LSTM-based models are used to learn representations of the summary and estimate its quality automatically, without having to manually extract features from it.

5.6.1 Motivation

Neural network models are widely used in NLP tasks and have proved to be highly successful in many applications such as text classification (Kim, 2014), automated essay scoring (Taghipour and Ng, 2016) and machine comprehension (Hermann et al., 2015). Compared to traditional methods which rely heavily on laborious feature extraction, neural approaches have shown considerable ability to model complex patterns in the data and can be used to solve problems in an end-to-end fashion. A neural approach can encode the required information to evaluate the summary quality without having to manually extract carefully designed features.

Specifically, recurrent neural networks with LSTM units have been very successful in machine translation (Bahdanau et al., 2014), question answering (Wang and Jiang, 2016), textual entailment (Rocktaschel et al., 2015), and many other NLP tasks. LSTMs are capable of

⁹<https://keras.io/>

embedding long sequences of text into a vector representation which can later be decoded for use in many applications.

For example, Taghipour and Ng (2016) built an automated essay scoring system by encoding student essays with an RNN model with LSTM units. They compared the LSTM based model with a publicly available feature-based essay scoring system and found that the LSTM model significantly outperformed the feature based model. Their work showed that RNNs can be trained to capture the necessary information in the embedded vector representation of the student essay, without requiring any feature engineering.

Inspired by their work, we use RNNs to encode the summary and the reading text separately, and use a combination of the embedded summary and embedded reading text to predict the summary quality.

5.6.2 Recurrent neural networks

We use RNNs to learn representations of the summary and the original reading passage. An RNN is a neural network model with recurrent hidden states that makes use of sequential information. Figure 5.12 demonstrates a typical RNN model.

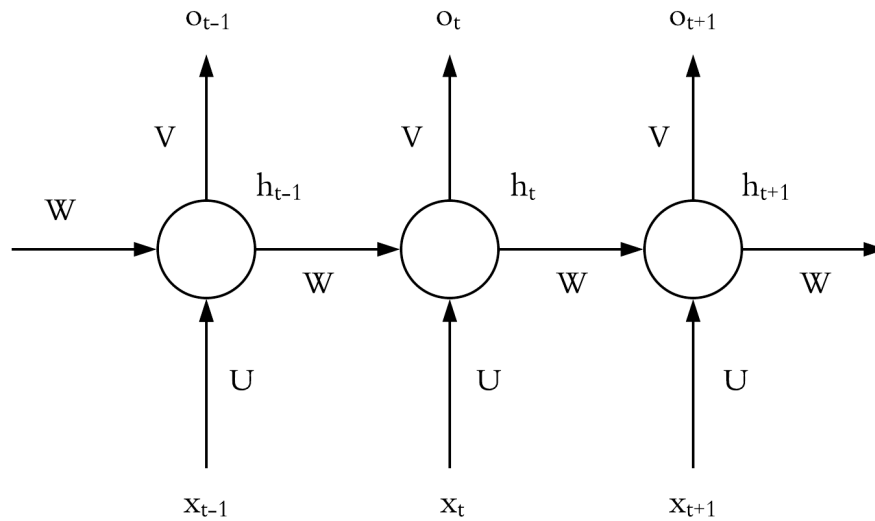


Figure 5.12: A recurrent neural network model

At each time step t , the recurrent hidden state h_t is calculated from the input at the current step x_t and the previous hidden state h_{t-1} :

$$h_t = f(Ux_t + Wh_{t-1} + b) \tag{5.23}$$

where f is usually a nonlinear activation function such as ReLu or tanh, W and U are weight

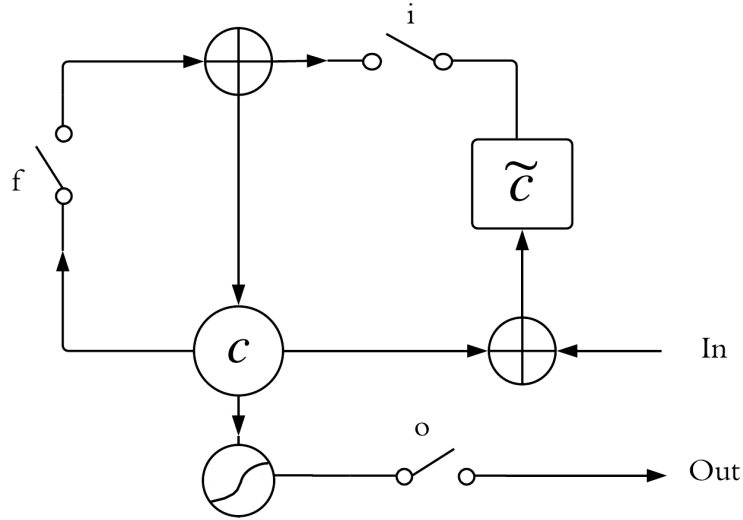


Figure 5.13: An illustration of an LSTM unit

matrices, and b is an optional bias term. Unlike a traditional neural network, which uses different weight matrices at each layer, an RNN shares the same weights across all time steps. As the current hidden state in an RNN depends to some extent on the hidden states of all previous time steps, we can think of the recurrent hidden state as having a “memory” that captures information from what has been calculated so far. In addition, RNN models allow having optional output at each time step o_t .

The RNN can be used to encode a sequence of vectors $\mathbf{x} = (x_1, \dots, x_T)$ into a vector c from the computed hidden states:

$$c = q(h_1, \dots, h_T) \quad (5.24)$$

where q is a function that combines the hidden states. Typically, q is a nonlinear function that selects the last hidden state to represent the entire sequence, as theoretically it captures the information from all previous steps.

In theory, RNNs can encode information in arbitrarily long sequences, but in practice RNNs are more likely to depend heavily on the most recent input. Previous study has shown that RNNs cannot effectively handle long term dependencies due to the vanishing and exploding gradient problems (Pascanu et al., 2013). For this reason, more advanced recurrent units, such as LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Cho et al., 2014), have been proposed to better capture long term dependencies. We experimented with the basic recurrent units, LSTMs, and GRUs to identify the best choice for our task, and found that LSTMs outperformed the other two types of units. Therefore, we focus on the LSTM architecture and describe it in this section.

LSTM was introduced by Hochreiter and Schmidhuber (1997). LSTMs do not have fun-

damentally different architecture from RNNs, but they use a different function to compute the hidden state. Figure 5.13 illustrates the components inside an LSTM hidden state (aka an LSTM unit). An LSTM unit consists of a memory cell and three sets of gating units.

The LSTM defines an internal cell state c_t at the time step t and uses the gating units to regulate whether to remove or to add information to the current cell state. A forget gate f_t is a sigmoid function that controls the extent to which previous information should be dropped. An input gate i_t decides how much new information should be added from a temporary memory state \tilde{c}_t .

$$f_t = \sigma(U_f x_t + W_f h_{t-1} + b_f) \quad (5.25)$$

$$i_t = \sigma(U_i x_t + W_i h_{t-1} + b_i) \quad (5.26)$$

$$\tilde{c}_t = \tanh(U_c x_t + W_c h_{t-1} + b_c) \quad (5.27)$$

The current memory cell c_t is updated by partially removing information from the previous cell state c_{t-1} and adding the new information from \tilde{c}_t :

$$c_t = f_t c_{t-1} + i_t \tilde{c}_t \quad (5.28)$$

Finally, the hidden state h_t at the time step t is computed by multiplying the current cell state c_t with an output gate o_t that determines the degree to which the information from the cell state will output:

$$o_t = \sigma(U_o x_t + W_o h_{t-1} + b_o) \quad (5.29)$$

$$h_t = o_t \tanh(c_t) \quad (5.30)$$

Unlike the regular recurrent unit which overwrites its content at each time step, the LSTM unit uses the memory cell state and the gating mechanism to decide when to keep, forget and output information from the previous cell states. Therefore, if the LSTM units detect useful information in the input sequence at an early stage, it is more likely that this information will be carried over a long distance. Hence, potential long-distance dependencies are better captured.

Other extensions have been proposed to deal with shortcomings of the regular RNNs with respect to the long-term dependencies. For example, the bidirectional RNN model that encodes information from both the previous and the following context was proposed by Schuster and Paliwal (1997). The bidirectional RNN is formed of a forward RNN and a backward RNN. The forward RNN reads the input sequence from the first element to the last element, and the backward RNN reads the input sequence in reverse order. Two hidden states, the forward hidden

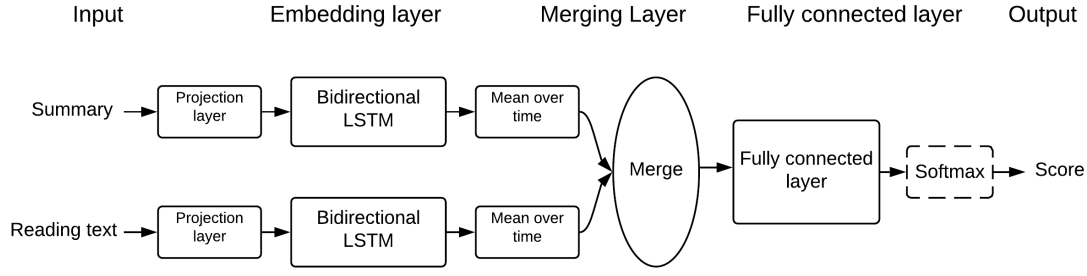


Figure 5.14: The merged LSTM model structure

state \vec{h}_t and backward hidden states \overleftarrow{h}_t , are calculated:

$$\vec{h}_t = f(\vec{U}x_t + \vec{W}\vec{h}_{t-1} + \vec{b}) \quad (5.31)$$

$$\overleftarrow{h}_t = f(\overleftarrow{U}x_t + \overleftarrow{W}\overleftarrow{h}_{t-1} + \overleftarrow{b}) \quad (5.32)$$

where \vec{U} , \vec{W} and \vec{b} are the forward weight matrices and the bias term as before, and \overleftarrow{U} , \overleftarrow{W} and \overleftarrow{b} are their backward counterparts.

The forward hidden state and the backward hidden state are combined into a vector state y_t using a nonlinear function g :

$$y_t = g(\vec{h}_t, \overleftarrow{h}_t) \quad (5.33)$$

Based on the bidirectional hidden states, a new vector representation c that encodes information from both the previous and the following context can be learnt over the input sequence $\mathbf{x} = (x_1, \dots, x_T)$:

$$c = q(y_1, \dots, y_T) \quad (5.34)$$

where q is the function as in Equation 5.24.

5.6.3 Merged LSTM model

In this section, we propose a merged LSTM model for assessing learner summaries. The merged LSTM model encodes the summary and the reading text separately and merges the embedded summary and embedded reading text representations into a joint representation to predict the summary score. The model is mainly composed of three layers: an embedding layer, a merging layer and a fully connected layer. Figure 5.14 illustrates the model architecture.

At the embedding layer, the summary and the reading text are encoded into two separate vector representations. To encode an input sequence $x = [x_1, \dots, x_T]$, it first goes through a

projection layer that maps the one-hot representation of sequence of words into a sequence of dense vector embeddings:

$$e = [Ex_1, \dots, Ex_T] \quad (5.35)$$

where E is a word embedding matrix. We use the 100-dimensional pretrained word embeddings from Pennington et al. (2014) for mapping the word vectors at the projection layer. We use lower dimensional word embeddings than those we used in the feature extraction approach because our training data is relatively small, and reducing the dimensionality of the word embeddings can significantly reduce the size of the neural network and has lead to better performance of the model in our preliminary experiments. The projected sequence e is then fed into a bidirectional LSTM to compute the recurrent hidden states $[y_1, \dots, y_T]$. Instead of using the last hidden state as the representation for the entire sequence, we feed the output of the LSTM into a mean-over-time layer. The mean-over-time layer combines the hidden states of all time steps into a new vector c of the same length by taking the average of all the hidden state vectors:

$$c = \frac{1}{T} \sum_{t=1}^T y_t \quad (5.36)$$

From this procedure, we obtain the embedded summary and the embedded reading text.

At the merging layer, the encoded vectors are merged together to form a joint representation. We explore four functions to merge the encoded vectors, including concatenation, addition, dot product and linear combination. We apply a nonlinear activation function \tanh after the merging layer. Formally, let a be the embedded reading text and s be the embedded summary, then the merged vector m is obtained by:

- concatenation: The joint representation is a concatenation of the two vector embeddings.

$$m = [a||s] \quad (5.37)$$

- addition: The joint representation is the sum of the two vectors. This requires vector a and vector s are of equal size.

$$m = a + s \quad (5.38)$$

- multiplication: The joint representation is the element-wise product of the two vectors. This requires vector a and vector s are of equal size.

$$m = a \times s \quad (5.39)$$

where each element $m_t = a_t s_t$ for $t = 1, \dots, T$

- linear combination: The joint representation is a linear combination of the two vectors.

$$m = W_a \cdot a + W_s \cdot s + b \quad (5.40)$$

where W_a and W_s are the weight matrices and b is the bias term to be learnt by the model.

The merged vector is used for prediction at the fully connected layer. We build a classification model with a softmax function to predict the binary summary quality for the simulated data and a regression model to predict a real-valued score for the learner data. Similar to the CNN model, we minimize cross entropy loss to train the classification model and mean square loss to train the regression model.

The joint representation learnt in the merged LSTM model encodes the information from both the summary and the original text, and therefore can be directly applied to evaluate how good the summary is with respect to the input text.

5.6.4 Attention-based LSTM model

As the merged LSTM model encodes the summary and reading text separately, it needs to propagate dependencies over long sequences to compare the summary and the text. The joint representation obtained in the merged LSTM model cannot fully capture the connection between the summary and the text. In this section, we propose an attention-based LSTM model which makes use of the attention mechanism to better model the relation between the summary and the reading text.

The attention mechanism was first introduced by Bahdanau et al. (2014) for machine translation. It later proved successful in a wide range of tasks such as image captioning (Xu et al., 2015), sentence summarisation (Nallapati et al., 2016), as well as speech recognition (Hinton et al., 2012b). In general, the attention model learns a soft alignment between the output and the input in the encoder-decoder framework. The attention mechanism allows the model to learn what to attend to in the input states and mitigates the long-dependency bottleneck of the LSTM.

In the attention-based model for summary assessment, the original text and the summary are still encoded using two separate bidirectional LSTMs. However, the text representation is formed by a weighted sum of the hidden states of the text encoder, where the weights can be interpreted as the degree to which the summary attends to a particular token in the text. The summary representation and the text representation are combined with a nonlinear function into a joint representation and then fed into the fully connected layer to predict the summary quality. Figure 5.15 is an illustration of the attention mechanism between the embedded summary and the embedded input text.

Mathematically, suppose s is the encoded summary vector, $a(t)$ is the hidden state of the LSTM for the text at each token t , the final representation r of the encoded text is a weighted

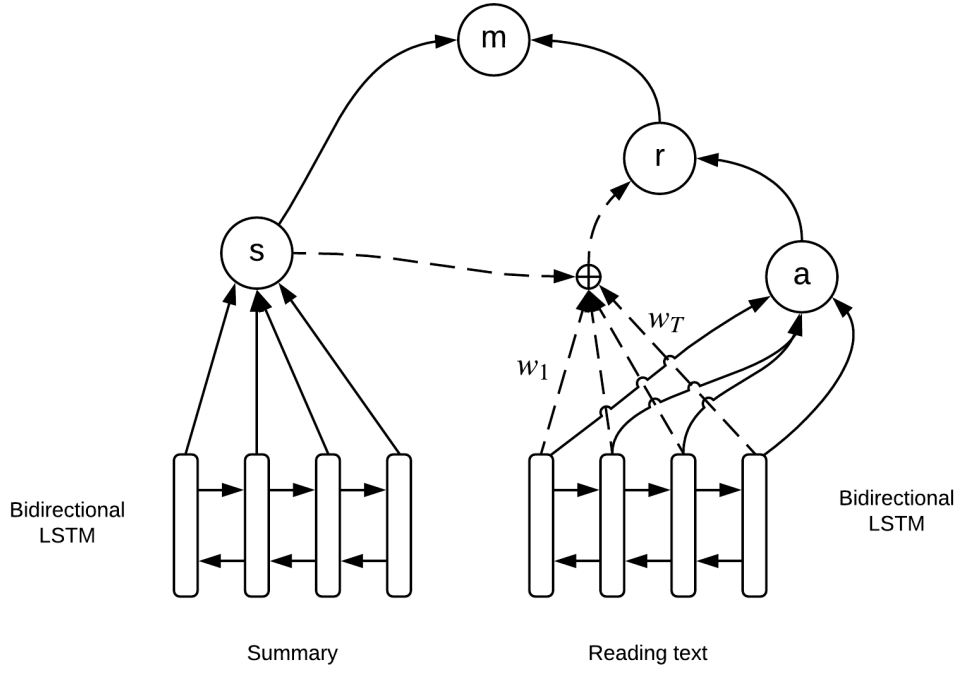


Figure 5.15: An illustration of the attention mechanism in the attention-based LSTM model for summary assessment

sum of $a(t)$, where the weights represent the attention of the summary to each token in the input text. The weight for each token $w(t)$ is computed by:

$$w(t) = \frac{\exp(\alpha(t))}{\sum_{t=1}^T \exp(\alpha(t))} \quad (5.41)$$

where

$$\alpha(t) = W_{a\alpha} \cdot a(t) + W_{s\alpha} \cdot s \quad (5.42)$$

is an alignment model that scores how well the summary and the word token t in the input text match. The score is calculated from the hidden state $a(t)$ and the embedded summary s .

The representation of the text is the dot product between the LSTM hidden states vector a and the weights w :

$$r = a \cdot w = \sum_{t=1}^T a(t)w(t) \quad (5.43)$$

The joint representation m of the summary and the text is a combination of the summary

vector s and the weighted input text vector r .

$$m = \tanh(W_{sm} * s + W_{rm} * r + b) \quad (5.44)$$

where W_{sm} , W_{rm} and b are the parameters of a linear combination function.

Finally, a fully connected layer maps the joint vector m into a score representing the summary quality.

Unlike the merged LSTM model which builds completely independent representations of the summary and the original text, the attention-based model learns what the summary emphasizes in the text, hence, better capturing the connections between the summary and the text.

We use the Keras package to implement our LSTM models.

5.7 Ensemble modelling

Ensemble modelling refers to a group of meta-algorithms that are used to combine several machine learning techniques into one predictive model in order to improve the stability and accuracy of the prediction.

Ordinary machine learning algorithms can be seen as searching through a space of possible functions, called “hypotheses”, to find one function that best approximates the data. For example, one of the most simple and widely used algorithms that searches the hypotheses space is the linear regression model, where the model is trained to identify a linear function that best fits the data. Ensemble modelling takes a different perspective. Rather than finding the one best hypothesis to model the data, ensemble modelling constructs a set of hypotheses and combines the multiple hypotheses to form a better hypothesis. Experiments have shown that ensemble methods are often more accurate than any single hypothesis (Arbib, 2002). This is due to several reasons. First, the search space of hypotheses can be too large for a model to identify a single best function. For example, there may be several different hypotheses that all give the same accuracy on the training data, and the individual learning algorithm has to choose from one of them. However, the chosen hypothesis may not be the best for new data. Having multiple hypotheses and making them vote can reduce the risk. Second, for many machine learning algorithms, the optimal solution can be computationally intractable. For example, in neural network models, we use gradient descent to train the networks. However, such heuristic methods can be trapped in local minima and hence fail to find the best hypothesis. A weighted combination of several hypotheses can reduce the risk of choosing the wrong local minima for prediction. Finally, it sometimes happens that the hypothesis space does not contain any function that is a good approximation to the data. Taking a weighted combination of several hypotheses expands the hypothesis space and hence may form a better approximation.

A typical ensemble method trains several base models in parallel and combines the predictions from the base models into a single prediction. The parallel ensemble method exploits

the independence and diversity of the base models and is usually capable of obtaining better predictive performance than any of the constituent models.

We explore combining the three different machine learning models into a single model by taking the majority vote from the binary classification models and taking the average value of the predicted scores from the regression models. We compare the performance of the combined models against the individual models to investigate if and to what extent ensemble modelling is useful for assessing the summaries.

5.8 Experiments and evaluation

5.8.1 Experimental set-up

We evaluate our models on the simulated learner data (see section 5.3.1) and the real learner data (see section 5.3.2). The data is pre-processed with the NLTK¹⁰ tokenizer to tokenize and convert the text to lowercase.

A 5-fold cross validation approach is adopted for evaluation. In each fold, 60% of the data is used as the training set, 20% as the development set, and 20% as the test set. When splitting the data into different folds, we maintain a similar score distribution in each fold. We train the models on the training set and use the development set to tune model hyperparameters. We choose the best model based on the development set and retrain the selected model using a combination of the training and the development data. Finally, we evaluate the model on the test set.

We compare our models against five baselines:

1. Most frequent baseline: A baseline that predicts the most frequent score in the data. If the scores are equally distributed, we randomly choose one score and always predict that score. For example, in the simulated data, half of the data are good summaries and half are bad summaries. We set the most-frequent baseline to always predict “good”.
2. Random baseline: A random baseline that randomly predicts a score from the probability distribution in the data.
3. ROUGE baseline: We only use the ROUGE metrics to measure the similarity between the summary and the reading text and train a linear SVM or a linear KRR to assess the summary quality.
4. BLEU baseline: Similar to the ROUGE baseline, we only extract the BLEU measures to assess the summary quality.

¹⁰<https://www.nltk.org/>

5. ROUGE + BLEU baseline: A combination of ROUGE and BLEU metrics is used for this baseline.

5.8.2 Evaluation metrics

We use accuracy to evaluate model performance on the simulated data.

On the learner data, we report scores according to three evaluation metrics: Pearson correlation coefficient (introduced in Section 3.4.2), Root Mean Square Error (RMSE), and Spearman's Rank-order Correlation (RHO), which are commonly used for evaluating regression models.

RMSE is frequently used for assessing the difference between the values predicted by a model and the actual observed values. It serves as a good measure of average accuracy by aggregating the magnitudes of the errors in predictions into a single measure of predictive power. The RMSE of the model prediction is the square root of the mean square error between the predicted score \hat{y} and the gold-standard score y :

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2} \quad (5.45)$$

Spearman's correlation is calculated from the Pearson correlation between the ranked values of two variables.

$$\text{RHO} = \text{PCC}_{rg_x, rg_y} = \frac{\text{cov}(rg_x, rg_y)}{\sigma_{rg_x} \sigma_{rg_y}} \quad (5.46)$$

where rg_x and rg_y represents the rank variables converted from the raw scores of pairs of instances in the data.

In contrast to Pearson, Spearman's correlation assesses the strength of a monotonic relation between two variables, rather than a linear relation. Compared to Pearson, it exhibits robustness to the outliers. Yannakoudakis and Cummins (2015) evaluated the performance of the Spearman's correlation for automated essay scoring systems and suggested that Spearman's correlation can be reported alongside other metrics as it can facilitate error analysis and system interpretation. For example, a low accuracy and a high rank correlation would indicate a large misprediction magnitude between the gold and predicted scores, but high agreement with respect to the ranking; a high accuracy and a low rank correlation would indicate that the predicted scores are close to the gold-standard scores, but with many ranking errors.

5.8.3 Results on the simulated learner data

When evaluating our models on the simulated learner data, we train binary classification models using the three proposed methods to predict whether a summary is good or bad. The simulated data was mainly used to develop the models before the real learner data was collected.

Models	Variants		Accuracy	
Baseline	Baseline type	most-frequent	50.0%	
		random	50.0%	
		ROUGE	59.3%	
		BLEU	51.7%	
		ROUGE + BLEU	59.6%	
SVM	reference type	random	58.8%	
		TextRank	63.8%	
		MEAD	62.9%	
		original text	65.6%	
CNN	similarity matrix type	avg word embeddings	65.8%	
		skip-thought vectors	63.4%	
LSTM	Merged LSTM	merging	concatenate	68.0%
			addition	68.1%
		function	multiplication	69.1%
			linear combination	70.4%
	Attention LSTM		71.1%	
Combined model	SVM+CNN+LSTM		75.3%*	

Table 5.3: Model performance on the simulated data. We use the bold font to indicate the best model for each method. The asterisk sign indicates the best performance across all models.

Table 5.3 shows the results of the baseline models and the results of the four types of models on the simulated data. For simplicity, we refer to the feature extraction based model as the “SVM model”, the similarity matrix based model as the “CNN model”, the two variants of the LSTM-based model as the “merged LSTM model” and the “attention LSTM model”. We call the model obtained from the ensemble modelling method as the “combined model”, and in contrast, the models obtained from the three methods independently as “individual models”.

Depending on the reference used in evaluating the summary, there are four variants of the SVM model. The results show that the best SVM variant for assessing the summary uses the original reading text directly as the reference, with an accuracy of 65.6%. Although the sentences selected with extractive summarizers serve as a better reference than the randomly chosen sentences from the original text, they are not as effective as comparing the candidate summary directly to the original input text. This might be due to the fact that the extractive summarizers only select sentences that are highly related to others, where the relation is typically judged by the word overlap, therefore missing the bits of text where topical words occur less often.

Depending on the sentence embedding used to compute the similarity matrix, we compare

Models	Variants		PCC	RMSE	RHO	
Baseline	Baseline type	most-frequent	-	1.30	-	
		random	0.011	1.79	0.007	
		ROUGE	0.499	1.12	0.461	
		BLEU	0.208	2.88	0.285	
		ROUGE + BLEU	0.499	1.11	0.479	
KRR	reference type	random	0.517	1.11	0.504	
		TextRank	0.576	1.06	0.551	
		MEAD	0.557	1.08	0.546	
		original text	0.636	0.99	0.618	
CNN	similarity matrix type	avg word embeddings	0.504	1.12	0.500	
		skip-thought vectors	0.458	1.14	0.446	
LSTM	Merged LSTM	merging	concatenate	0.487	1.13	0.496
			addition	0.466	1.13	0.475
		function	multiplication	0.490	1.12	0.494
			linear combination	0.484	1.13	0.488
	Attention LSTM			0.494	1.12	0.496
Combined model	KRR+CNN+LSTM		0.665*	0.97*	0.653*	

Table 5.4: Results of the regression model performance on the learner data

two variants of the CNN model. Similarity matrix calculated with the average word embeddings yields better results than the one that uses the skip-thought vectors. The CNN model trained on the average word embedding based similarity matrix achieves an accuracy of 65.8%.

The attention LSTM model performs the best among the five variants of the LSTM-based models, which gives an accuracy of 71.1%. It is also the best individual model compared to the SVM and CNN approaches.

The best variants from all three methods outperform the baselines. The improvement is statistically significant ($p < 0.05$) using t -test for all three methods.

We combine the best variants from the three approaches into a single system by taking the majority vote from the models. The resulting system achieves the best accuracy of 75.1% in predicting the binary type of the summary in the simulated data.

5.8.4 Results on the real learner data

To predict the summary scores on the learner data, we train and evaluate regression models built using the three proposed methods. We combine the best individual systems into a single system and compare them against the other models. The results of the regression models are shown in Table 5.4. We also present the results of the five baselines.

For simplicity, we use the term “KRR model” to refer to the feature extraction based

feature type	PCC	RMSE	RHO
verbatim	0.518	1.10	0.507
semantic similarity	0.431	1.17	0.422
document representation	0.137	1.33	0.089
discourse and textual	0.413	1.18	0.408
all	0.636	0.99	0.618

Table 5.5: Regression results with individual feature types on the learner data

ablation tests	PCC	RMSE	RHO
all	0.636	0.99	0.618
- verbatim	0.564	1.07	0.539
- semantic similarity	0.604	1.03	0.588
- document representation	0.617	1.02	0.596
- discourse and textual	0.590	1.04	0.593

Table 5.6: Results of the feature ablation tests on the real learner data. A single type of feature is removed in each of the tests.

regression model and the same terms used for the binary classification models to refer to the models trained with other approaches.

The best individual model on the learner data is the KRR model that uses the original input text as the reference. Similar to the case of the simulated data, comparing the candidate summary to the original text consistently yields better results than using other types of reference.

We compare the four types of features (verbatim features, semantic similarity features, document representations, discourse and other textual features) extracted for the best-performing KRR model and run ablation tests to examine how each type of features contributes to the overall model performance. The results are reported in Table 5.5 and Table 5.6.

According to the ablation tests, all four types of features have contributed to the overall performance of the KRR model. However, when used independently, the verbatim features are the most predictive features for assessing the quality of the summary. This is no surprise as verbatim similarity is the most indicative measure of content similarity in the two texts.

Further, we explore how the individual features correlate with the summary score. Table 5.7 lists the top 10 features that have a positive correlation with the summary score and the top 10 features that have a negative correlation with the summary score.

Unigram overlap features (including METEOR, ROUGE-1 and BLEU-1) are among the top ranked features that have a positive correlation with the summary score. However, as the N-gram size increases, verbatim features that indicate long sequences of copied text in the summary (such as ROUGE-4 and ROUGE-3) correlate negatively with the summary quality. This is because

Top 10 positively correlated features		Top 10 negatively correlated features	
Sentence similarity (skip-thought)	0.284	Text readability	-0.305
Compression ratio	0.267	ROUGE-4	-0.276
Meteor	0.247	Lexical chain - max chain range	-0.249
ROUGE-W	0.196	ROUGE-3	-0.240
ROUGE-1	0.186	Length	-0.239
BLEU-1	0.178	ROUGE-2	-0.187
LDA	0.169	Lexical chain - avg chain range	-0.183
ROUGE-L	0.162	Lexical chain - long chain num	-0.173
TF-IDF	0.155	Word similarity (avg word, greedy)	-0.144
Word similarity (avg word)	0.140	ROUGE-S	-0.144

Table 5.7: The top ranked features with positive and negative correlations with the summary score on the learner data

while lexical overlap is a good indicator of content similarity, good summaries tend to use more paraphrasing and exhibit a higher level of abstraction, which results in a lower overlap of long phrases. It attests our observation that a summary made up of copied sentences is usually not a good summary.

Apart from the verbatim measures, several semantic similarity features and features based on distributed representations of summaries are among the most predictive features for assessing summary quality. Compared to the verbatim measures, these features are capable of capturing content similarity where paraphrasing and abstraction is used. Additionally, we find that the superficial text features are very useful for predicting the summary quality. While there is a negative correlation between the length of the summary and its quality, the quality of the summary also depends on how much information is compressed in the summary. A summary with a low compression ratio is short in length, however, it can also be overly abstract compared to the original passage, which could lead to a low summary score. Surprisingly, lexical chain related features have shown negative correlations with the summary score on the learner data. The negative correlation suggests that the longer the lexical chain is, the worse the summary is. This may be because long lexical chains indicate repetitive information in the summary, which could lead to the summary being partial or missing important information from the original passage. The readability of the original passage is the top ranking feature that has a negative correlation with the summary quality. This is expected because as the reading difficulty of the original text gets higher, it becomes more difficult for the learners to comprehend the text, thus producing summaries of poorer quality.

While the KRR model has shown significant improvement over all the baselines,¹¹ CNN models and LSTM models are not as effective for predicting the summary score on the learner data. The best CNN model only shows a marginal improvement over the correlation of the

¹¹We use the William's test for significance test for the PCC and *t*-test for RMSE and RHO.

ROUGE + BLEU baseline, and the best LSTM model yields poorer performance compared to the ROUGE baseline. On the other hand, both the CNN model and the LSTM model outperform the most frequent, random and BLEU baselines. The results suggest that the neural network based models are not as predictive as the traditional feature extraction based method on this task when the training data is limited.

Although the CNN and LSTM models are not the best-performing models individually, a combination of the three methods (KRR, CNN and LSTM) still improves the performance. We believe that this is because the three independent models capture different aspects of the summary quality that are complementary. In addition, the combined model is more robust to outliers, hence achieving a better performance in estimating the summary quality.

5.8.5 Comparing the automated system with human performance

Pairs of annotation		PCC	RMSE	RHO
Model	with annotator A	0.666	1.19	0.655
	with annotator B	0.577	0.88	0.561
	with annotator C	0.613	1.07	0.606
Model-human average		0.619	1.05	0.607
Human	inter-annotator A-B	0.693	1.13	0.687
	inter-annotator B-C	0.690	1.01	0.690
	inter-annotator A-C	0.794	0.94	0.795
Human-human average		0.726	1.03	0.724

Table 5.8: The results of comparing the automated system with human performance

To measure how our best system compares with human performance, we treat our system as an additional annotator and calculate the agreement between our system and each of the human annotators (A, B, and C) separately. In addition, we calculate the agreement between each pair of annotators. We compute the average agreement among the human annotators and the average agreement between the model and human annotators and compare the scores. The result are shown in Table 5.8.

According to Table 5.8, the automated system doesn't outperform humans on assessing the summary quality, but overall the model performance is close to human performance.

5.8.6 Comparing the systems with previous work

We compare our systems with the work by Madnani et al. (2013), where they extracted 8 linguistic features and trained a logistic regression model to assess student summaries (see Section 2.2). However, it should be noted that their definition of the summarisation task is different from ours.

In their task, students were asked to read a three-paragraph passage and write a four-sentence summary, where the first sentence is about the whole passage and the following three sentences are about each of the three paragraphs. There is no limit to the length of the summary. Although our models are not purposely developed for their task, we examine how well our models perform on their data, and compare our models with their results.

5.8.6.1 ETS-RUF data

Madnani et al. (2013) collected summaries written by more than 2,600 American students from the 6th, 7th and 9th grades on two different passages under a Reading for Understanding Framework (RUF). Specifically, there are 2,695 summaries, with 1,016 summaries on a passage describing the evolution of permanent housing through history and 1679 summaries on a passage describing living conditions at the South Pole.

The summaries consist of four sentences, where first sentence should identify the “global concept” of the passage and the next three sentences should identify “local concepts” corresponding to the main points of each subsequent paragraph.

The summaries are scored by an experienced human rater on a 5-point scale. The definition of the scores are as follows:

Grade 4: summary demonstrates excellent global understanding and understanding of all 3 local concepts from the passage; does not include verbatim text (3+ words) copied from the passage; contains no inaccuracies.

Grade 3: summary demonstrates good global understanding and demonstrates understanding of at least 2 local concepts; may or may not include some verbatim text; contains no more than 1 inaccuracy.

Grade 2: summary demonstrates moderate local understanding only (2-3 local concepts but no global); with or without verbatim text; contains no more than 1 inaccuracy; or good global understanding only with no local concepts.

Grade 1: summary demonstrates minimal local understanding (1 local concept only), with or without verbatim text; or contains only verbatim text.

Grade 0: summary is off topic, garbage, or demonstrates no understanding of the text; or response is “I don’t know”.

Figure 5.16 shows the distribution of the summary scores for the two passages.

5.8.6.2 Results on the ETS-RUF data

We train classification models on the ETS-RUF data to predict the summary scores. Following the practice by Madnani et al. (2013), separate models are trained for each of the two passages and a 5-fold cross-validation protocol is used. We select the best individual models developed on the learner data to train the models and apply an ensemble method to combine the individual

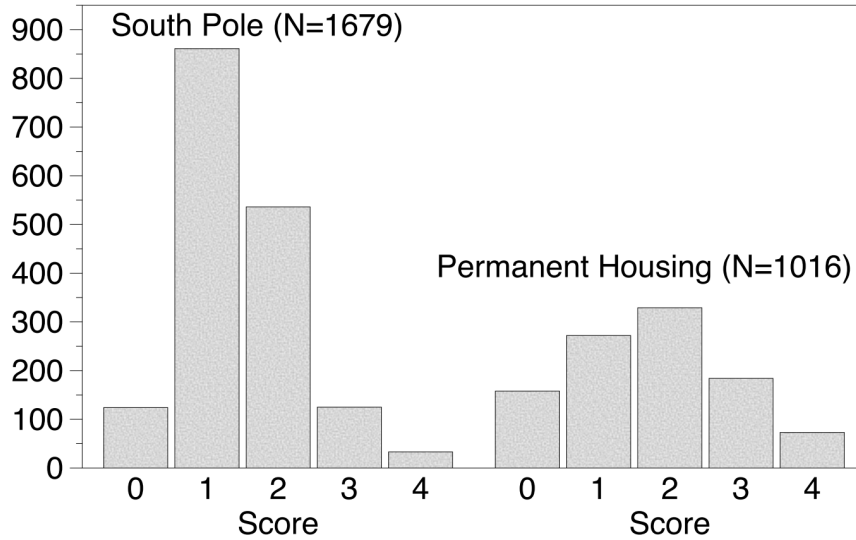


Figure 5.16: The score distribution in the summaries written for the two passages “South Pole” and “Permanent Housing” (taken from Madnani et al., 2013)

Models	South Pole	Permanent Housing
Baseline	0.51	0.32
Madnani et al. (2013)	0.65	0.52
SVM	0.68	0.53
CNN	0.66	0.53
attention LSTM	0.69	0.56
combined model	0.69	0.56

Table 5.9: Performance of the models on the ETS-RUF data

models into a single system. To combine the predictions of the three models, we take the average of the predicted probabilities for each class by the models, and select the class with the highest probability as the final prediction of the summary score. Table 5.9 shows our model performance on the ETS-RUF data. We compare our models with the most frequent baseline and the logistic regression models proposed in Madnani et al. (2013). The models are evaluated using accuracy. The result is reported using two significant digits as in the previous work.

All three models outperform the most-frequent baseline and the logistic regression model. The best model for assessing the summaries on the ETS-RUF data is the attention-based LSTM model. We can see that the neural network based model yields better performance than the SVM model on the ETS-RUF data, perhaps because more training data is available. However, the CNN model which uses a similarity matrix to assess the summary is not as effective as the other two models on this task. We believe this is because the similarity matrix which measures the sentence-to-sentence similarity between the summary and the text is not suited to encode

the four-sentence summary in the ETS-RUF framework, as the four sentences are not of equal weights. As a result, combining the three models doesn't improve the system performance.

5.9 Summary

In this chapter, we introduced a summarisation task for testing reading comprehension of learners and built several automated systems to assess the quality of the learner summary. We collected summaries from members of our university and from the real learners to evaluate our systems. We proposed and compared three methods to assess the summaries, including the feature extraction based model, the similarity matrix and CNN based model, and the LSTM-based model. Although the neural network based models yield better results on the simulated data, they are not as effective as the feature extraction based model on the real learner data. In addition, we investigated how the extracted features contribute to the overall performance on the real learner data. Our experiments showed that the best system for evaluating the learner summary is the one that uses a combination of the three models. By comparing the models with five baselines, we showed that our systems achieved significant improvement over the baseline models. We also compared our best system with the human performance on the real learner data and found that the automated system does not outperform human on assessing the summary quality, but yields results that are close to human performance. When comparing our systems with previous work on the ETS-RUF data, our system outperforms the system presented in the previous work.

Chapter 6

Conclusion

This thesis has focused on two important aspects of non-native reading comprehension: text readability assessment, which estimates the difficulty of text for L2 learners, and summarisation assessment, which evaluates the quality of learner summaries to examine their reading comprehension. We built state-of-the-art systems to assess L2 text readability and to assess learner summaries. We looked into a range of linguistic features affecting L2 readability and explored generalization methods to improve L2 readability assessment using data aimed at native readers. We approached the summarisation assessment task from three different perspectives and demonstrated how the models can effectively assess the summary quality.

In Chapter 3 and Chapter 4, we investigated text readability assessment for both native and L2 learners. We collected a publicly available dataset of texts tailored for language learners and explored methods to adapt models trained on larger existing native corpora for estimating text reading difficulty for learners. In particular, we developed a system that achieves state-of-the-art performance in readability estimation, with $ACC=0.803$ and $PCC=0.900$ on the data aimed at native readers, and $ACC=0.785$ and $PCC=0.924$ on the data aimed at L2 learners, using a linear SVM. We compared the ranking model against the classification model for the task and showed that, although the ranking model does not necessarily outperform the classification model in readability assessment on the same data, it is more accurate when generalizing to an unseen dataset. Following this, we showed that, by applying a mapping function on the output of the ranking model, the model trained on the data aimed at native readers can be applied to estimate the CEFR levels in unseen texts. This model achieves an accuracy of 0.622 and PCC of 0.864, and considerably outperforms the naive generalization of the classification model, which achieves an accuracy of 0.233 and PCC of 0.730.

We compared the linguistic features on predicting reading difficulty on texts aimed at native readers and texts aimed at L2 readers and found that the traditional, lexical and discourse features play a more important role for the L2 readability estimation.

In addition, to make use of the more plentiful data aimed at native readers to produce better estimation of readability when the L2 data is limited in size, we utilised domain adaptation and

self-training approaches. When treating the data aimed at native readers as a source domain and L2 data as a target domain, applying the EasyAdapt algorithm for ranking achieves an accuracy of 0.707 and $PCC=0.899$. The best result is achieved by using self-training to include native data as unlabelled data in training the classification model, with $ACC=0.797$ and $PCC=0.938$.

In Chapter 5, we introduced a summarisation task to assess learners' reading comprehension and successfully built three different models to evaluate the quality of a summary beyond counting exact word match with the original reading passage. In the first approach, we extracted a range of features to measure summary quality and applied a machine learning model to predict the summary score. We made use of distributed representations of various text units in combination with verbatim overlap to capture the content similarity of the summary to the key ideas of the original text. In the second approach, we defined a similarity matrix that measures sentence-to-sentence similarity between the text and the summary, and trained a CNN model to assess the summary quality. In the third approach, we encoded the summary and the text with two separate bidirectional LSTMs and combined the summary embedding and text embedding into a joint representation of the summary quality using a merging function. We also explored using the attention mechanism to improve the joint representation. Finally, we combined the three approaches into a single system, using a simple parallel ensemble modelling method.

We compiled two publicly available datasets to build and evaluate our summarisation assessment system: the simulated data, which is collected from proficient English users and contains summaries of binary quality, and the real learner data, which is collected from real L2 learners of different proficiency levels and consists of summaries annotated on the scale of 0-5. We trained classification models on the simulated data and regression models on the learner data. The best system is built using a combination of three approaches and yields an accuracy of 75.3% on the simulated data, and $PCC = 0.665$, $RMSE = 0.97$, $RHO = 0.653$ on the real learner data. To compare our system with the previous work, we also trained classification models on the ETS-RUF data, which contains four-sentence summaries collected from native English students. We presented state-of-the-art results using our models on the ETS-RUF data.

When comparing the three approaches for assessing summaries, we found that the feature extraction based model outperformed the CNN model and LSTM based models on the real learner data. The result suggested that the neural network based models are not as effective as the traditional feature extraction based method for the regression task at least when the training data is limited in size. We also compared our model performance with that of human, and found that the automated system did not outperform humans on assessing the summary quality but delivered results that are close to the human performance. Although the CNN model did not perform the best compared to our other models, we believe that it has the potential to work well on more structured data and therefore we would like to explore applying the model to other tasks such as measuring the similarity of scientific papers or facts in the knowledge graph.

There are several other directions in which the work presented could be extended. For the

text readability assessment, we plan to improve the readability assessment framework for L2 learners and identify the optimal feature set that can generalize well to unseen text. For the summarisation assessment, we would aim to collect more data from the real learners to improve the performance of our models.

Additionally, in future work, we would like to put the two components together to build a framework for helping learners improve their reading ability efficiently. The framework can be used to find appropriate reading materials of suitable reading difficulty for the L2 learners and at the same time pertain to the personal interests of the readers, and then assess the reading comprehension of the learners using the summarisation task. The score assigned to the summaries can serve as a feedback to adjust the reading difficulty of the passages presented to the learner.

We would also like to explore using text simplification techniques to render the text simple for learners with lower reading ability. For example, if the text has elements that are beyond the learners' CEFR level, we shall reduce the lexical and grammatical complexity of the text to make the information more accessible to the learners.

Another interesting extension is the investigation of the automated methods for other reading tasks. For example, a task immediately related to summarisation is paraphrasing, where the learners are asked to rephrase a sentence in their own words. We would like to see how our models can be applied to detect paraphrases in two sentences.

Additionally, although the current application focuses on assessing learner summaries, the techniques developed for summarisation assessment are also potentially useful for benchmarking automated summarisation systems. Therefore, evaluation of these techniques for benchmarking automated summarization systems will be one direction for our future research.

Finally, we would like to see how the techniques developed in this thesis can contribute to the research in readability assessment and summarisation assessment and be used in educational applications to help learners improve their reading skills.

Bibliography

- Alderson, J Charles (2005). *Assessing reading*. Cambridge University Press.
- Arbib, Michael A. (2002). *The handbook of brain theory and neural networks, Second edition*. The MIT Press.
- Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473*.
- Baroni, Marco, Georgiana Dinu, and German Kruszewski (2014). “Don’t count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors”. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Vol. 1. Association for Computational Linguistics, pp. 238–247.
- Barzilay, Regina and Mirella Lapata (2008). “Modeling local coherence: An entity-based approach”. In: *Computational Linguistics* 34.1, pp. 1–34.
- Bean, Thomas W and Fern L Steenwyk (1984). “The effect of three forms of summarization instruction on sixth graders’ summary writing and comprehension”. In: *Journal of Reading Behavior* 16.4, pp. 297–306.
- Beinborn, Lisa, Torsten Zesch, and Iryna Gurevych (2014). “Readability for foreign language learning: The importance of cognates”. In: *ITL-International Journal of Applied Linguistics* 165.2, pp. 136–162.
- Benjamin, Rebekah George (2012). “Reconstructing readability: Recent developments and recommendations in the analysis of text difficulty”. In: *Educational Psychology Review* 24.1, pp. 63–88.
- Bensoussan, Marsha and Isabelle Kreindler (1990). “Improving advanced reading comprehension in a foreign language: summaries vs. short-answer questions”. In: *Journal of Research in Reading* 13.1, pp. 55–68.
- Bernhardt, Elizabeth B and Michael L Kamil (1995). “Interpreting relationships between L1 and L2 reading: Consolidating the linguistic threshold and the linguistic interdependence hypotheses”. In: *Applied Linguistics* 16.1, pp. 15–34.
- Blei, David M, Andrew Y Ng, and Michael I Jordan (2003). “Latent dirichlet allocation”. In: *Journal of Machine Learning Research* 3.Jan, pp. 993–1022.
- Blondel, Mathieu, Akio Onogi, Hiroyoshi Iwata, and Naonori Ueda (2015). “A ranking approach to genomic selection”. In: *PLOS ONE* 10.6.

- Boureau, Y-Lan, Jean Ponce, and Yann LeCun (2010). “A theoretical analysis of feature pooling in visual recognition”. In: *Proceedings of the 27th International Conference on Machine Learning*, pp. 111–118.
- Branco, Antonio, Jose Rodrigues, Francois Costa, Jaime Silva, and Richard Vaz (2014). “Assessing automatic text classification for interactive language learning”. In: *Proceedings of the 2014 International Conference on Information Society*, pp. 70–78.
- Briscoe, Ted, John Carroll, and Rebecca Watson (2006). “The second release of the RASP system”. In: *Proceedings of the COLING/ACL on Interactive Presentation Sessions*. Association for Computational Linguistics, pp. 77–80.
- Brown, James Dean (1997). “An EFL readability index”. In: *University of Hawai’i Working Papers in English as a Second Language 15 (2)*.
- Bullinaria, John A and Joseph Levy (2007). “Extracting semantic representations from word co-occurrence statistics: A computational study”. In: *Behavior Research Methods 39.3*, pp. 510–526.
- Burnard, Lou (1995). *Users reference guide British National Corpus Version 1.0*. Oxford University Computing Services, UK.
- Callan, Jamie and Maxine Eskenazi (2007). “Combining lexical and grammatical features to improve readability measures for first and second language texts”. In: *Proceedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pp. 460–467.
- Capel, Annette (2012). “Completing the English Vocabulary Profile: C1 and C2 vocabulary”. In: *English Profile Journal 3*.
- Carrell, Patricia L (1987). “Readability in ESL.” In: *Reading in a foreign language 4.1*, pp. 21–40.
- Chall, Jeanne S (1958). *Readability: An appraisal of research and application*. Ohio State University Columbus.
- Chall, Jeanne S and Edgar Dale (1995). *Readability revisited: The new Dale-Chall readability formula*. Brookline Books.
- Chang, Chih-Chung and Chih-Jen Lin (2011). “LIBSVM: A library for support vector machines”. In: *ACM Transactions on Intelligent Systems and Technology 2.3*, p. 27.
- Chapelle, Olivier and S Sathiya Keerthi (2010). “Efficient algorithms for ranking with SVMs”. In: *Information Retrieval 13.3*, pp. 201–215.
- Chen, Stanley F and Joshua Goodman (1999). “An empirical study of smoothing techniques for language modeling”. In: *Computer Speech & Language 13.4*, pp. 359–394.
- Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014). “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *arXiv preprint arXiv:1406.1078*.

- Coleman, Meri and Ta Lin Liao (1975). "A computer readability formula designed for machine scoring". In: *Journal of Applied Psychology* 60.2, p. 283.
- Collins-Thompson, Kevyn (2011). "Enriching information retrieval with reading level prediction". In: *Proceedings of the 2011 Workshop on Enriching Information Retrieval*.
- Collins-Thompson, Kevyn (2013). "Enriching the web by modeling reading difficulty". In: *Proceedings of the Sixth International Workshop on Exploiting Semantic Annotations in Information Retrieval*. ACM, pp. 3–4.
- Collins-Thompson, Kevyn (2014). "Computational assessment of text readability: A survey of current and future research". In: *ITL-International Journal of Applied Linguistics* 165.2, pp. 97–135.
- Collins-Thompson, Kevyn, Paul N Bennett, Ryen W White, Sebastian de la Chica, and David Sontag (2011). "Personalizing web search results by reading level". In: *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, pp. 403–412.
- Collins-Thompson, Kevyn and Jamie Callan (2004a). "A language modeling approach to predicting reading difficulty". In: *Proceedings of the 2004 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pp. 193–200.
- Collins-Thompson, Kevyn and Jamie Callan (2004b). "Information retrieval for language tutoring: An overview of the REAP project". In: *Proceedings of the 27th annual international ACM conference on Research and Development in Information Retrieval*. ACM, pp. 544–545.
- Collins-Thompson, Kevyn and Jamie Callan (2005). "Predicting reading difficulty with statistical language models". In: *Journal of the American Society for Information Science and Technology* 56.13, pp. 1448–1462.
- Cortes, Corinna and Vladimir Vapnik (1995). "Support-vector networks". In: *Machine learning* 20.3, pp. 273–297.
- Coster, William and David Kauchak (2011). "Simple English Wikipedia: a new text simplification task". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*. Vol. 2. Association for Computational Linguistics, pp. 665–669.
- Coxhead, Averil (2000). "A new academic word list". In: *TESOL Quarterly*, pp. 213–238.
- Crossley, Scott A, Jerry Greenfield, and Danielle S McNamara (2008). "Assessing text readability using cognitively based indices". In: *Tesol Quarterly* 42.3, pp. 475–493.
- Dale, Edgar and Jeanne S Chall (1948). "A formula for predicting readability: Instructions". In: *Educational Research Bulletin*, pp. 37–54.
- Dale, Edgar and Jeanne S Chall (1949). "The concept of readability". In: *Euentary English* 26.1, pp. 19–26.

- Daumé III, Hal (2007). “Frustratingly easy domain adaptation”. In: *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Association for Computational Linguistics, pp. 256–263.
- Davison, Alice and Robert N Kantor (1982). “On the failure of readability formulas to define readable texts: A case study from adaptations”. In: *Reading Research Quarterly*, pp. 187–209.
- Day, Richard R and Julian Bamford (1998). *Extensive reading in the second language classroom*. Cambridge University Press.
- Deerwester, Scott, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman (1990). “Indexing by latent semantic analysis”. In: *Journal of the American society for information science* 41.6, p. 391.
- Denkowski, Michael and Alon Lavie (2011). “Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems”. In: *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, pp. 85–91.
- Denning, Joel, Maria Soledad Pera, and Yiu-Kai Ng (2015). “A readability level prediction tool for K-12 books”. In: *Journal of the Association for Information Science and Technology*.
- Drucker, Harris, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik (1997). “Support vector regression machines”. In: *Advances in Neural Information Processing Systems*, pp. 155–161.
- Duke, Nell K and P David Pearson (2009). “Effective practices for developing reading comprehension”. In: *Journal of education* 189.1-2, pp. 107–122.
- Durán, Pilar, David Malvern, Brian Richards, and Ngoni Chipere (2004). “Developmental trends in lexical diversity”. In: *Applied Linguistics* 25.2, pp. 220–242.
- Eisner, Micha and Eugene Charniak (2011). “Extending the entity grid with entity-specific features”. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*. Vol. 2. Association for Computational Linguistics, pp. 125–129.
- Erkan, Günes and Dragomir Radev (2004). “Lexrank: Graph-based lexical centrality as salience in text summarization”. In: *Journal of Artificial Intelligence Research* 22, pp. 457–479.
- Faruqui, Manaal, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith (2014). “Retrofitting word vectors to semantic lexicons”. In: *arXiv preprint arXiv:1411.4166*.
- Feng, Lijun, Noemie Elhadad, and Matt Huenerfauth (2009). “Cognitively motivated features for readability assessment”. In: *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 229–237.
- Feng, Lijun, Martin Jansche, Matt Huenerfauth, and Noemie Elhadad (2010). “A comparison of features for automatic readability assessment”. In: *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics, pp. 276–284.

- Ferne, Tracy and Andre A Rupp (2007). “A synthesis of 15 years of research on DIF in language testing: Methodological advances, challenges, and recommendations”. In: *Language Assessment Quarterly* 4.2, pp. 113–148.
- Finkel, Jenny Rose, Trond Grenager, and Christopher Manning (2005). “Incorporating non-local information into information extraction systems by Gibbs sampling”. In: *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 363–370.
- Forman, George and Ira Cohen (2004). “Learning from little: Comparison of classifiers given little training”. In: *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, pp. 161–172.
- François, Thomas and Cédric Fairon (2012). “An AI readability formula for French as a foreign language”. In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, pp. 466–477.
- François, Thomas and Eleni Miltsakaki (2012). “Do NLP and machine learning improve traditional readability formulas?” In: *Proceedings of the First Workshop on Predicting and Improving Text Readability for Target Reader Populations*. Association for Computational Linguistics, pp. 49–57.
- Friend, Rosalie (2001). “Effects of strategy instruction on summary writing of college students”. In: *Contemporary Educational Psychology* 26.1, pp. 3–24.
- Fuglede, Bent and Flemming Topsøe (2004). “Jensen-Shannon divergence and Hilbert space embedding”. In: *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*. IEEE, p. 31.
- Galley, Michel and Kathleen McKeown (2003). “Improving word sense disambiguation in lexical chaining”. In: *International Joint Conference on Artificial Intelligence*. Vol. 3, pp. 1486–1488.
- Gardner, Matt W and SR Dorling (1998). “Artificial neural networks (the multilayer perceptron): A review of applications in the atmospheric sciences”. In: *Atmospheric environment* 32.14-15, pp. 2627–2636.
- Gershman, Samuel and Joshua B Tenenbaum (2015). “Phrase similarity in humans and machines”. In: *Proceedings of the 37th Annual Meeting of the Cognitive Science Society*. Citeseer.
- Good, Irving J (1953). “The population frequencies of species and the estimation of population parameters”. In: *Biometrika* 40.3-4, pp. 237–264.
- Graesser, Arthur C, Danielle S McNamara, and Jonna M Kulikowich (2011). “Coh-Metrix: Providing multilevel analyses of text characteristics”. In: *Educational Researcher* 40.5, pp. 223–234.

- Green, Spence, Daniel Cer, and Christopher Manning (2014). “An empirical comparison of features and tuning for phrase-based machine translation”. In: *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pp. 466–476.
- Greenfield, Gerald Richard (1999). “Classic readability formulas in an EFL context: Are they valid for Japanese speakers?” PhD thesis. Temple University.
- Hamsik, Marie J (1985). “READING, READABILITY, AND THE ESL READER.” In:
- Harnly, Aaron, Ani Nenkova, Rebecca Passonneau, and Owen Rambow (2005). “Automation of summary evaluation by the pyramid method”. In: *Recent Advances in Natural Language Processing*, pp. 226–232.
- Heilman, Michael, Kevyn Collins-Thompson, Jamie Callan, and Maxine Eskenazi (2006). “Classroom success of an intelligent tutoring system for lexical practice and reading comprehension”. In: *Proceedings of the Ninth International Conference on Spoken Language Processing*. Vol. 2, pp. 829–832.
- Heilman, Michael, Kevyn Collins-Thompson, Jamie Callan, Maxine Eskenazi, Alan Juffs, and Lois Wilson (2010). “Personalization of reading passages improves vocabulary acquisition”. In: *International Journal of Artificial Intelligence in Education* 20.1, pp. 73–98.
- Heilman, Michael, Kevyn Collins-Thompson, and Maxine Eskenazi (2008a). “An analysis of statistical models and features for reading difficulty prediction”. In: *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, pp. 71–79.
- Heilman, Michael and Nitin Madnani (2013). “ETS: Domain adaptation and stacking for short answer scoring”. In: *Proceedings of the Seventh International Workshop on Semantic Evaluation: Second Joint Conference on Lexical and Computational Semantics*. Vol. 2, pp. 275–279.
- Heilman, Michael, Le Zhao, Juan Pino, and Maxine Eskenazi (2008b). “Retrieval of reading materials for vocabulary and reading practice”. In: *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, pp. 80–88.
- Hermann, Karl Moritz, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom (2015). “Teaching machines to read and comprehend”. In: *Advances in Neural Information Processing Systems*, pp. 1693–1701.
- Hill, Felix, Roi Reichart, and Anna Korhonen (2015). “Simlex-999: Evaluating semantic models with (genuine) similarity estimation”. In: *Computational Linguistics* 41.4, pp. 665–695.
- Hill, Margaret (1991). “Writing summaries promotes thinking and learning across the curriculum: But why are they so difficult to write?” In: *Journal of Reading* 34.7, pp. 536–539.
- Hinton, Geoffrey E, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov (2012a). “Improving neural networks by preventing co-adaptation of feature detectors”. In: *arXiv preprint arXiv:1207.0580*.

- Hinton, Geoffrey, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, and Tara N Sainath (2012b). “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups”. In: *IEEE Signal Processing Magazine* 29.6, pp. 82–97.
- Hochreiter, Sepp and Jurgen Schmidhuber (1997). “Long short-term memory”. In: *Neural Computation* 9.8, pp. 1735–1780.
- Hsu, Chih-Wei and Chih-Jen Lin (2002). “A comparison of methods for multiclass support vector machines”. In: *IEEE transactions on Neural Networks* 13.2, pp. 415–425.
- Huang, Anna (2008). “Similarity measures for text document clustering”. In: *Proceedings of the Sixth New Zealand Computer Science Research Student Conference*, pp. 49–56.
- Jurafsky, Daniel and James H Martin (2008). *Speech and language processing: An introduction to natural language processing, 2nd Edition*. Pearson.
- Kate, Rohit J, Xiaoqiang Luo, Siddharth Patwardhan, Martin Franz, Radu Florian, Raymond J Mooney, Salim Roukos, and Chris Welty (2010). “Learning to predict readability using diverse linguistic features”. In: *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, pp. 546–554.
- Kenter, Tom, Alexey Borisov, and Maarten de Rijke (2016). “Siamese CBOW: Optimizing word embeddings for sentence representations”. In: *arXiv preprint arXiv:1606.04640*.
- Kenter, Tom and Maarten De Rijke (2015). “Short text similarity with word embeddings”. In: *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*. ACM, pp. 1411–1420.
- Khalifa, Hanan and Cyril J Weir (2009). *Examining reading*. Cambridge University Press.
- Kidwell, Paul, Guy Lebanon, and Kevyn Collins-Thompson (2011). “Statistical Estimation of Word Acquisition With Application to Readability Prediction”. In: *Journal of the American Statistical Association* 106.493, pp. 21–30.
- Kim, Yoon (2014). “Convolutional neural networks for sentence classification”. In: *arXiv preprint arXiv:1408.5882*.
- Kincaid, J Peter, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom (1975). *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel*. Tech. rep. DTIC Document.
- Kintsch, Walter (1998). *Comprehension: A paradigm for cognition*. Cambridge University Press.
- Kintsch, Walter and Teun A Van Dijk (1978). “Toward a model of text comprehension and production.” In: *Psychological Review* 85.5, p. 363.
- Kiros, Ryan, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler (2015). “Skip-thought vectors”. In: *Advances in Neural Information Processing Systems*, pp. 3294–3302.

- Kneser, Reinhard and Hermann Ney (1995). “Improved backing-off for m-gram language modeling”. In: *Proceedings of the 1995 International Conference on Acoustics, Speech, and Signal Processing*. Vol. 1. IEEE, pp. 181–184.
- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet classification with deep convolutional neural networks”. In: *Advances in Neural Information Processing Systems*, pp. 1097–1105.
- Kuhn, Harold W (1955). “The Hungarian method for the assignment problem”. In: *Naval Research Logistics* 2.1-2, pp. 83–97.
- Kullback, Solomon and Richard A Leibler (1951). “On information and sufficiency”. In: *The annals of mathematical statistics* 22.1, pp. 79–86.
- Landauer, Thomas K (2006). *Latent semantic analysis*. Wiley Online Library.
- Lawrence, Steve, C Lee Giles, Ah Chung Tsoi, and Andrew D Back (1997). “Face recognition: A convolutional neural-network approach”. In: *IEEE transactions on Neural Networks* 8.1, pp. 98–113.
- Le, Quoc V and Tomas Mikolov (2014). “Distributed representations of sentences and documents”. In: *International Conference on Machine Learning*. Vol. 14, pp. 1188–1196.
- Lee, Ching-Pei and Chuan-bi Lin (2014). “Large-scale linear RankSVM”. In: *Neural Computation* 26.4, pp. 781–817.
- Lee, Jeong-Won and Diane Lemonnier Schallert (1997). “The relative contribution of L2 language proficiency and L1 reading ability to L2 reading performance: A test of the threshold hypothesis in an EFL context”. In: *Tesol Quarterly* 31.4, pp. 713–739.
- Lemaire, Benoit, Sonia Mandin, Philippe Dessus, and Guy Denhiere (2005). “Computational cognitive models of summarization assessment skills”. In: *Proceedings of the 27th Annual Meeting of the Cognitive Science Society*, pp. 1266–1271.
- Levy, Omer, Yoav Goldberg, and Ido Dagan (2015). “Improving distributional similarity with lessons learned from word embeddings”. In: *Transactions of the Association for Computational Linguistics* 3, pp. 211–225.
- Liaw, Andy and Matthew Wiener (2002). “Classification and regression by randomForest”. In: *R News* 2.3, pp. 18–22.
- Lin, Chin-Yew (2004). “Rouge: A package for automatic evaluation of summaries”. In: *Proceedings of the Workshop on Text Summarization Branches Out*. Vol. 8. Association for Computational Linguistics, pp. 74–81.
- Lin, Chin-Yew and Eduard Hovy (2003). “Automatic evaluation of summaries using n-gram co-occurrence statistics”. In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Vol. 1. Association for Computational Linguistics, pp. 71–78.
- Liu, Tie-Yan (2009). “Learning to rank for information retrieval”. In: *Foundations and Trends® in Information Retrieval* 3.3, pp. 225–331.

- Louis, Annie and Ani Nenkova (2013). “Automatically Assessing Machine Summary Content Without a Gold Standard”. In: *Comput. Linguist.* 39.2, pp. 267–300. ISSN: 0891-2017.
- Lu, Xiaofei (2011). “A corpus-based evaluation of syntactic complexity measures as indices of college-level ESL writers’ language development”. In: *Tesol Quarterly* 45.1, pp. 36–62.
- Madnani, Nitin, Jill Burstein, John Sabatini, and Tenaha O’Reilly (2013). “Automated scoring of a summary writing task designed to measure reading comprehension”. In: *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 163–168.
- Madnani, Nitin, Joel Tetreault, and Martin Chodorow (2012). “Re-examining machine translation metrics for paraphrase identification”. In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pp. 182–190.
- McClosky, David, Eugene Charniak, and Mark Johnson (2006). “Effective self-training for parsing”. In: *Proceedings of the 2006 Conference of the North American Chapter of the Association of Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pp. 152–159.
- Mihalcea, Rada, Courtney Corley, and Carlo Strapparava (2006). “Corpus-based and knowledge-based measures of text semantic similarity”. In: *Proceedings of the 21st National Conference on Artificial Intelligence*. Vol. 1, pp. 775–780.
- Mihalcea, Rada and Paul Tarau (2004). “TextRank: Bringing order into texts”. In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 404–411.
- Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013a). “Distributed representations of words and phrases and their compositionality”. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119.
- Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig (2013b). “Linguistic regularities in continuous space word representations”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 746–751.
- Miller, George A (1995). “WordNet: a lexical database for English”. In: *Communications of the ACM* 38.11, pp. 39–41.
- Miltsakaki, Eleni and Audrey Troutt (2007). “Read-X: Automatic evaluation of reading difficulty of web text”. In: *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*. Vol. 2007. 1, pp. 7280–7286.
- Miltsakaki, Eleni and Audrey Troutt (2008). “Real-time web text classification and analysis of reading difficulty”. In: *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, pp. 89–97.

- Mitchell, Jeff and Mirella Lapata (2010). “Composition in distributional models of semantics”. In: *Cognitive Science* 34.8, pp. 1388–1429.
- Nallapati, Ramesh, Bowen Zhou, Caglar Gulcehre, and Bing Xiang (2016). “Abstractive text summarization using sequence-to-sequence rnns and beyond”. In: *arXiv preprint arXiv:1602.06023*.
- Nenkova, Ani and Kathleen McKeown (2011). “Automatic summarization”. In: *Foundations and Trends® in Information Retrieval* 5.2–3, pp. 103–233.
- Nenkova, Ani and Rebecca Passonneau (2004). “Evaluating content selection in summarization: The pyramid method”. In: *Proceedings of the 2004 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pp. 145–152.
- Nenkova, Ani, Rebecca Passonneau, and Kathleen McKeown (2007). “The pyramid method: Incorporating human content selection variation in summarization evaluation”. In: *ACM Transactions on Speech and Language Processing* 4.2, p. 4.
- Ney, Hermann, Ute Essen, and Reinhard Kneser (1994). “On structuring probabilistic dependencies in stochastic language modelling”. In: *Computer Speech & Language* 8.1, pp. 1–38.
- Nigam, Kamal, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell (2000). “Text classification from labeled and unlabeled documents using EM”. In: *Machine learning* 39.2-3, pp. 103–134.
- Nuttall, Christine (2005). *Teaching reading skills in a foreign language (3rd Edition)*. Macmillan Education.
- Ogilvie, Paul and Jamie Callan (2001). “Experiments Using the Lemur Toolkit”. In: *Proceedings of the Tenth Text Retrieval Conference*. Vol. 1, pp. 103–108.
- Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd (1999). *The PageRank citation ranking: Bringing order to the web*. Tech. rep. Stanford InfoLab.
- Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002). “BLEU: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 311–318.
- Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (2013). “On the difficulty of training recurrent neural networks”. In: *International Conference on Machine Learning*, pp. 1310–1318.
- Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014). “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 1532–1543.
- Petersen, Sarah E and Mari Ostendorf (2009). “A machine learning approach to reading level assessment”. In: *Computer Speech & Language* 23.1, pp. 89–106.

- Phandi, Peter, Kian Ming A. Chai, and Hwee Tou Ng (2015). “Flexible domain adaptation for automated essay scoring using correlated linear regression”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 431–439.
- Pilán, Ildikó, Sowmya Vajjala, and Elena Volodina (2016a). “A readable read: automatic assessment of language learning materials based on linguistic complexity”. In: *arXiv preprint arXiv:1603.08868*.
- Pilán, Ildikó, Elena Volodina, and Torsten Zesch (2016b). “Predicting proficiency levels in learner writings by transferring a linguistic complexity model from expert-written coursebooks”. In: *Proceedings of the 26th International Conference on Computational Linguistics*, pp. 2101–2111.
- Pitler, Emily and Ani Nenkova (2008). “Revisiting readability: A unified framework for predicting text quality”. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 186–195.
- Porter, Martin F (1980). “An algorithm for suffix stripping”. In: *Program* 14.3, pp. 130–137.
- Qumsiyeh, Rani and Yiu-Kai Ng (2011). “ReadAid: a robust and fully-automated readability assessment tool”. In: *2011 23rd IEEE International Conference on Tools with Artificial Intelligence*. IEEE, pp. 539–546.
- Radev, Dragomir R, Timothy Allison, Sasha Blair-Goldensohn, John Blitzer, Arda Celebi, Stanko Dimitrov, Elliott Drabek, Ali Hakim, Wai Lam, and Danyu Liu (2004a). “MEAD - A platform for multidocument multilingual text summarization”. In: *Proceedings of the Fourth International Conference on Language Resources and Evaluation*.
- Radev, Dragomir, Hongyan Jing, Małgorzata Styś, and Daniel Tam (2004b). “Centroid-based summarization of multiple documents”. In: *Information Processing & Management* 40.6, pp. 919–938.
- Rasmussen, Carl Edward (2004). “Gaussian processes in machine learning”. In: *Advanced lectures on machine learning*. Springer, pp. 63–71.
- Rinehart, Steven D, Steven A Stahl, and Lawrence G Erickson (1986). “Some effects of summarization training on reading and studying”. In: *Reading Research Quarterly*, pp. 422–438.
- Rocktaschel, Tim, Edward Grefenstette, Karl Moritz Hermann, Tomas Kovcsisky, and Phil Blunsom (2015). “Reasoning about entailment with neural attention”. In: *arXiv preprint arXiv:1509.06664*.
- Rodrigo Wilkens, Leonardo Zilio and Cédric Faron (2018). “SW4ALL: a CEFR Classified and Aligned Corpus for Language Learning”. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*. European Language Resources Association.
- Rus, Vasile and Mihai Lintean (2012). “A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics”. In: *Proceedings of*

- the Seventh Workshop on Building Educational Applications Using NLP*. Association for Computational Linguistics, pp. 157–162.
- Schakel, Adriaan MJ and Benjamin J Wilson (2015). “Measuring word significance using distributed representations of words”. In: *arXiv preprint arXiv:1508.02297*.
- Schroff, Florian, Dmitry Kalenichenko, and James Philbin (2015). “Facenet: A unified embedding for face recognition and clustering”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 815–823.
- Schuster, Mike and Kuldip K Paliwal (1997). “Bidirectional recurrent neural networks”. In: *IEEE Transactions on Signal Processing* 45.11, pp. 2673–2681.
- Schwarm, Sarah E and Mari Ostendorf (2005). “Reading level assessment using support vector machines and statistical language models”. In: *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 523–530.
- Shen, Wade, Jennifer Williams, Tamas Marius, and Elizabeth Salesky (2013). *A language-independent approach to automatic text difficulty assessment for second-language learners*. Tech. rep. DTIC Document.
- Si, Luo and Jamie Callan (2001). “A statistical model for scientific readability”. In: *Proceedings of the Tenth International Conference on Information and Knowledge Management*. ACM, pp. 574–576.
- Singhal, Meena (1998). “A comparison of L1 and L2 reading: Cultural differences and schema”. In: *The internet TESL journal* 4.10, pp. 4–10.
- Sparck Jones, Karen (1972). “A statistical interpretation of term specificity and its application in retrieval”. In: *Journal of documentation* 28.1, pp. 11–21.
- Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014). “Dropout: A simple way to prevent neural networks from overfitting”. In: *The Journal of Machine Learning Research* 15.1, pp. 1929–1958.
- Stajner, Sanja, Richard Evans, Constantin Orasan, and Ruslan Mitkov (2012). “What can readability measures really tell us about text complexity”. In: *Proceedings of Workshop on Natural Language Processing for Improving Textual Accessibility*, pp. 14–22.
- Stolcke, Andreas (2002). “SRILM - an extensible language modeling toolkit”. In: *Proceedings of the Seventh International Conference on Spoken Language Processing*. Vol. 2, pp. 901–904.
- Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich (2015). “Going deeper with convolutions”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9.
- Taghipour, Kaveh and Hwee Tou Ng (2016). “A neural approach to automated essay scoring”. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 1882–1891.

- Tanaka-Ishii, Kumiko, Satoshi Tezuka, and Hiroshi Terada (2010). “Sorting texts by readability”. In: *Computational Linguistics* 36.2, pp. 203–227.
- Theil, Henri (1992). “A rank-invariant method of linear and polynomial regression analysis”. In: *Henri Theil’s contributions to economics and econometrics*. Springer, pp. 345–381.
- Thiede, Keith W and Mary CM Anderson (2003). “Summarizing can improve metacomprehension accuracy”. In: *Contemporary Educational Psychology* 28.2, pp. 129–160.
- Urquhart, Alexander H and Cyril J Weir (2014). *Reading in a second language: Process, product and practice*. Routledge.
- Vajjala, Sowmya and Detmar Meurers (2012). “On improving the accuracy of readability classification using insights from second language acquisition”. In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 163–173.
- Vajjala, Sowmya and Detmar Meurers (2014). “Readability assessment for text simplification: From analysing documents to identifying sentential simplifications”. In: *International Journal of Applied Linguistics* 165.2, pp. 194–222.
- Van Dijk, Teun Adrianus, Walter Kintsch, and Teun Adrianus Van Dijk (1983). *Strategies of discourse comprehension*. Academic Press New York.
- Verhelst, Norman, Piet Van Avermaet, Sauli Takala, Neus Figueras, and Brian North (2009). *Common European Framework of Reference for Languages: learning, teaching, assessment*. Cambridge University Press.
- Wade-Stein, David and Eileen Kintsch (2004). “Summary Street: Interactive computer support for writing”. In: *Cognition and Instruction* 22.3, pp. 333–362.
- Wagner, Filho, Alberto Jorge, Rodrigo Wilkens, and Aline Villavicencio (2016). “Automatic Construction of Large Readability Corpora”. In: *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity*, pp. 164–173.
- Wang, Shuohang and Jing Jiang (2016). “Machine comprehension using match-lstm and answer pointer”. In: *arXiv preprint arXiv:1608.07905*.
- Weir, Cyril J (1993). *Understanding and developing language tests*. Prentice-Hall.
- Weir, Cyril J (2005). “Language testing and validation”. In: *Hampshire: Palgrave MacMillan*.
- Weir, Cyril J, Ivana Vidakovic, and Evelina Galaczi (2013). *Measured constructs*. Cambridge University Press.
- Williams, Evan James and EJ Williams (1959). *Regression analysis*. Vol. 14. Wiley New York.
- Xia, Menglin, Ekaterina Kochmar, and Ted Briscoe (2016). “Text readability assessment for second language learners”. In: *Proceedings of the 11th Workshop on Building Educational Applications Using NLP*. Association for Computational Linguistics, pp. 12–22.
- Xu, Kelvin, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio (2015). “Show, attend and tell: Neural image caption gener-

- ation with visual attention”. In: *International Conference on Machine Learning*, pp. 2048–2057.
- Yannakoudakis, Helen (2013). *Automated assessment of English-learner writing*. Tech. rep. UCAM-CL-TR-842.
- Yannakoudakis, Helen and Ronan Cummins (2015). “Evaluating the performance of automated text scoring systems”. In: *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 213–223.
- Yu, Guoxing (2008). “Reading to summarize in English and Chinese: A tale of two languages?” In: *Language Testing* 25.4, pp. 521–551.
- Yu, Lei, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman (2014). “Deep learning for answer sentence selection”. In: *arXiv preprint arXiv:1412.1632*.
- Zanzotto, Fabio Massimo, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar (2010). “Estimating linear models for compositional distributional semantics”. In: *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, pp. 1263–1271.
- Zeeland, Hilde van and Norbert Schmitt (2012). “Lexical coverage in L1 and L2 listening comprehension: The same or different from reading comprehension?” In: *Applied Linguistics* 34.4, pp. 457–479.
- Zhao, Jin and Min-Yen Kan (2010). “Domain-specific iterative readability computation”. In: *Proceedings of the 10th Annual Joint Conference on Digital Libraries*. ACM, pp. 205–214.
- Zhu, Xiaojin (2005). *Semi-supervised learning literature survey*. Tech. rep. University of Wisconsin-Madison.
- Zhu, Yukun, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler (2015). “Aligning books and movies: Towards story-like visual explanations by watching movies and reading books”. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 19–27.
- Zhu, Zhemin, Delphine Bernhard, and Iryna Gurevych (2010). “A monolingual tree-based translation model for sentence simplification”. In: *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, pp. 1353–1361.

Appendix A

Examples of the original WeeBit corpus

We obtained a copy of the original WeeBit corpus from Vajjala and Meurers (2012). The original texts are webpage documents stored in raw HTML format. We identified that some of the texts extracted by the authors contain broken sentences, missing content or extraneous content, such as copyright declarations and links, that are likely to influence performance of supervised machine learning models on the task. Here are some examples of texts in the original version of the WeeBit corpus that have such problems:

Example 1 (a text at Level 1): Arthur the aardvark will celebrate his 25th birthday later this month. To readers, Arthur will always be 8 years old. But Marc Brown has been writing and drawing the Arthur books for 25 years. talked to Marc Brown about his work. Here's what he had to say. Telling bedtime stories to my son was one of my favorite things to do. So I decided to write and draw kids' books. Why did you decide to write about an aardvark? My son wanted to hear a story about a weird animal. How do you decide what each book will be about? I try to make the stories fun to read. But I also write about things that are important to kids. Did you have a favorite book as a kid? Do you have any advice for a kid who wants to become a writer? Read, read, read! That's a great way to learn how to put words together. And keep a journal.

All trademarks and logos are property of Weekly Reader Corporation.

Example 2 (a text at Level 1) The table below shows about how long and how heavy four different dinosaurs were. Look at the table and answer the questions.

Which of the four dinosaurs weighed the most?

Which of the four dinosaurs had the shortest length?

If -saurus means "lizard" and brachio- means "arm," then what might Brachiosaurus mean?

All trademarks and logos are property of Weekly Reader Corporation.

Example 3 (a text at Level 2) In November, astronauts began living aboard the

People will be able to live in the space station for long periods of time.

U.S. astronaut Bill Shepherd is the leader of the first

All trademarks and logos are property of Weekly Reader Corporation.

Example 4 (a text at Level 3) Find the adjectives in the list below.

All trademarks and logos are property of Weekly Reader Corporation.

Example 5 (a text at Level 4) We highlight important parts of Shakespeare's plays and show you how to use quotes to explain your points.

All things historic - from ancient to modern.

The BBC is not responsible for the content of external internet sites.

This page is best viewed in an up-to-date web browser with style sheets (CSS) enabled.

While you will be able to view the content of this page in your current browser, you will not be able to get the full visual experience. Please consider upgrading your browser software or enabling style sheets (CSS) if you are able to do so.

Example 6 (a text at Level 5) yourself on Food acidity, oxidation and temperature

To avoid these problems, we re-extracted texts from the raw HTML and discarded text documents that do not contain proper reading passages by filtering the text with a Perl script.

Appendix B

Examples of the modified WeeBit corpus

Here are some examples of texts at different reading levels from our re-extracted WeeBit corpus:

Example 1 (a text at Level 1) What Is a Cavity ?

A cavity is a hole in a tooth. Cavities can start when food and germs stick onto teeth. The food and germs can turn into acid. The acid makes a tiny hole in the tooth. If the hole isn't fixed , the cavity can get bigger.

Thinking Cue

Cavities can form in between teeth. Where else might a cavity form ?

Example 2 (a text at Level 2) Ancient Shipwreck Found

Secrets about the past lie deep below the surface of the sea. In January , explorers reported their discovery of an ancient Greek ship on the bottom of the Black Sea. The Black Sea is located between Europe and Asia. Scientists say the ship is about 2,400 years old! "The Greeks went into the Black Sea for fish and gold," said Robert Ballard, the explorer whose team found the wreck. Ballard is the same explorer who discovered the Titanic. Ballard said the ship in the Black Sea was used for trade. The ship was filled with jars. At one time, the jars held items such as fish, olive oil, and honey. This summer Ballard's team plans to dig up the shipwreck. They are hoping to learn more secrets about the ancient world.

Example 3 (a text at Level 3): A Mammoth Discovery

Scientists recently confirmed that large, mysterious bones found near a river in San Jose, California, weren't those of any ordinary animal. They belonged to a Columbian mammoth, an elephant-like creature that roamed the area's grasslands between 10,000 and 40,000 years ago. Those ancient animals weighed 10 tons and stood 13 feet tall. Columbian mammoths were found throughout the southern half of North America 2 million years ago. They lived during the mild weather between Ice Ages. An ice age was a period

of time when large sheets of ice covered Earth. Similar to elephants today, Columbian mammoths are believed to have had short gray hair. They are thought to be related to woolly mammoths – large animals with shaggy coats that lived during the Ice Age. Scientists don't know why either mammoth vanished. This find marks the most complete mammoth skeleton ever discovered in California. Scientists will study the bones to learn more about these creatures. "This is important because it creates a record of the animals that wandered the area," scientist Mark Goodwin told Weekly Reader. "They are an important part of California history."

Example 4 (a text at Level 4) Attack of Guinea-Zilla!

A rodent the size of a buffalo? Researchers say they have found fossils of a 1,545-pound giant guinea pig-like rodent that thrived 6 million to 8 million years ago in a swampy region of South America. "Imagine a weird guinea pig, but huge, with a long tail for balancing on its hind legs and continuously growing teeth," scientist Marcelo R. Sanchez-Villagra said. Researchers from Germany, Venezuela, and the United States who identified the skeleton have nicknamed the creature "Guinea-zilla" because it looked like a Godzilla-sized guinea pig. The team dug up the skeleton of the giant rodent three years ago in the Urumaco fossil fields in Venezuela. Scientists believe that Urumaco, now a desert town, was once lush with vegetation. The prehistoric animal, whose scientific name is *Phoberomys pattersoni*, was 9 feet long and more than 4 feet tall. The prehistoric rodent roamed near an ancient river, eating grasses and dodging predators. One of those predators was a tractor-trailer-sized crocodile that weighed 12 tons and grew to 40 feet long. Rodents, such as mice, rats, hamsters, guinea pigs, beavers, muskrats, and squirrels, now account for about 1,500 of the 4,000 living species of mammals. So why aren't buffalo-sized rodents roaming alleyways and subway tunnels today? Being big is no guarantee of survival. "The question that really puzzles me is not how *Phoberomys* could have been so large but why (most) rodents are so small," biologist R. McNeill Alexander said.

Example 5 (a text at Level 5) How drugs affect our nervous system

Some drugs and toxins affect how impulses pass from one neuron to the next across a synapse. Drugs and synapses Strychnine is used by Australian aborigines to paralyze fish. Some drugs stop the impulse from passing across the synapse. Drugs such as curare (the South American plant toxin used in arrow poison) do this. They cause complete paralysis, and even stop the person from breathing. Other drugs stimulate the synapse so that once an impulse crosses the gap the impulse is repeated over and over again. Drugs such as strychnine do this. They cause all the muscles in the body to go into a continuous spasm of constriction. This also stops the person from breathing. Read on if you are taking the Higher paper. Serotonin is a chemical that is released into synapses in the brain. An increase in serotonin levels in the synapses makes us feel happier. However, serotonin is

normally absorbed by receptor molecules on the other side of the synapse. This prevents the levels of serotonin from increasing. Ecstasy (also called MDMA) is a drug that blocks the serotonin receptor sites in the synapses in the brain. This prevents the serotonin from being absorbed by the receptor molecules. As a result, the level of serotonin in the synapse increases. This produces a feeling of wellbeing. However, there is evidence to suggest that the use of Ecstasy reduces memory. Ecstasy can also cause severe dehydration which can result in death.

Appendix C

An example from the Cambridge English Exams data

We collected a L2 readability dataset from the past reading papers of Cambridge English Exams. Here is an example reading text from the past paper of the First Certificate in English (FCE) test:


Part 2

You are going to read an article about a bird called the kingfisher. Seven sentences have been removed from the article. Choose from the sentences A–H the one which fits each gap (9–15). There is one extra sentence which you do not need to use.

Mark your answers on the separate answer sheet.

The kingfisher

Wildlife photographer Charlie James is an expert on the kingfisher: a beautiful blue-green bird that lives near streams and rivers, feeding on fish.



Old trees overhang the stream, half shading shallow water. Soft greens, mud browns and the many different yellows of sunlight are the main colours, as soft as the sounds of water in the breeze. The bird cuts like a laser through the scene, straight and fast, a slice of light and motion so striking you almost feel it. It has gone in a split second, but a trace of the image lingers, its power out of proportion to its size.

Charlie James's first hideout was an old blanket which he put over his head while he waited near a kingfisher's favourite spot. [9] But it took another four years, he reckons, before he got his first decent picture. In the meantime, the European kingfisher had begun to dominate his life. He spent all the time he could by a kingfisher-rich woodland stream.

The trouble was, school cut the time available to be with the birds. So he missed lessons, becoming what he describes as an 'academic failure'. [10]

At 16, he was hired as an advisor for a nature magazine. Work as an assistant to the editor followed, then a gradual move to life as a freelance wildlife film cameraman. What he'd really like to do now is make the ultimate kingfisher film. [11] 'I'm attracted to the simple approach. I like to photograph parts of kingfisher wings ...'

The sentence trails off to nothing. He's thinking of those colours of the bird he's spent more than half his life getting close to, yet which still excites interest. [12] But, as Charlie knows, there's so much more to his relationship with the kingfisher than his work can ever show.

Charlie James fell in love with kingfishers at an early age. [13] After all, it is the stuff of legend. Greek myth makes the kingfisher a moon goddess who turned into a bird. Another tale tells how the kingfisher flew so high that its upper body took on the blue of the sky, while its underparts were scorched by the sun.

[14] For despite the many different blues that appear in their coats, kingfishers have no blue pigment at all in their feathers. Rather, the structure of their upper feathers scatters light and strongly reflects blue.

[15] It's small wonder that some wildlife photographers get so enthusiastic about them. Couple the colours with the fact that kingfishers, though shy of direct human approach, can be easy to watch from a hideout, and you have a recipe for a lifelong passion.

A This is why a kingfisher may appear to change from bright blue to rich emerald green with only a slight change in the angle at which light falls on it.

B But his interest in this, the world's most widespread kingfisher and the only member of its cosmopolitan family to breed in Europe, was getting noticed.

C A sure sign of his depth of feeling for this little bird is his inability to identify just what it is that draws him to it.

D The movement sends a highly visible signal to rivals, both males and females, as it defends its stretch of water against neighbours.

E The bird came back within minutes and sat only a metre away.

F The photographs succeed in communicating something of his feelings.

G 'No speech, just beautiful images which say it all,' he says.

H There is some scientific truth in that story.

Figure C.1: An example reading text from FCE

We removed the reading tasks associated with the reading passage (in this case for example, by inserting the missing paragraphs in the right order to the reading passage) and kept only the reading passage in our dataset.¹

¹Due to copyright reasons, we can only provide one example in the thesis.

The extracted reading passage reads as follow:

Example

The Kingfisher

Wildlife photographer Charlie James is an expert on the kingfisher: a beautiful blue-green bird that lives near streams and rivers, feeding on fish. Old trees overhang the stream, half shading shallow water. Soft greens, mud browns and the many different yellows of sunlight are the main colours, as soft as the sounds of water in the breeze. The bird cuts like a laser through the scene, straight and fast, a slice of light and motion so striking you almost feel it. It has gone in a split second, but a trace of the image lingers, its power out of proportion to its size.

Charlie James fell in love with kingfishers at an early age. A sure sign of his depth of feeling for this little bird is his inability to identify just what it is that draws him to it. After all, it is the stuff of legend. Greek myth makes the kingfisher a moon goddess who turned into a bird. Another tale tells how the kingfisher flew so high that its upper body took on the blue of the sky, while its underparts were scorched by the sun.

There is some scientific truth in that story. For despite the many different blues that appear in their coats, kingfishers have no blue pigment at all in their feathers. Rather, the structure of their upper feathers scatters light and strongly reflects blue.

This is why a kingfisher may appear to change from bright blue to rich emerald green with only a slight change in the angle at which light falls on it. It's small wonder that some wildlife photographers get so enthusiastic about them. Couple the colours with the fact that kingfishers, though shy of direct human approach, can be easy to watch from a hideout, and you have a recipe for a lifelong passion.

Charlie James' first hideout was an old blanket which he put over his head while he waited near a kingfisher's favourite spot. The bird came back within minutes and sat only a metre away. But it took another four years, he reckons, before he got his first decent picture. In the meantime, the European kingfisher had begun to dominate his life. He spent all the time he could by a kingfisher-rich woodland stream.

The trouble was, school cut the time available to be with the birds. So he missed lessons, becoming what he describes as an 'academic failure'. But his interest in this, the world's most widespread kingfisher and the only member of its cosmopolitan family to breed in Europe, was getting noticed. At 16, he was hired as an advisor for a nature magazine. Work as an assistant to the editor followed, then a gradual move to life as a freelance wildlife film cameraman. What he'd really like to do now is make the ultimate kingfisher film. 'No speech, just beautiful images which say it all,' he says. 'I'm attracted to the simple approach. I like to photograph parts of kingfisher wings ...'

The sentence trails off to nothing. He's thinking of those colours of the bird he's spent more than half his life close to, yet which still excites interest. The photographs succeed in communicating something of his feelings. But, as Charlie knows, there's so much more to his relationship with the kingfisher than his work can ever show.

Appendix D

Examples of “good” and “bad” summaries in the simulated learner data

Here we give examples of a good summary and a bad summary to the reading text titled “Kingfisher” (see Appendix C) in the simulated learner data:

Example A (a good summary) The kingfisher is a blue-green bird that lives near streams and rivers, feeding on fish. Kingfishers have no blue pigment at all in their feathers. Rather, the structure of their upper feathers scatters light and strongly reflects blue. The kingfisher is the stuff of legend. For instance, a Greek myth makes the kingfisher a moon goddess who turned into a bird. Wildlife photographer Charlie James is an expert who fell in love with kingfishers at an early age. At 16, he was hired as an advisor for a nature magazine. In the future, he would like to make the ultimate kingfisher film, with ‘no speech, just beautiful images’.

Example B (a bad summary) Charlie James is an expert in kingfisher. Kingfisher cuts like laser through the scene in a split of a second. Charlie James cannot identify what it is that draws him to this bird. There are several legends about the kingfisher. One tells that it is a moon goddess who turned into a bird. Another describes that kingfisher flew so high that its upper body took on the blue of the sky, while its underparts were scorched by the sun. It took Charlie James several years to take the first decent picture of the kingfisher. He later became specifically interested in the European kingfisher - the world’s most widespread kingfisher and the only member of its cosmopolitan family to breed in Europe.

Appendix E

Examples of summaries in the real learner data

In this appendix, we include some examples of learner summaries to a reading passage titled “Winter Driving”.

The reading text is as follow:

Winter Driving

Winter is dangerous because it’s so difficult to know what is going to happen and accidents take place so easily. Fog can be waiting to meet you over the top of a hill. Ice might be hiding beneath the melting snow, waiting to send you off the road. The car coming towards you may suddenly slide across the road.

Rule Number One for driving on icy roads is to drive smoothly. Uneven movements can make a car suddenly very difficult to control. So every time you either turn the wheel, touch the brakes or increase your speed, you must be as gentle and slow as possible. Imagine you are driving with a full cup of hot coffee on the seat next to you. Drive so that you wouldn’t spill it.

Rule Number Two is to pay attention to what might happen. The more ice there is, the further down the road you have to look. Test how long it takes to stop by gently braking. Remember that you may be driving more quickly than you think. In general, allow double your normal stopping distance when the road is wet, three times this distance on snow, and even more on ice. Try to stay in control of your car at all times and you will avoid trouble.

Here are the example summaries and their scores, as well as our analysis of the possible reasons how their scores are resolved:

Example 1 Last Winter, I drank coffee at the top of a hill. After I finished, I slid down to the bottom on a toboggan. One year, I had an accident because the brakes failed and I crashed into a tree. In future, I will learn to be more careful.

Score: 0 Reason: The summary is completely irrelevant.

Example 2 Driving in the winter season, it's too dangerous, even though you can considerate two rules to less the probability of having an accident: the first one, is to drive as lower as possible, and the second one, you need to focus on giving three times the stopping distance on snow.

Score: 2 Reason: Summary covers some of the main points of the passage, and it focuses on details rather than the main point towards the end.

Example 3 There are many reasons that make driving in winter dangerous, such as fog, ice and coming cars. The first rule for driving in winter is to drive smoothly. The second rule is to be more careful and stay in control of the car.

Score: 3 Reason: Summary covers most of the main points of the passage, however it misses some of the key information (in both the first and second rules).

Example 4 Driving in Winter can be dangerous due to ice and fog. You should drive slowly and carefully and take no sudden actions. You should also pay attention to the road and anticipate any hazards in advance because the brakes are less effective in bad conditions.

Score: 5 Reason: The summary covers all the main points of passage. Everything is relevant. The content is accurate and without repetition.