A Thesis submitted for the degree of Master of Science in Intelligent Adaptive Systems

Complex Word Identification for Language Learners

Ankit Srivastava (Matr. Nr.: 7133638)
ankit.srivastava@uni-hamburg.de

Supervisors
Prof. Dr. Chris Biemann,
Dr. Özge Alaçam,
Xintong Wang

13^{th} March 2022

Universität Hamburg

Abstract

Lexical Simplification (LS) aims to provide simpler alternatives for the complex words in a text preserving equivalent meaning. This helps various groups of people like non-native speakers, and children to better understand a given text. One of the crucial steps in the LS pipeline is to identify complex words in a text to simplify, which considerably improves simplification. Therefore in this thesis, I propose my approaches to build a Complex Word Identification (CWI) system for English language learners, which takes context into account. As a basis of my CWI system, I use the sequential model by Rei (2017), which has been adopted by the current state-of-the-art SEQ model (Gooding and Kochmar 2019) and further modify it to improve the model performance. For model training and evaluation I used the same data that was used by the SEQ model, i.e., CWI Shared Task 2018 data sets by Yimam et al. (2018). The SEQ model does not incorporate any feature engineering and outperforms the previous state-of-the-art systems on CWI shared task 2018 Yimam et al. (2018). I hypothesized that uniting linguistic knowledge with the power of a neural network could improve the model performance. In this thesis, I proposed a hybrid model, by modifying the sequential framework to include hand-engineered features along with word embeddings. Among various features identified for this task, I answered which features are important or do not have a significant impact on the hybrid model performance for the CWI task. Further, I also fine-tuned transformer models for this task as it is broadly adopted in various Natural Language Processing (NLP) tasks owing to its parallelism and advantage in modeling the long-range context. There is a lack of research in using transformers for CWI task on CWI shared task 2018 data, which I tried to cover in this thesis. Using transformers, I achieved the state-of-the-art results for the CWI task, proving transformer-like encoder is just as effective for CWI as other NLP tasks.

Contents

1 Introduction **1**

2 Related Work **5**
2.1 SemEval-2016 Shared Task on CWI 5
2.2 Shared Task 2018 on CWI 6
2.3 SemEval-2021 Shared Task on CWI 7

3 Datasets **9**
3.1 L2 Data Annotation 12

4 Data Setup **15**

5 Evaluation Metrics **16**

6 Sequential Model for CWI **17**
6.1 Sequential Architecture 17
6.2 Approach 18
6.3 The Pilot Experiment 19
6.3.1 Word Embeddings 19
6.3.2 Recall-oriented Learning 21
6.4 Hybrid Architecture 22
6.5 Features for CWI 22
6.6 Features Details & Importance in Hybrid Approach 24
6.6.1 Sentence Length 25
6.6.2 Sentence Complexity 25
6.6.3 Cosine Similarity 26
6.6.4 Word Length 27
6.6.5 Syllables 27
6.6.6 Synonyms 28
6.6.7 Hypernyms 29
6.6.8 Hyponyms 29
6.6.9 Word Frequency: Google dataset 30
6.6.10 Word Frequency: Lang-8 Corpus 31
6.6.11 SubIMDB 31
6.6.12 Simple Wikipedia 32
6.6.13 Ogden's Basic English 32

6.7 Experiment using Hybrid Approach . . . 33
6.7.1 Features Selection . . . 33
6.7.2 Hybrid Experiment . . . 34

7 Tuning Transformers for CWI **37**
7.1 Training Data Setup . . . 37
7.2 Approach . . . 40
7.3 Experiment . . . 40
7.3.1 Fine-Tuning RoBERTa Large Model . . . 41
7.3.2 Fine-Tuning BERT Base Cased Model: Training with $\mathbf{Train_{All}}$ data 44

8 Results and Error Analysis **46**
8.1 Results . . . 46
8.1.1 CWI Shared Task 2018 Dataset . . . 46
8.1.2 L2 Dataset . . . 48
8.2 Error Analysis . . . 49
8.2.1 News Testset . . . 49
8.2.2 WikiNews Testset . . . 51
8.2.3 Wikipedia Testset . . . 52
8.2.4 InScript Testset . . . 54
8.2.5 OneStop Testset . . . 56

9 Conclusion and Future Work **59**
9.1 Conclusion . . . 59
9.2 Future Work . . . 61

Bibliography **62**

List of Figures

1.1 Example demonstrating steps in Lexical Simplification Pipeline. 1

3.1 Distribution of complex and non-complex words in Shared Task 2018 testsets. 10
3.2 Distribution of complex and non-complex words in L2 data. 13

4.1 CWI Shared Task 2018 data format. 15

6.1 The unfolded network structure for a sequence labeling model with an additional language modeling objective, performing CWI on the sentence "The Hamburg University". The input tokens are shown at the bottom, the expected output labels are at the top. The arrows above variables indicate the directionality of the Bi-LSTM. 18
6.2 The unfolded network structure for a sequence labeling model with an additional language modeling objective, performing CWI on the sentence "The Hamburg University". The input tokens are shown at the bottom, the expected output labels are at the top. The arrows above variables indicate the directionality of the Bi-LSTM. Features (f_1, f_2, f_3) are concatenated to the output vector from Bi-LSTM 23
6.3 Permutation feature importance using Logistic Regression. 35

7.1 Tokens distribution in different training sets. 38
7.2 Word complexity distribution in different training sets. 39
7.3 Learning curve for training data with all stop words and punctuation ($Train_{All}$). 41
7.4 Learning curve for train data without punctuation 42
7.5 Learning curve for train data without stop words and punctuation 43
7.6 Learning curve for train data using BERT model 45

8.1 Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the News testset. . . . 49
8.2 Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the News testset. 50
8.3 Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the WikiNews testset. 51
8.4 Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the WikiNews testset. 52

8.5 Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the Wikipedia testset. 53
8.6 Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the Wikipedia testset. 54
8.7 Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the InScript testset. . 55
8.8 Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the InScript testset. 56
8.9 Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the OneStop testset. . 57
8.10 Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the OneStop testset. 58

List of Tables

1.1 Words with different meaning based on context 2

3.1 Number of words (w) and phrases (ph) annotated in CWI shared task datasets. 9
3.2 Total number of sentence and token distribution in Shared Task 2018 data. 10
3.3 Number of tokens, sentences and maximum sentence length of InScript data. 11
3.4 Number of tokens, sentences and maximum sentence length of OneStop data. 11
3.5 Token distribution in L2 data. 12
3.6 Text examples from L2 Data (bold words are complex). 14

6.1 Precision, recall and macro F-score achieved for different testsets. 19
6.2 Comparison of macro F-scores achieved by $SEQ_{baseline}$ to the state-of-the-art SEQ system. 20
6.3 Precision, recall and macro F-score achieved for different testsets using FastText embeddings 20
6.4 Comparison of results achieved from $SEQ_{baseline}$ to $SEQ_{FastText}$ 21
6.5 Precision, recall and macro F-score achieved for different testsets while improving recall. 21
6.6 Comparison of results achieved from $SEQ_{FastText}$ model to SEQ_{FTRec} model. 22
6.7 F-score comparison of the SEQ_{FTRec} model results to the hybrid model with normalized sentence length feature. 25
6.8 F-score comparison of the SEQ_{FTRec} model to the hybrid model with sentence complexity feature. 26
6.9 F-score comparison of the SEQ_{FTRec} model to the hybrid model with cosine similarity feature. 27
6.10 F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized word length feature. 27
6.11 F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized syllables feature. 28
6.12 F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized Synonym feature. 28
6.13 F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized Hypernyms feature. 29
6.14 F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized Hyponyms feature. 30

6.15 F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized word frequency in Google dataset as feature. 30
6.16 F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized word frequency in Lang-8 corpus as feature. 31
6.17 F-score comparison of the SEQ_{FTRec} model to the hybrid model with subIMDB as feature. 32
6.18 F-score comparison of the SEQ_{FTRec} model to the hybrid model with simple Wikipedia as feature. 32
6.19 F-score comparison of the SEQ_{FTRec} model to the hybrid model with Ogden's Basic English as binary feature. 33
6.20 F-score comparison of the SEQ_{FTRec} model to the SEQ_{hybExp} model introduced with $Features_{exp}$ features. 35
6.21 F-score comparison of the SEQ_{FTRec} model to the SEQ_{hybPI} model introduced with $Features_{PI}$ features. 36

7.1 Total number of sentence, tokens, complex and non-complex tokens and percentage of non-complex tokens in each training data for transformer . . 39
7.2 Comparison of macro F-Scores achieved by fine-tuning RoBERTa on $Train_{All}$ data to baseline results. 42
7.3 Comparing macro F-Scores achieved by fine-tuning RoBERTa on $Train_{WoP}$ data to baseline results. 43
7.4 Comparing macro F-Scores achieved by fine-tuning RoBERTa on $Train_{clean}$ data to baseline results. 44
7.5 Comparison of macro F-Scores and accuracies achieved by fine-tuning RoBERTa model with different training sets. 44
7.6 Comparing macro F-Scores achieved by fine-tuning BERT using $Train_{All}$ data to the baseline results. 45

8.1 Comparison of macro F-scores achieved from state-of-the-art SEQ model to various approaches in this thesis for CWI in News, WikiNews and Wikipedia testsets from Shared Task 2018. The values displayed in this table are in percentage. 47
8.2 Comparison of macro F-scores achieved from baseline $SEQ_{baseline}$ model to various approaches in this thesis for CWI in InScript and OneStop testsets from L2 data. 48
8.3 Similar text (bold words are annotated as complex) from News train and test set and predicted complex words from various models. 50
8.4 Text from Wikipedia testset (bold words are annotated as complex) and incorrect predictions from models. 54
8.5 Advanced level text (bold words are annotated complex) from OneStop test set and word complexity predictions from different models. 58

1 Introduction

Complex Word Identification (CWI) is a base for many other applications like Lexical Simplification (LS), a multimodal application to show images for complex words or machine translation. The words which can not be visually depictable, or translated to other languages due to rareness are simplified using the LS system for further processing. The LS system aims at replacing complex words of text with simpler alternatives while preserving its meaning and grammaticality. It helps people with low literacy, reading difficulties, and non-native speakers to understand the text better. Most of the modern LS pipeline involves the following steps:

1. **Complex Word Identification:** This step identifies what all words in our input sentence can be possible candidates for being complex for language learners and hence must be simplified.
2. **Substitution Generation:** The process of finding words or phrases that could replace the target complex word.
3. **Substitution Selection:** This step discards candidates that distort the meaning of the input sentence or affect its grammaticality, and retain those that fit the context.
4. **Substitution Ranking:** After generating all possible candidates for our target complex word, they are ranked based on the amount of semantic and syntactic similarity preserved by them.

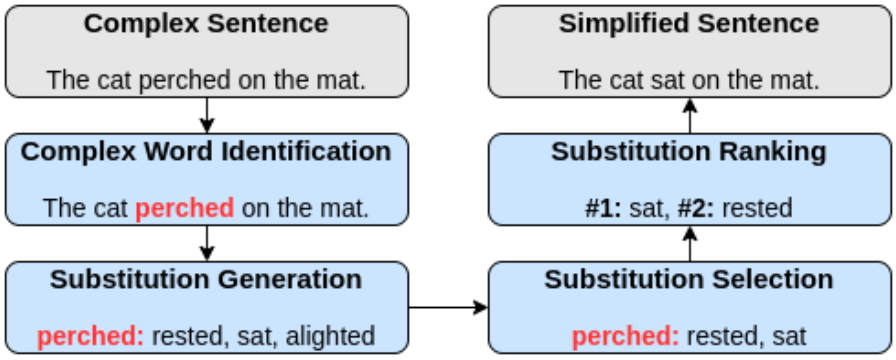

Figure 1.1: Example demonstrating steps in Lexical Simplification Pipeline.

Figure 1.1 demonstrates an example sentence passing through various LS pipeline (G. H. Paetzold and Specia 2017). The input text is '*The cat perched on the mat.*'

and the complex word identified is '*perched*'. This identified complex word '*perched*' is passed through substitution generation, selection, and ranking to simplify it, preserving semantic similarity of the input text. The output results in a non-complex simplified sentence '*The cat sat on the mat.*'.

The Complex Word Identification (CWI) is a crucial first step as inadequate identification of complex words in the text will either result in an overly difficult text if many complex words are missed, or meaning distortion with an LS system trying to unnecessarily simplify non-complex words (Shardlow, 2013a). Shardlow (2014) and G. Paetzold and Specia (2016a) have shown in their research that the performance of CWI is crucial as it considerably improves lexical simplification. They found that the most frequent errors in the LS pipeline occur due to misclassification of target words as complex or non-complex. This ensures CWI as a separate research area in itself since many popular LS systems do not focus much on it. Nation (2006) shows that to understand the content, the reader should be familiar with at least 95% of its text. Therefore, CWI is not just an extension to the LS system, but a stand-alone application within intelligent tutoring systems for second language learners or in reading devices for people with low literacy skills. A CWI system can help identify unfamiliar words and thus support providing readers other cues such as their definitions even when simpler alternatives are not available. These challenges motivate us to work on the CWI system in this thesis, targeting English language learners.

Systems performing CWI in a static manner will unable to take the context into account, thus failing to predict word complexity for the word that has different meanings with a common origin as well as words in various metaphorical or novel contexts. For instance, words like 'bat', 'bank', and 'minute' have many different meanings in the dictionary, which depends on the context where they occur. Table 1.1 demonstrates some of the examples where word meaning changed based on the context.

Sentence	**Word**	**Dictionary meaning**
He hit the ball with bat.	bat	an implement with a handle and a solid surface, typically of wood, used for hitting the ball in games
The bat is flying in night.	bat	a small animal like a mouse with wings
He deposited money in bank.	bank	a financial establishment
He sat at the bank of Elbe river.	bank	the land alongside or sloping down to a river or lake.
It took a minute to find the paper.	minute	a period of time equal to sixty seconds
There is a minute difference between 2 cars.	minute	extremely small

Table 1.1: Words with different meaning based on context

The same word 'minute', which is non-complex for language learners when used in the

context of 'time' could become complex when used to describe 'extremely small' things. A similar complexity pattern based on context has been noticed in the CWI Shared Task 2018 data (Yimam et al. 2018) for the word 'molar'. Whilst 'molar' has been annotated as complex by 17 out of 20 annotators when it is used in the sentence '*Elephants have four molars...*', none of them annotated it complex when it is used in the sentence '*... new molars emerge in the back of the mouth*'. The annotators may have found the word 'molar' simpler when it is surrounded by familiar words that imply the meaning (e.g., mouth), and found it complex when it is used in a rarer and less semantically similar co-occurrence context (e.g., elephants). These complexities only can be captured by using the CWI system that takes context into account. The current state-of-the-art SEQ system (Gooding and Kochmar 2019) based on the sequential framework by Rei (2017) is capable of taking word context into account, detecting complex words as well as phrases, eliminating extensive feature engineering by relying on word embeddings only and does not require genre-specific training. This system frames CWI as the process of identifying words that are difficult for a given target population (for example, non-native speakers of English) based on the annotation from a sample of that target population (Yimam et al. 2018).

Most research works used the *CWI shared task 2018* datasets (Yimam et al. 2018) for the CWI task. However, for my application, I needed more general, common and basic texts where second language learners would come across frequently. This requires us to collect data for Second Language Learners (L2) from the InScript (Narrative Texts Instantiating Script structure) (Modi et al. 2017) and OneStop (Vajjala and Lučić 2018) corpus. Where OneStop texts are balanced in elementary, intermediate, and advanced level texts, the InScript texts are based on ten different frequently faced scenarios by people. The details of the datasets will be elaborated in Section 3.

The CWI system trained on a huge amount of data can find latent features in the data successfully without the need for implicit feature engineering. However, when the task gets complicated and the amount of data is limited, we can still profoundly benefit from feature engineering approaches. Thus I hypothesize that the hand-engineered features along with context-dependent features might be helpful to improve the CWI model performance. I propose **hybrid model** in my thesis, which uses feature engineering along with word embedding. The set of features employed in my experiments are based on the insights from the CAMB at CWI shared task 2018 (Gooding and Kochmar 2018), which is another state-of-the-art system based on feature engineering. In this thesis, I identified thirteen different features for introducing into the sequential model. Further, I selected the most significant features needed to improve the performance of the CWI system for second language learners instead of incorporating extensive feature engineering.

Recently, the fully-connected self-attention architecture i.e., Transformer is broadly adopted in various natural language processing (NLP) tasks such as sequence classification, token classification, or question answering owing to its parallelism and advantage in modeling the long-range context. While transformer models achieve state-of-the-art performances in numerous NLP tasks, I observe a lack of research in the area of CWI. In this thesis, I also present an empirical study of the performance of CWI approaches derived from a pre-trained RoBERTa and BERT model.

Specifically, I will address the following research questions:

- **RQ1:** Do the Hybrid approach outperforms the sequential model for the CWI task?
- **RQ2:** Do the transformer-based models outperforms the sequential model for the CWI task?
- **RQ3:** Are the models trained on CWI Shared Task data are good enough for second language learners?

Further, in this thesis, I overview the related work in this field in Chapter 2. In Chapter 3, I have explained the data and their statistics that I am going to use for the model training and evaluation. Section 3.1 demonstrates the technique used to annotate L2 data. Chapter 4 shows the data format I used for the various models. Chapter 5 presents the evaluation metric used for the model evaluation. Chapter 6 presents the experimental setup of the sequential model for CWI. In Section 6.1, I explained the architecture of the sequential model, which I adapted to replicate the current state-of-the-art results for the CWI task. I used the same architecture to replicate the results and then converted it to a hybrid model by introducing features. In Section 6.3, I performed a pilot experiment to replicate the original results for CWI shared task 2018 testsets as well as checking model performance on my L2 testsets. I also presented the model performance by using more sophisticated FastText embeddings instead of GloVe embeddings from the original paper. Section 6.5 shows the features I am considering to introduce in the sequential model, making it hybrid architecture 6.4. In Section 6.6, I detailed all the features I adopted and their importance in the sequential model for CWI. The last Section 6.7 under Chapter 6, provides my feature selection approach and evaluates the performance of the hybrid model using selected features. In Chapter 7, I fine-tuned the transformer models RoBERTa and BERT for the CWI task and presented the performance on shared task 2018 and L2 testsets. Section 7.1 presents the three different sets of training data to be trained on transformer models. In Chapter 8, I discussed my various approaches for CWI and key findings. Finally, I provided my conclusion and outline future directions for this research in Chapter 9.

2 Related Work

In this chapter, I will discuss the previous work done for the CWI using different Shared Task datasets. In Section 2.1 I will discuss the very first dataset made available for the CWI task and the various system approaches for CWI. Section 2.2 and Section 2.3 will discuss about the Shared Task 2018 and SemEval-2021 Shared Task for CWI respectively.

2.1 SemEval-2016 Shared Task on CWI

CWI has often been regarded as a crucial first step for automatic lexical simplification (Shardlow 2014). Although lexical simplification methods have been proposed for more than a decade (Petersen and Ostendorf 2007), CWI has not been studied as a separate standalone task (Shardlow, 2013a). The SemEval-2016 shared task on CWI (G. Paetzold and Specia, 2016b) was the first evaluation campaign that provided a gold-standard dataset as well as an extensive comparison of different machine learning approaches for the task. The gold-standard dataset combines the data from the CW corpus of Shardlow (2013b), the LexMTurk corpus of Horn et al. (2014) and the Simple Wikipedia corpus of Kauchak (2013), all of which rely on Simple Wikipedia data. The dataset was annotated as complex or simple by a set of 400 non-native speakers.

The SemEval-2016 shared task featured 42 systems based on different types of classifiers and using different types of features, ranging from linguistic information like lexical, morphological, semantic and syntactic level features, over psycholinguistic measures to corpus-based information such as word frequencies. The most popular classifier among the top participants was Random Forest (Brooke et al. (2016); Mukherjee et al. (2016); Ronzano et al. (2016); Zampieri et al. (2016)), while the most common type of features were lexical and semantic features (Brooke et al. (2016); Mukherjee et al. (2016); Ronzano et al. (2016); G. Paetzold and Specia (2016c); Quijada and Medero (2016)). Along with Random Forest classifier some participant used Naive Bayes (Mukherjee et al. 2016) or SVM (Zampieri et al. 2016). Many other classification methods has been used by participants like Maximum Entropy (Konkol 2016), Decision Trees (Quijada and Medero 2016) or Nearest Centroid (Palakurthi and Mamidi 2016). The results on the shared task showed how ensemble methods (G. Paetzold and Specia 2016c) trained on morphological, lexical, and semantic features outperformed any other ML technique and neural approaches in particular (Bingel et al. 2016). Also, the results showed that simpler features based on word frequency (Konkol (2016), Wróbel (2016), and Zampieri et al. (2016)) and word presence in certain lexicons (Mukherjee et al. (2016) and Wróbel (2016)), work best.

2.2 Shared Task 2018 on CWI

The CWI 2018 shared task (Yimam et al. 2018) used the data from News, Wikinews, and Wikipedia articles, derived from the dataset of Yimam et al. (2017). In CWI 2018, a multilingual dataset was made available containing English, German, and Spanish training and testing data for monolingual tracks, and a French testset for multilingual predictions. They approach CWI from two perspectives: under the *binary* setting, a word can be either simple or complex, while in the *probabilistic* setting a word receives a probability score reflecting the proportion of annotators that consider the word complex. The dataset includes annotation for content words as well as for phrases. The most recent research on CWI uses the data from the CWI 2018 shared task. AbuRa'ed and Saggion (2018) evaluated the performance of five classification algorithms on the English part of the dataset: Support Vector Machine (with linear and radial basis function kernels), Naive Bayes, Logistic Regression, Random Tree, and Random Forest. They designed two systems based on binary classifiers, one represents the context as word embedding vectors, and the other uses a set of lexical, semantic, and contextual features. Random Forest classifier performed best over the whole dataset for both systems. Kajiwara and Komachi (2018) TMU system demonstrates the usefulness of a learner corpus for the CWI task proving the word frequency counted from the Lang-8 learner corpus (Mizumoto et al. 2011) worked better than that from the in-domain corpus written by the native speakers. They use random forest classifiers and regressors with eight features including the number of characters and words and the frequency of target words in various corpora.

CAMB system (Gooding and Kochmar 2018) was the winning system submitted to the CWI Shared Task 2018. This system considers words irrespective of their context and relies on 27 features of various types, encoding lexical, syntactic, frequency-based, and other types of information about individual words. They incorporated Random Forests and AdaBoost with 5000 estimators for binary classification while Linear Regression algorithm for probabilistic. CAMB system incorporated features based on the insights from the CWI shared task 2016 (G. Paetzold and Specia 2016c) in addition to the number of words grammatically related to the target; a range of psycholinguistic features from the MRC Psycholinguistic Database (Wilson 1988); CEFR levels (Council of Europe 2011); and the use of Google N-gram word frequencies. However, the CAMB system is genre-dependent as the performance of the classifier varies according to the genre of data leading to the choice of the features, algorithm, and training data depending on the genre. Gooding and Kochmar (2019) presented a novel approach to CWI based on the sequence modeling (SEQ system), which takes the context into account. SEQ system eliminates the need for extensive feature engineering by relying on word embeddings only, due to which it does not require genre-specific training and represents a one-model-fits-all approach. SEQ system uses the sequential architecture by Rei (2017), which has achieved state-of-the-art results on several NLP tasks. SEQ model benefits from the use of bi-directional long short-term memory (BiLSTM) (Hochreiter and Schmidhuber 1997), as these units can capture the long-term contextual dependencies in natural language and able to consider both the left and right contexts of a word. The SEQ system also outperformed the CAMB system submitted to the CWI Shared Task 2018.

2.3 SemEval-2021 Shared Task on CWI

Previous efforts (G. Paetzold and Specia (2016b); Yimam et al. (2018)) have focused on framing this as a binary classification task, which might not be ideal, since a word close to the decision boundary is assumed to be just as complex as one further away (Shardlow et al. 2020). CompLex (Shardlow et al. 2020) is the most recent dataset used for the Lexical Complexity Prediction (LCP) shared task at SemEval-2021 (Shardlow et al. 2021), which formulates CWI task as a regression task. It is an English multi-domain corpus in which words and multi-word expressions (bigrams) were annotated with respect to their complexity using a five-point Likert scale, depending on whether it seemed more or less easy to understand in the context. Each instance has been annotated multiple times and authors have taken the mean average of these annotations as the label for each data instance. To add variation in the data, authors selected English text at almost equal proportions from three sources: the Bible (Christodouloupoulos and Steedman 2015), Europarl (Koehn 2005), and Biomedical (Bada et al. 2012) texts. SemEval-2021 task featured two Sub-tasks: Subtask 1 focused on single words and Subtask 2 focused on multi-word expressions. Three main types of systems that were submitted to this task were Feature-based systems, Deep Learning Systems, and Systems which use a hybrid of the former two approaches.

JUST-BLUE system (Bani Yaseen et al. 2021) makes use of an ensemble of BERT (Devlin et al. 2018) and RoBERTa (Liu et al. 2019) and attained the highest Pearson's Correlation for Subtask 1. Authors fine-tuned BERT and RoBERTa models with the 'token' and the 'complexity' label to be trained. As a second strategy, they also inserted 'sentence' and 'complexity' columns to both models. The results have been combined using an ensembling voting method with weighted averaging. To calculate the degree of complexity for a single word, authors consider 80% voting from the 'token' model and 20% voting from the 'sentence' model as the complexity of the word is affected by the complexity of the sentence. DeepBlueAI system by Song et al. (2021) has achieved the best Pearson's Correlation for Subtask 2 and was runner-up for Subtask 1. This system used an ensemble of pre-trained language models RoBERTa and ALBERT (Lan et al. 2019), which were fine-tuned with various hyperparameters and different training strategies. The final prediction has been provided by applying a stacking mechanism on top of the fine-tuned pre-trained language models.

Deep Learning based systems have attained the highest Pearson's Correlation on SemEval-2021 task, however, feature-based systems were not far behind and have been placed in the third and fourth spots on Subtask 1. Mosquera (2021) used a feature-based approach, incorporating 51 features based on length, frequency, semantic features from WordNet (Miller 1995) and sentence level readability features. These features were passed through a Light-GBM (Ke et al. 2017) implementation of gradient tree boosting to give the final output score. While examining feature importance, it has been observed that several sentence readability features are being identified as top contributors. Rotaru (2021) combined traditional feature-based approach with features from pre-trained language models. They use psycholinguistic features, context-independent features: Skip-gram (Pennington et al. 2014), Word2Vec (Mikolov et al. 2013) and Con-

ceptNet NumberBatch (Speer et al. 2016) embeddings and context-dependent features from an ensemble of pre-trained language models: BERT, RoBERTa, ELECTRA (Clark et al. 2020), ALBERT, DeBERTa (He et al. 2020). All these three types of features are passed through gradient boosted regression to give the final output score.

Taya et al. (2021) proposed an ensemble of pre-trained language models RoBERTa and BERT along with various training strategies to boost performance. They also added hand-crafted features during training to enrich contextual representations. As a feature, word frequency information of the target word is introduced that has been computed from the log frequency values using the Wiki40B corpus (Guo et al. 2020). Hybrid approaches use a mixture of deep learning by fine-tuning a neural network alongside feature-based approaches. The hand-crafted features could be concatenated directly to the input embeddings, or at the output prior to further training. This strategy appears to be the best of both worlds, uniting linguistic knowledge with the power of pre-trained language models, however, the hybrid systems do not tend to perform as well as either feature-based or deep learning systems (Shardlow et al. 2021).

In this thesis, I use the CWI Shared Task 2018 English data, which contains annotation for both words and phrases, and represents three different genres of text. I focus on the binary setting, i.e., complex or non-complex, and compare my results to the previous state-of-the-art system. I have also incorporated transformer-based pre-trained language models for this data, which have achieved state-of-the-art results in the newest shared task data SemEval-2021. The hybrid approach has also been utilized for the SemEval-2021 task, however, they did not perform better than the feature-based and deep learning based systems, and ranked seventh in the CWI task.

3 Datasets

For my experiment, I considered CWI Shared Task 2018 English dataset (Yimam et al. 2018), which provides binary judgment of a word's complexity, i.e, whether a word is complex or not? Whilst SemEval-2016 dataset (G. Paetzold and Specia, 2016b) relied only on Simple Wikipedia, the Shared Task 2018 dataset uses texts from 3 different genres: professionally written news articles (NEWS), amateurishly written news articles (WIKINEWS), and WIKIPEDIA articles, making it an optimum choice for language learners. Also, I used one of the models, i.e., the sequential model from Shared Task 2018 as the baseline, therefore Shared Task 2018 seems to be the most logical choice.

The dataset includes annotation for a selected set of content words as well as for phrases, which is provided alongside the full sentence and the word span. Table 3.1 presents the statistics on the number of words (w) and phrases (ph) annotated in the training (train), development (dev), and test subsets of datasets. The annotation for the English data is collected from 10 native and 10 non-native speakers of English using the Amazon Mechanical Turk platform. The annotation contains both binary (bin) and probabilistic (prob) labels. In binary labels, words and phrases are annotated complex if at least one of the 20 annotators annotated them as complex, otherwise simple. However, in probabilistic labels, words and phrases receive a label in the range between [0.0, ..., 1.0] depending on the number of annotators who marked them complex.

Dataset	**Train**	**Test**	**Dev**
News (w)	11,949	1,502	1,813
News (ph)	2,053	262	282
WikiNew (w)	6,780	776	1,138
WikiNew (ph)	966	94	149
Wikipedia (w)	4,833	606	750
Wikipedia (ph)	718	88	120

Table 3.1: Number of words (w) and phrases (ph) annotated in CWI shared task datasets.

In Table 3.2, I have shown the Shared Task 2018 English dataset tokens distribution for training, validation, and various test sets. This dataset has most of the data proportion from the News genre and least from the Wikipedia genre. Also, it is very clear for this table that every set has almost five times more non-complex words than complex words. Every set has more than 80 % of words marked as non-complex making it imbalance. The graphical representation of the distribution of tokens in Shared Task 2018 testsets

is presented in Figure 3.1.

Dataset	#Sen	#tokens	#C	#N	N (percentage)
Full Train	2,063	52,476	8,408	44,068	83.98
Full Val	266	6,627	1,049	5,578	84.17
News Test	177	4,068	576	3,492	85.84
WikiNews Test	105	2,454	440	2,014	82.07
Wikipedia Test	61	1,775	353	1,422	80.11

Table 3.2: Total number of sentence and token distribution in Shared Task 2018 data.

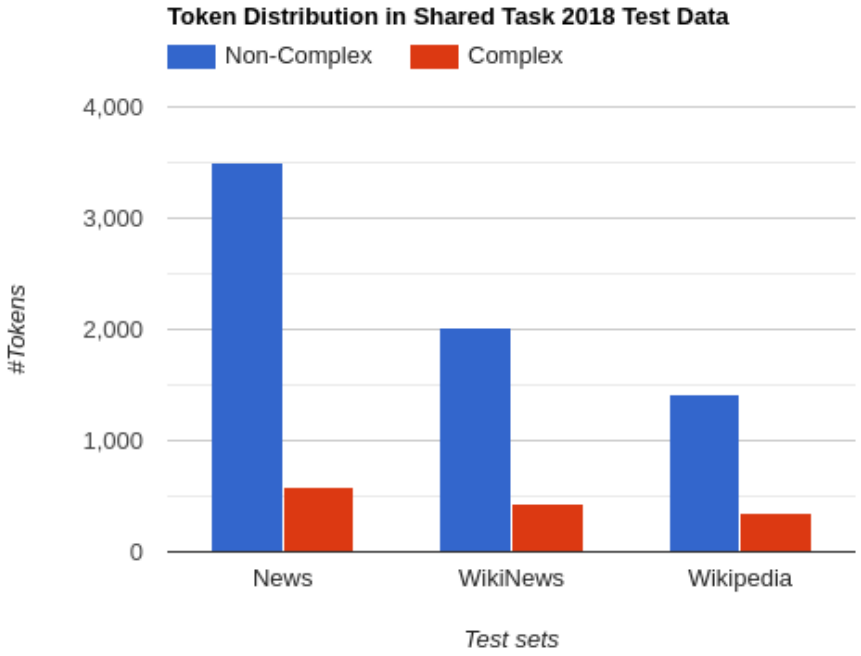

Figure 3.1: Distribution of complex and non-complex words in Shared Task 2018 testsets.

For my experiment, I further need an English dataset that is annotated with complex words at the same time frequently used by language learners, which makes us select my L2 text from InScript (Modi et al. 2017) and OneStopEnglish (Vajjala and Lučić 2018) corpus. InScript is a corpus of simple narrative texts in the form of stories, wherein each story is centered around a specific scenario from 10 different scenarios. The 10 different scenarios used in InScript corpus are bath, bicycle, bus, cake, flight, grocery, haircut, library, train, and tree. The stories have been collected via Amazon Mechanical Turk, asking the participants to describe a scenario in form of a story as if explaining it to a

child. They collected 100 stories per scenario, giving a total of 1,000 stories with about 200,000 words. Table 3.3 presents the data statistics for different scenarios of InScript data.

Category	**#tokens**	**#sentences**	**max_sen_len**
bath	4,016	261	42
bicycle	3,783	256	38
bus	3,918	258	38
cake	3,950	278	32
flight	4,120	254	39
grocery	4,175	274	48
haircut	3,830	256	40
library	3,980	248	39
train	3,833	258	33
tree	3,729	237	36
Total/Max	39,334	2,580	48

Table 3.3: Number of tokens, sentences and maximum sentence length of InScript data.

The OneStopEnglish corpus was compiled from onestopenglish.com, which is the English language learning resources website. The website contains articles sourced from The Guardian newspaper, which has been collected over the period 2013–2016 and rewritten by teachers to suit three levels of English as Second Language learners: Elementary, Intermediate, and Advanced. The advanced version is close to the original article, although not with the exact same content. I selected OneStop data that are balanced in 3 categories with 33 paragraphs for each level. Table 3.4 presents the statistics as the number of tokens, number of sentences, and maximum length of the sentence in different levels of the OneStop data.

Category	**#tokens**	**#sentences**	**max_sen_len**
advanced	30,787	1,364	79
elementary	20,136	1,125	72
intermediate	25,190	1,231	62
Total/Max	76,113	3,720	79

Table 3.4: Number of tokens, sentences and maximum sentence length of OneStop data.

3.1 L2 Data Annotation

For my experiment, I annotated all the words in L2 data utilizing Common European Framework of Reference for Languages (CEFR) (Council of Europe 2011) levels. The CEFR describes six broad levels of ability, with A1 being the lowest and C2 the highest. Learners are classified into three distinct groups: the Basic User (levels A1 and A2), the Independent User (B1 and B2), and the Proficient User (C1 and C2). The English Vocabulary Profile (EVP) (https://www.englishprofile.org/wordlists/evp) offers reliable information about which words (and importantly, which meanings of those words) and phrases are known and used by learners at each level. Cambridge University Press is making the A1-C2 EVP available free of charge to teachers and educationalists around the world. A single word can exist in different CEFR levels based on some guidance, the topic where it is used, and part of speech. For instance, the word 'minute' is assigned level A1 when it is used in the context of time and it is also assigned level C2 when it is used in the context of describing small things. For simplicity, I annotated words as complex only if they exist above B1 level, else non-complex. However, this would not ensure the word complexity on the basis of context at all. Also, the words which could be complex for language learners but not present in the EVP were annotated as non-complex using this technique (like 'cyclones').

Dataset	**#tokens**	**#C**	**#N**	**N (percentage)**
InScript	39,334	947	38,387	97.72
OneStop	76,113	4,311	71,802	94.56

Table 3.5: Token distribution in L2 data.

Table 3.5 shows the distribution of tokens in the InScript and OneStop data. Figure 3.2 shows the graphical representation of the distribution of tokens in L2 data. The total number of words marked as complex using the CEFR level technique for the InScript and OneStop data are 947 and $4,311$ respectively. OneStop data have almost twice the number of tokens as InScript, however, it has almost four times more complex words than InScript. This makes sense as OneStop data are based on English language complexity categories including advanced English. The L2 data have a very high proportion of non-complex words, which makes sense due to my annotation technique. As these L2 data are not annotated by some language learners, I consider this data only as the testset to compare the performance of models in totally unseen data. Table 3.6, shows some example texts from these L2 data with bold words indicating complex words.

Initially, I pass my L2 data (annotated based on CEFR levels) on the pre-trained SEQ model from Gooding and Kochmar (2019) on the English part of the CWI datasets from Yimam et al. (2018) for identifying complex words in it. However, the results achieved in my text did not seem to be promising. It has been found that most of the non-complex nouns (whales, ducks, tub, bathtub, bubble bath, soaps, shower, temperature, oils, salts, scent, sunscreen, bathrobe, lavender, scented, gym, clothes, shampoo, conditioner, taps,

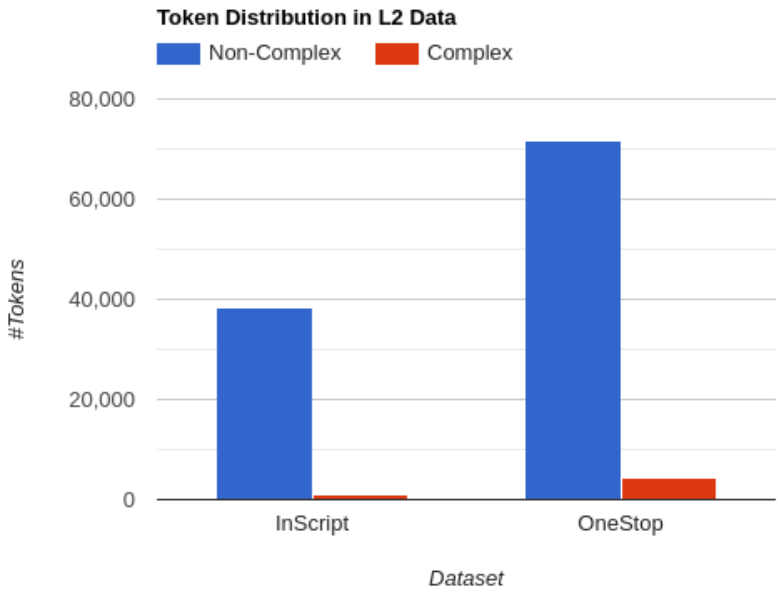

Figure 3.2: Distribution of complex and non-complex words in L2 data.

etc.) in my L2 data are annotated as complex by this system. It has also been seen that some words, which seem to be complex according to CEFR levels such as 'squeaky' (CEFR level C2) are not identified as complex by the system. This ensures that the vocabulary complexity for shared task data sets versus L2 data sets is different and for the L2 CWI application, I need to train the model using human-annotated data similar to my L2 data.

'After I get **squeaky** clean' - bold identified as non-complex

'The bottom of the **bath tub**.' - bold identified as complex

'Fun **toys** for the **bathtub** are boats and **plastic** water animals like **whales** and **ducks**.' - bold identified as complex

L2 Data	Text
InScript (Bath)	After going to the gym, I knew I needed to take a bath. I went to the bathroom and turned the faucet on in the tub while plugging the **drain** to fill it. When the water was at a good level I shut the water off. I took off all my dirty gym clothes. I then made sure the temperature of the water was nice and warm before I slowly got inside.
InSCript (Train)	Today I am going to ride a train to visit my grandmother. We are going to visit for my birthday. We left our home and drove in our car to the train station. We then went up to the **counter** and bought two tickets to ride the train to my grandmother's town. We had to wait for twenty minutes in the station before it was time to board our train. When it was time, we went over to the train and handed our tickets to a man in a uniform that works on the train. He checked our tickets and helped us up onto the train.
OneStop (Intermediate)	False memories are a **major** problem with **witness** statements in courts of law. **Evidence** that eyewitnesses give often leads to guilty verdicts, but later the convictions are overturned when DNA or some other **evidence** is used. Susumu Tonagawa, a neuroscientist at the Massachusetts **Institute** of Technology (MIT), and his team wanted to study how these false memories might form in the human brain, so they encoded memories in the brains of mice by manipulating individual neurons.
OneStop (Advanced)	Scientists have connected the brains of a pair of animals and allowed them to share sensory information in a **major** step towards what the researchers call the worlds first **organic** computer.The US team fitted two rats with devices called brain-to-brain interfaces that let the animals **collaborate** on simple tasks to earn rewards, such as a drink of water. In one **radical demonstration** of the technology, the scientists used the internet to link the brains of two rats separated by **thousands** of **miles**, with one in the researchers lab at Duke University.

Table 3.6: Text examples from L2 Data (bold words are complex).

4 Data Setup

Shared Task 2018 has data in the format as shown in Figure 4.1. The second column in the format shows the actual sentence and the fifth column represents the target word. The gold-standard label for the corresponding target word is present in the tenth and eleventh columns. The tenth column shows the label for binary classification tasks while the eleventh column is for probabilistic classification tasks.

```
<ID> Both China and the Philippines flexed their muscles on Wednesday. 31 51 flexed their muscles 10 10 3 2 1 0.25
<ID> Both China and the Philippines flexed their muscles on Wednesday. 31 37 flexed 10 10 2 6 1 0.4
<ID> Both China and the Philippines flexed their muscles on Wednesday. 44 51 muscles 10 10 0 0 0 0.0
```

Figure 4.1: CWI Shared Task 2018 data format.

The sequential framework by Rei (2017), which I adopted as baseline needed the data in a format with complete word context as an input. This makes us tokenize the sentences and includes the corresponding annotation for each word token, using C for the annotated complex words otherwise, N, resulting in the following format:

The N
barren C
islands N
. N

Spanish N
lenders C
...

The data results in standard CoNLL-type tab-separated format. The file format is one word per line, a separate column for token and label, and an empty line between sentences. The empty line in this format lets the model know the end of the sentence. There can be other columns in the middle of the token and label, which I utilized to introduce hand-engineered features. I also converted L2 data in the same format.

5 Evaluation Metrics

The evaluation metric reported is the macro-averaged F1, as it was used in the 2018 CWI shared task (Yimam et al. 2018). Also, the same metric has been reported by the SEQ model by Gooding and Kochmar (2019), which demonstrates state-of-the-art results for CWI shared task 2018 dataset.

To evaluate model performance comprehensively, we should examine both precision and recall. The precision is the ratio of correctly predicted positive observations to the total predicted positive observations 5.1. This metric is used to measure how many words, which are labeled as complex are actually complex. High precision is related to the low false-positive rate, i.e., falsely classified as complex. The recall is the ratio of correctly predicted positive observations to all observations in the actual class 5.2. This metric is used to measure how many words, which are actually complex are labelled as complex. The high recall is related to the low false-negative rate, i.e., falsely classified as non-complex.

$$Precision = (TruePositive/(TruePositive + FalsePositive)) \tag{5.1}$$

$$Recall = (TruePositive/(TruePositive + FalseNegative)) \tag{5.2}$$

The F1 score 5.3 serves as a helpful metric that considers both of them by considering harmonic mean of precision and recall for a more balanced summarization of model performance. The macro-averaged F1 score is computed by taking the arithmetic mean of all the per-class F1 scores. In my application, I have two classes for the target words named complex and non-complex. Equation 5.4 demonstrates the macro-averaged F1 score computation for binary classification. For instance, if F1 score for *complex* and *non-complex* class are 0.80 and 0.90 respectively then the macro-averaged F1 score will be 0.85.

$$F1 = ((2 * precision * recall)/(precision + recall)) \tag{5.3}$$

$$MacroF1 = ((F1_{complex} + F1_{non-complex})/2) \tag{5.4}$$

In Chapter 3, I have also shown that shared task 2018 data sets have imbalanced classes, using the macro average would be a good choice as it treats all classes equally. It will allow us to compare the performance achieved by training the sequential model from scratch to replicate original results. This evaluation metric will also allow us to illustrate whether the hybrid model (sequential model introduced with features) technique or fine-tuning transformers, improves the performance over the SEQ system. I will also present precision and recall metrics to improve the recall of the model. This step is taken to make sure that the model reduces the number of errors while identifying complex words.

6 Sequential Model for CWI

6.1 Sequential Architecture

As the baseline architecture (Figure 6.1) for my CWI system, I use the sequence labelling architecture of Rei (2017). This sequential model framework has the ability to infer representations for previously unseen words and to share information about morpheme-level regularities. Gooding and Kochmar (2019) has used this framework to build their *SEQ* system and provided the state-of-the-art results in CWI Shared Task 2018. Within this framework, the model receives as input a sequence of tokens $(w_1, w_2, ..., w_T)$ and predicts a label for each token using a bidirectional LSTM. The input tokens are first mapped to a sequence of word embeddings $(x_1, x_2, ..., x_T)$. The model use 300-dimensional GloVe embeddings as word representations (Pennington et al. 2014). To construct context-dependent representations for every word, two LSTM (Hochreiter and Schmidhuber 1997) moving in opposite directions through the sentence are used. To obtain a context-specific representation for each word in the sentence, the hidden representation $(h_1, h_2, ..., h_T)$ from both directional LSTM are concatenated.

$$\overrightarrow{h_t} = LSTM(x_t, \overrightarrow{h_{t-1}})$$

$$\overleftarrow{h_t} = LSTM(x_t, \overleftarrow{h_{t+1}})$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]$$

Finally, the concatenated representation is passed through a feedforward layer $(d_1, d_2, ..., d_T)$ allowing the model to learn from both context directions.

$$d_t = tanh(W_d h_t)$$

In the above equation, W_d is a weight matrix and $tanh$ is used as the non-linear activation function. The architecture use softmax to predict the label (Complex or Non-Complex) for each token.

To handle previously unseen words, while taking full advantage of the word embeddings, the framework also makes use of the character-level component. The individual characters of a word are mapped to character embeddings and passed through a bidirectional LSTM. The hidden representation from both directions is concatenated before passing through the non-linear layer. Further, this resulting vector representation is combined with a regular word embedding using a dynamic weighting mechanism allowing the model to learn character-based patterns.

Instead of only using F1-Score 5.3, Rei (2017) has defined an additional metric F05-Score 6.1 to define their best model and saving it. F05-Score is the weighted harmonic

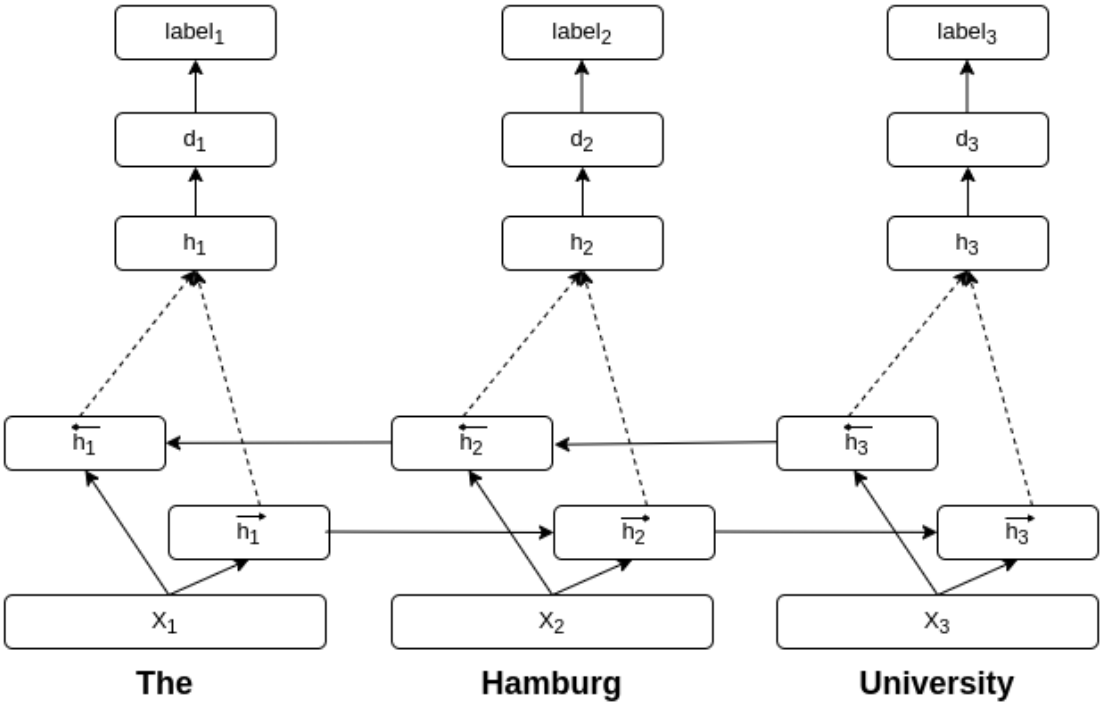

Figure 6.1: The unfolded network structure for a sequence labeling model with an additional language modeling objective, performing CWI on the sentence "The Hamburg University". The input tokens are shown at the bottom, the expected output labels are at the top. The arrows above variables indicate the directionality of the Bi-LSTM.

mean of the precision and recall, giving more weight to the precision. If the F05-score of development data is better from the previous epochs, the model is saved and if the performance did not improve for the next continuous 7 epochs the model training stopped. Also, they keep decreasing the learning rate at the factor of 0.9 starting from 1, if the model results did not improve for the 3 or more epochs continuously. They used a batch size of 32 and set the maximum number of epochs to 200.

$$F05 = ((1.25 * precision * recall)/((0.25 * precision) + recall)) \tag{6.1}$$

6.2 Approach

I will consider sequence labeling architecture by Rei (2017) as a baseline architecture and further do some changes on it to improve the CWI performance. Firstly, I will try to replicate the original results from the SEQ model (Gooding and Kochmar 2019) and set them as baseline results for further improvement. I will incorporate FastText (Bojanowski et al. 2016) embeddings instead of GloVe (Pennington et al. 2014) to deal with unseen

and rare words and better generalization from the model. Also, I will update the F05-Score 6.1, used by the SEQ model to improve the recall. This change will make my application more robust towards language learners as better recall results in less error in identifying the complex words. Later, I will incorporate many hand-engineered features (Section 6.5) useful for CWI and introduce them in the sequential model, resulting in a hybrid model. All the results showed in this thesis for various experiments are taken from the average of three experiments.

6.3 The Pilot Experiment

In this experiment, I tried to replicate the original results from Gooding and Kochmar (2019) using the same model framework, hyperparameters, and embeddings. I trained the sequential model using training datasets from News, WikiNews, and Wikipedia and tested it on each test data individually. Table 6.1 displays the results achieved by this experiment. The performance achieved for the InScript and OneStop data (L2 data) is very low in comparison to News, WikiNews, and Wikipedia test data (Shared Task 2018 test data). This makes sense as the vocabulary complexity of L2 data is different from shared task data, and I did not consider the L2 data for model training. For the rest of the experiment, I will consider the results achieved by this model ($SEQ_{baseline}$) as baseline results.

Testset	Precision	Recall	F-score
News	0.8945	0.8495	0.8694
WikiNews	0.9013	0.8236	0.8544
Wikipedia	0.8805	0.8097	0.8369
InScript	0.5610	0.6612	0.5815
OneStop	0.6064	0.6986	0.6317

Table 6.1: Precision, recall and macro F-score achieved for different testsets.

Table 6.2 compares the macro F-score achieved by $SEQ_{baseline}$ to the results published by original paper (Gooding and Kochmar 2019). The performance achieved for News and WikiNews testsets is almost similar to the published SEQ system. However, for Wikipedia testsets I have achieved a better F-Score by 2.29%. The reason could be either the way the SEQ system was built using sequence labeling architecture by Rei (2017) or the way they formatted the system accepted file from Shared Task 2018 data.

6.3.1 Word Embeddings

In this experiment, I tried to improve the model performance by incorporating FastText (Bojanowski et al. 2016) embeddings developed by Facebook instead of GloVe embeddings. GloVe builds word embeddings in a way that a combination of word vectors relates

Testset	SEQ	$\mathbf{SEQ_{baseline}}$
News	0.8763	0.8694 (-0.69)
WikiNews	0.8540	0.8544 (+0.04)
Wikipedia	0.8140	0.8369 (+2.29)

Table 6.2: Comparison of macro F-scores achieved by $SEQ_{baseline}$ to the state-of-the-art SEQ system.

directly to the probability of these words co-occurrence in the corpus. Its embeddings can be interpreted as a summary of the training corpus with low dimensionality that reflects co-occurrences. However, there was one unsolved problem: generalization to unknown words. FastText overcome this problem by going one level deeper instead of using only words to build word embeddings. FastText improves embeddings by taking word parts into account as well. This approach has two advantages: (1) generalization is possible as long as new words have the same characters as known ones (2) less training data is needed since much more information can be extracted from each piece of text.

The advantages of FastText above GloVe word embeddings are useful for my experiment as I am determined to improve performance over the L2 testset while training the model using the training set from CWI Shared Task 2018. In my experiment, I used the pre-trained English word vectors from FastText, which contains 2 million word vectors with subword information on Common Crawl (600B tokens). Table 6.3 displays the results achieved by training the sequential model using FastText embeddings ($SEQ_{FastText}$ model).

Testset	Precision	Recall	F-score
News	0.9055	0.8724	0.8876
WikiNews	0.9019	0.8349	0.8626
Wikipedia	0.8918	0.8213	0.8497
InScript	0.5664	0.6614	0.5890
OneStop	0.6216	0.7252	0.6512

Table 6.3: Precision, recall and macro F-score achieved for different testsets using FastText embeddings

In Table 6.4, I compare the results achieved from $SEQ_{baseline}$ model (GloVe embeddings) to $SEQ_{FastText}$ model (fastText embeddings). Incorporating FastText embeddings has increased the performance of sequential model noticeably in all testsets.

Testset	Precision		Recall		F-Score	
	$\mathbf{SEQ_{baseline}}$	$\mathbf{SEQ_{FastText}}$	$\mathbf{SEQ_{baseline}}$	$\mathbf{SEQ_{FastText}}$	$\mathbf{SEQ_{baseline}}$	$\mathbf{SEQ_{FastText}}$
News	0.8945	0.9055	0.8495	0.8724	0.8694	0.8876
WikiNews	0.9013	0.9019	0.8236	0.8349	0.8544	0.8626
Wikipedia	0.8805	0.8918	0.8097	0.8213	0.8369	0.8497
InScript	0.5610	0.5664	0.6612	0.6614	0.5815	0.5890
OneStop	0.6064	0.6216	0.6986	0.7252	0.6317	0.6512

Table 6.4: Comparison of results achieved from $SEQ_{baseline}$ to $SEQ_{FastText}$

6.3.2 Recall-oriented Learning

In this section, I tried to improve the recall of the $SEQ_{FastText}$ model without compromising the overall performance. Considering some applications utilizing the CWI system like language learners application that chooses and displays the most relevant images for the complex words, it does not matter if the system misidentifies some non-complex words as complex (low precision). However, I do not want to miss complex words to be identified as complex, so that language learners can understand the context easily.

For improving recall, I updated the metric F05-Score 6.1 introduced by Rei (2017) in sequential framework to give more weight to the recall instead of the precision. The resulting F05-Score is displayed in Equation 6.2.

$$F05 = ((1.25 * precision * recall)/(precision + (0.25 * recall))) \tag{6.2}$$

Further, I trained the model using these changes (SEQ_{FTRec} model), and the results achieved are displayed in Table 6.5.

Testset	Precision	Recall	F-score
News	0.8750	0.9140	0.8929
WikiNews	0.8737	0.8870	0.8801
Wikipedia	0.8679	0.8720	0.8699
InScript	0.5688	0.7434	0.5945
OneStop	0.6135	0.7694	0.6444

Table 6.5: Precision, recall and macro F-score achieved for different testsets while improving recall.

In Table 6.6, I compare the model results achieved from $SEQ_{FastText}$ to the SEQ_{FTRec}, where I tried to improve recall. The comparison shows that the recall for each testset is improved by at least 4% with this experiment. However, there is always a trade-off between precision and recall, which has been reflected here as well. But as the model

was trained on more non-complex words than complex words, the precision drop is very less (approx. 2.5%) in comparison to recall growth, improving the macro F-score overall.

Testset	Precision		Recall		F-Score	
	SEQ$_{\text{FastText}}$	**SEQ$_{\text{FTRec}}$**	**SEQ$_{\text{FastText}}$**	**SEQ$_{\text{FTRec}}$**	**SEQ$_{\text{FastText}}$**	**SEQ$_{\text{FTRec}}$**
News	0.9055	0.8750	0.8724	0.9140	0.8876	0.8929
WikiNews	0.9019	0.8737	0.8349	0.8870	0.8626	0.8801
Wikipedia	0.8918	0.8679	0.8213	0.8720	0.8497	0.8699
InScript	0.5664	0.5688	0.6614	0.7434	0.5890	0.5945
OneStop	0.6216	0.6135	0.7252	0.7694	0.6512	0.6444

Table 6.6: Comparison of results achieved from $SEQ_{FastText}$ model to SEQ_{FTRec} model.

The best results were achieved for CWI using the SEQ_{FTRec} model, which incorporated the FatText embeddings as well as tuned to improve the recall. In further sections, I will try to introduce hand-engineered features in this model.

6.4 Hybrid Architecture

I propose the Hybrid approach for my CWI sequence system by including hand-engineered features along with context-dependent features (word embeddings) in the sequence model. I explicitly added hand-engineered features to the sequential architecture (see Figure 6.1), resulting in hybrid architecture displayed in the Figure 6.2. The idea behind building this architecture is that I could provide some external features useful for CWI that the network could not learn on its own, like word frequencies in some learning corpus.

I achieve a hybrid model by concatenating the features $(f_1, f_2, ..., f_T)$ from the sequence of words, to the concatenated hidden representation $(h_1, h_2, ..., h_T)$ from both directional LSTM, before passing through a feedforward layer $(d_1, d_2, ..., d_T)$.

$$\overrightarrow{h_t} = LSTM(x_t, \overrightarrow{h_{t-1}})$$

$$\overleftarrow{h_t} = LSTM(x_t, \overleftarrow{h_{t+1}})$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}; \mathbf{f_t}]$$

$$d_t = tanh(W_d h_t)$$

6.5 Features for CWI

We can profoundly benefit from feature engineering approaches when some NLP task is complicated or the amount of data is limited. In my case, I have limited training data for specific genres. Also, some features like target word frequency in most common English words or some second language learner corpus can introduce external information that the model can not learn from the training data. With this idea, I believed that adding

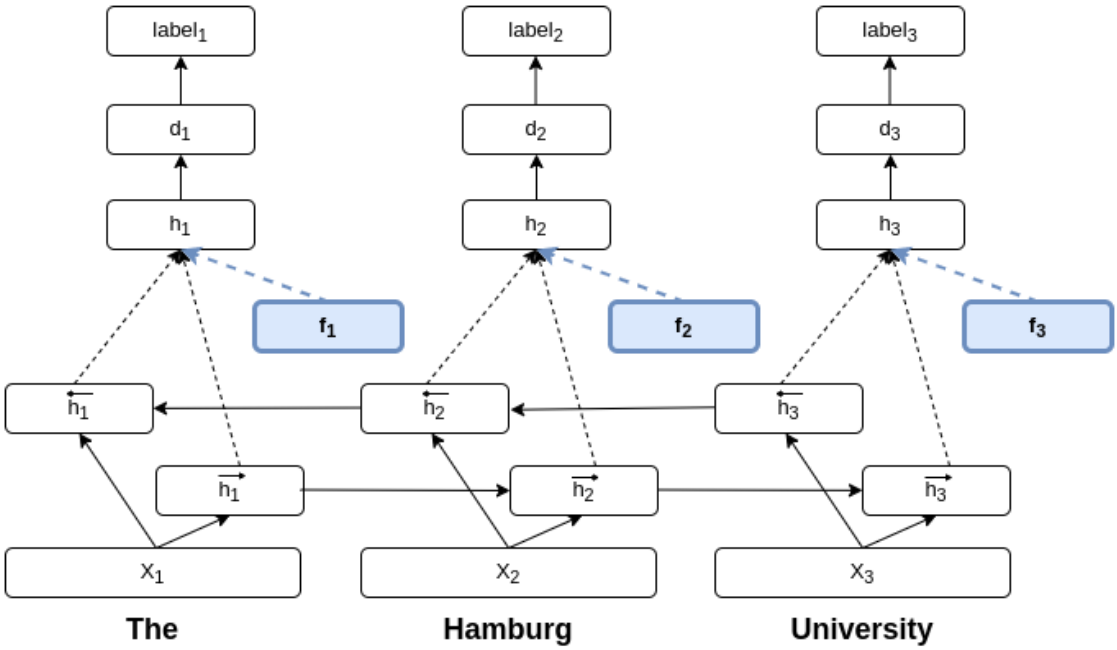

Figure 6.2: The unfolded network structure for a sequence labeling model with an additional language modeling objective, performing CWI on the sentence "The Hamburg University". The input tokens are shown at the bottom, the expected output labels are at the top. The arrows above variables indicate the directionality of the Bi-LSTM. Features (f_1, f_2, f_3) are concatenated to the output vector from Bi-LSTM

some of these features could improve the model performance as well for the CWI task overall, especially for L2 data. Instead of using extensive feature engineering, the features are identified and selected for my model from the features used in the CAMB system (Gooding and Kochmar 2018) because the CAMB system currently provides state-of-the-art results based on feature engineering only. The most common features used by several CWI systems are lexical and semantic features and corpus-based information such as word frequencies. Additionally, I also identified some features based on sentences to check the impact on target word complexity based on sentences. All the features I identified for my experiment are listed below and would be described in more detail in Section 6.6.

1) Sentence Features

1) ***Sentence length*:** the number of words in the sentence where target word occurs.

2) ***Sentence complexity*:** ratio of complex words to the total words in the current sentence.

3) ***Cosine Similarity*:** cosine similarity of target word to the other words in

the same sentence.

2) **Lexical Features**

1) ***Word length*:** the number of characters in the word.

2) ***Number of syllables*:** the syllable count for the target word, collected using the Datamuse API.

3) ***WordNet Features*:** number of synonyms, number of hypernyms and hyponyms for the word's lemma from WordNet.

3) **Word Frequency**

1) ***Google dataset of syntactic n-grams*** (Goldberg and Orwant 2013). This dataset is based on a corpus of 3.5 million English books containing 345 billion words. It contains several different forms of linguistic information on sequences of words up to 5 tokens long, with frequency counts over a corpus.

2) ***Lang-8 English learner corpus*** (Mizumoto et al. 2011). This corpus contains English learners' text extracted from Lang-8: an online platform for learning and practicing foreign languages. It has 100,051 English entries written by 29,012 active users.

4) **Lexicon-Based Features:** binary features indicating the presence of the word within a lexicon.

1) ***SubIMDB*:** a list produced using the SubIMDB corpus (G. Paetzold and Specia 2016d). The word frequency in the subtitles from the 'Movies and Series for Children' section is calculated, and the top 1,000 words are included in this list.

2) ***Simple Wikipedia (SimpWiki)*:** a list of the top 6,368 words contained in the Simple Wikipedia (Coster and Kauchak 2011).

3) ***Ogden's Basic English*:** a list of 1,000 words from Ogden's Basic English list.

6.6 Features Details & Importance in Hybrid Approach

My goal for hybrid architecture is to get better model performance by using the least number of features to avoid extensive feature engineering. This makes us identify features in Section 6.5, which are used by previous research effectively. However, as I am using deep learning most of the features do not put much impact on model learning. In this section, I will try to identify the features from Section 6.5 that make more impact on the model learning individually. For the hybrid approach, I will select the features, which provide a better F-score for at least three testsets than the best F-score achieved from the SEQ_{FTRec} model. For this purpose, I train the SEQ_{FTRec} model by introducing a single feature at a time and comparing the F-score achieved from the SEQ_{FTRec} model.

6.6.1 Sentence Length

For each target word, I computed the length of the sentence in which this target word occurs through tokenization. The idea behind using this feature is that longer sentences might be more complex for language learners than shorter sentences. In the whole training data, the maximum sentence length was 114 while the minimum sentence length was 3. The front propagation of neural networks involves the Dot Product of weights with input features. So, if the values are very high, the calculation of output takes a lot of computation time as well as memory. The same is the case during backpropagation due to which the model converges slowly if the inputs are not normalized. Considering this, I normalized the sentence length in the range [0,1] and passed this as an additional feature in the hybrid model. The F-score achieved from this experiment is compared in Table 6.7 to the F-score achieved from SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{senlen}}$
News	0.8929	0.8921
WikiNews	0.8801	0.8800
Wikipedia	0.8699	0.8683
InScript	0.5945	0.5964
OneStop	0.6444	0.6424

Table 6.7: F-score comparison of the SEQ_{FTRec} model results to the hybrid model with normalized sentence length feature.

The F-score achieved from the hybrid model using normalized sentence length as a feature has almost similar values to the F-Score achieved from the SEQ_{FTRec} model. This indicates adding this feature to the sequential model does not help in the model performance.

6.6.2 Sentence Complexity

I calculated the sentence complexity for each sentence by counting the total complex words in a particular sentence and dividing it by the length of that sentence. I did not consider only the complex word counts as sentence complexity because the shorter sentence would be more complex than the larger sentence with the same number of complex words. For instance, if *sentence1* is of length 10 and *sentence2* is of length 20 and both of them have 2 complex words then the complexity of both sentences would be 2 if we consider complex word counts. However, if we consider the ratio of complex words to sentence length then the *sentence1* will have 0.2 complexity which is bigger (or more complex sentence) than the 0.1 complexity of *sentence2*. For each target word in a particular sentence, I calculated the sentence complexity from the same sentence in which this word occur. I passed this complexity value as an additional feature in the

hybrid model. The F-score achieved from this experiment is compared in Table 6.8 to the F-score achieved from SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{sencomp}}$
News	0.8929	0.8934
WikiNews	0.8801	0.8804
Wikipedia	0.8699	0.8663
InScript	0.5945	0.5957
OneStop	0.6444	0.6465

Table 6.8: F-score comparison of the SEQ_{FTRec} model to the hybrid model with sentence complexity feature.

From the comparison Table 6.8 I can say that both of these models have almost similar performance, indicating, introducing sentence complexity as a feature does not help in the improvement of performance.

6.6.3 Cosine Similarity

I used the cosine similarity of the target word with the other non-stop words in the same sentence as an additional feature for the target word. I considered this feature as a combination of three values: maximum similarity, minimum similarity, and the mean of similarities from each non-stop word in the same sequence. The idea behind using this feature is to capture the complexity of target words that are complex in some sentences while non-complex in others. The cosine similarity could provide the logic behind this on the basis of word similarities in the sentence irrespective of their size.

For computing cosine similarity I used TfidfVectorizer from Sklearn, Word2Vec (Mikolov et al. 2013) and FastText (Bojanowski et al. 2016) embeddings. The similarities achieved by TfidfVectorizer were not promising at all. The best similarities I achieved using FastText embeddings while Word2Vec embeddings provided less similarity score for most similar words. Using FastText embeddings I computed the similarity of the target word with other non-stop words in the same sentence and used the maximum similarity, minimum similarity, and mean of all the similarities values as a feature for cosine similarity. I passed these values as an additional feature for cosine similarity in the hybrid model. Table 6.9 shows the comparison of F-Score achieved from this experiment to the SEQ_{FTRec} model.

For News, WikiNews, InScript, and OneStop testsets, the hybrid model slightly outperforms the F-score achieved from the SEQ_{FTRec} model. At the same time, the Wikipedia testset has a lower F-Score in the same proportion of higher in other shared task testsets. For L2 testsets, both models have almost similar values. This indicates that both models might converge to the same values if I will take an average of more experiments.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{cossim}}$
News	0.8929	0.8963
WikiNews	0.8801	0.8860
Wikipedia	0.8699	0.8628
InScript	0.5945	0.5961
OneStop	0.6444	0.6469

Table 6.9: F-score comparison of the SEQ_{FTRec} model to the hybrid model with cosine similarity feature.

6.6.4 Word Length

For each target word, I computed the number of characters the word included. The idea behind using this feature is that longer words might be more complex for language learners than shorter words. This feature is used by almost every paper using features to train the model for CWI. In the whole training data, the maximum word length was 24 while the minimum word length was 1. As I mentioned earlier, using a high range of values have their drawbacks in model training, I normalized the word length in the range [0,1] before passing it to the model as an additional feature. The F-score achieved from this experiment is compared in Table 6.10 to the F-Score achieved from SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{wordlen}}$
News	0.8929	0.8941
WikiNews	0.8801	0.8804
Wikipedia	0.8699	0.8643
InScript	0.5945	0.5997
OneStop	0.6444	0.6449

Table 6.10: F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized word length feature.

The F-Score achieved from word length as an additional feature is almost similar to the F-Score achieved from the SEQ_{FTRec} model. This ensures that the word length as an additional feature does not help the sequence model to improve the performance.

6.6.5 Syllables

A syllable is a unit of organization for a sequence of speech sounds. For instance, the word *cat* has one syllable while the word *computer* have three (com / pu / ter) syllables. For

each target word, I computed the number of syllables the word includes using Datamuse API. The idea behind using this feature is that a word could be complex if it requires many syllables to pronounce. In the whole training data, the maximum number of syllables used by a word was 7 while the minimum number of the syllable was 0. I normalized the values and passed them as an additional feature in the hybrid model. The F-score achieved from this experiment is compared in Table 6.11 to the F-Score achieved from SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{syll}}$
News	0.8929	0.8939
WikiNews	0.8801	0.8778
Wikipedia	0.8699	0.8654
InScript	0.5945	0.5953
OneStop	0.6444	0.6469

Table 6.11: F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized syllables feature.

The F-Score achieved from both models in the comparison table (6.11) have almost the same values, which conclude that passing syllables as an additional feature do not help to boost model performance.

6.6.6 Synonyms

For each target word, I computed the number of synonyms the word has using WordNet lexical database (Miller 1995). In the whole training data, the maximum number of synonyms of the word has 72 while the minimum number of synonyms was 0. I passed the normalized value of synonyms as an additional feature in the hybrid model. The F-score achieved from this experiment is displayed in Table 6.12 along with the F-Score achieved from SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{syn}}$
News	0.8929	0.8948
WikiNews	0.8801	0.8811
Wikipedia	0.8699	0.8693
InScript	0.5945	0.5948
OneStop	0.6444	0.6465

Table 6.12: F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized Synonym feature.

The F-Score achieved from synonym as an additional feature in sequence model is almost similar to the F-Score achieved from SEQ_{FTRec} model for all testsets. This indicates that introducing this feature in the sequential model is not much helpful for improving performance.

6.6.7 Hypernyms

For each target word, I computed the number of hypernyms the word has using WordNet lexical database (Miller 1995). Hypernyms is a word with a broad meaning constituting a category into which words with more specific meanings fall. For instance, the color is a hypernym of red. In the whole training data, the maximum number of hypernyms the word has 3 while the minimum number of hypernyms was 0. I passed the normalized value of hypernyms as an additional feature in the hybrid model. In Table Table 6.13, I have shown the F-score achieved from this experiment along with the F-Score achieved from SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{hyper}}$
News	0.8929	0.8961
WikiNews	0.8801	0.8826
Wikipedia	0.8699	0.8669
InScript	0.5945	0.5972
OneStop	0.6444	0.6435

Table 6.13: F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized Hypernyms feature.

Comparing the results achieved from introducing hypernyms as an additional feature in the sequence model to the SEQ_{FTRec} model, we can say that it does not help the model to improve the performance.

6.6.8 Hyponyms

For each target word, I computed the number of hyponyms the word has using WordNet lexical database (Miller 1995). Hyponyms are words that are specific examples of a general word. For example pigeons, crow, eagle, and seagull are all hyponyms of bird. In the training data, the maximum number of hyponyms the word has 402 while the minimum number of hyponyms was 0. I passed the normalized value of hyponyms as an additional feature in the hybrid model. The F-score achieved from this experiment is compared to the F-Score achieved from the SEQ_{FTRec} model in Table 6.14.

F-Score achieved for all testsets using hyponyms as an additional feature in the sequence model is almost similar to the F-Score achieved from the SEQ_{FTRec} model. Introducing hyponyms as an additional feature does not help the model to increase performance.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{hypo}}$
News	0.8929	0.8952
WikiNews	0.8801	0.8826
Wikipedia	0.8699	0.8672
InScript	0.5945	0.5934
OneStop	0.6444	0.6434

Table 6.14: F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized Hyponyms feature.

6.6.9 Word Frequency: Google dataset

For each target word, I estimated the frequency using Google dataset of syntactic n-grams (Goldberg and Orwant 2013), i.e, structures in which the contexts of words are based on their respective position in a syntactic parse tree, and not on their sequential order in the sentence. This dataset was derived from a very large (345 billion words) corpus of 3.5 million English books. The dataset includes over 10 billion distinct items covering a wide range of syntactic configurations. In the whole training data, the maximum word frequency was 6230 while the minimum word frequency was 0. I passed the normalized frequency from the Google dataset as an additional feature in the hybrid model. The F-score achieved from this experiment is displayed in Table 6.15 along with the F-Score achieved from the SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{googfreq}}$
News	0.8929	0.8928
WikiNews	0.8801	0.8769
Wikipedia	0.8699	0.8694
InScript	0.5945	0.5930
OneStop	0.6444	0.6458

Table 6.15: F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized word frequency in Google dataset as feature.

The F-Score achieved from both models shown in Table 6.15 has almost the same values, which conclude that adding target word frequency in the Google dataset as an additional feature does not help the model to improve the performance.

6.6.10 Word Frequency: Lang-8 Corpus

For each target word, I estimated the frequency using language learner corpus from Lang-8 crawled in September 2011. This corpus contains English learners' texts written by 29,012 active users. It has a total of 100,051 in English entries. The idea behind using this feature is that language learners mostly use non-complex words and higher frequencies of some word in this corpus indicates the word is non-complex. In the whole training data, I found that the maximum word frequency was 996373 while the minimum word frequency was 0 in the Lang-8 corpus. I normalized the word frequency from the Lang-8 corpus in the range [0,1] and passed this value as an additional feature in the hybrid model. The F-score achieved from this experiment is displayed in Table 6.16 along with the F-Score achieved from the SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{lang8freq}}$
News	0.8929	0.8942
WikiNews	0.8801	0.8791
Wikipedia	0.8699	0.8671
InScript	0.5945	0.5969
OneStop	0.6444	0.6455

Table 6.16: F-score comparison of the SEQ_{FTRec} model to the hybrid model with normalized word frequency in Lang-8 corpus as feature.

Comparing F-Scores achieved from this experiment to the one from the SEQ_{FTRec} model, I did not find any significant difference. Hence, it ensures introducing word frequency in the Lang-8 corpus as an additional feature in the sequence model does not help to boost its performance.

6.6.11 SubIMDB

For each target word, I checked its presence in the top 1000 frequent words used in the subtitles from the 'Movies and Series for Children' section provided by G. Paetzold and Specia (2016c). This is the binary feature, which is 1 if the target word is present in the top 1000 frequent words from subtitles, otherwise, 0. The idea behind this feature is that children's movies and subtitles frequently use non-complex words. I passed this binary value as an additional feature in the hybrid model. Table 6.17 compares the results achieved from this experiment to the SEQ_{FTRec} model.

The F-Scores achieved from this experiment for all testsets are negligibly less than the score achieved from the SEQ_{FTRec} model. However, there is no significant difference between the two models' F-Scores.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{subimdb}}$
News	0.8929	0.8852
WikiNews	0.8801	0.8752
Wikipedia	0.8699	0.8648
InScript	0.5945	0.5923
OneStop	0.6444	0.6392

Table 6.17: F-score comparison of the SEQ_{FTRec} model to the hybrid model with subIMDB as feature.

6.6.12 Simple Wikipedia

For each target word, I checked its presence in the top 6,370 frequent words used in the simple Wikipedia corpus provided by Coster and Kauchak (2011). This is the binary feature, which is 1 if the target word is present in the top 6,370 frequent words from simple Wikipedia, otherwise, 0. The idea behind using this feature is that Simple Wikipedia mostly uses non-complex, simple English words and grammar. I passed this binary value as an additional feature in the hybrid model. The F-Score achieved from this experiment is displayed in Table 6.18 along with F-score achieved from the SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{simwiki}}$
News	0.8929	0.8789
WikiNews	0.8801	0.8612
Wikipedia	0.8699	0.8674
InScript	0.5945	0.5836
OneStop	0.6444	0.6357

Table 6.18: F-score comparison of the SEQ_{FTRec} model to the hybrid model with simple Wikipedia as feature.

The hybrid model uses a binary value for target words from simple Wikipedia most frequent 6,370 words as the feature did not perform well than the SEQ_{FTRec} model for any testsets. Except for the Wikipedia testset this hybrid approach has an approximate 1% less score for all testsets.

6.6.13 Ogden's Basic English

For each target word, I checked its presence in the list of 1,000 words from Ogden's Basic English. This is the binary feature, which is 1 if the target word is present in Ogden's Basic English, otherwise, 0. The idea behind using this feature is that Ogden's Basic

English is the collection of non-complex words, which could help the model for better classification. I passed this binary value as an additional feature in the hybrid model. Table 6.19 displays the F-Score achieved from this experiment along with the F-Scores achieved from the SEQ_{FTRec} model.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{Hybrid_{ogdens}}$
News	0.8929	0.8882
WikiNews	0.8801	0.8736
Wikipedia	0.8699	0.8730
InScript	0.5945	0.6002
OneStop	0.6444	0.6451

Table 6.19: F-score comparison of the SEQ_{FTRec} model to the hybrid model with Ogden's Basic English as binary feature.

For Wikipedia, and InScript testsets, the hybrid model slightly outperforms the F-score achieved from the SEQ_{FTRec} model. However, for the News and WikiNews testsets it has a slightly less score than the SEQ_{FTRec} model. This is the only feature that improves the significance performance on the Wikipedia testset when it is introduced in the sequence model.

6.7 Experiment using Hybrid Approach

In the hybrid approach for CWI, I will introduce two or more features in the sequential model to improve the resulting hybrid model performance. My aim here is to introduce the least number of hand-engineered features while improving model performance. In Section 6.7.1, I provide my approach for features selection and Section 6.7.2 provides the results achieved from the hybrid approach using selective features.

6.7.1 Features Selection

Almost every features discussed in Section 6.6 have achieved similar performance to the SEQ_{FTRec} model (without any feature induction) except for *Ogden's Basic English* feature, which has improved the F-Score for Wikipedia testset significantly. However, this feature does not perform competitively for other testsets. So, for my hybrid approach, I decided to incorporate a small group of features together with the goal to improve the hybrid model performance. I created two groups of features based on different approaches.

Manually Features Selection

In this features selection process, I have selected some important features, which have negligibly better performance when introduced to the sequential model individually (see

Section 6.6). Intending to select the least number of hand-engineered features, I further filtered the list to five features that provide some external information as well as not being redundant. I named this group of features as $Features_{exp}$. The features placed in this group are:

1) The *Syllable* count of the target word.

2) The number of *Synonyms* of target word.

3) The number of *Hypernyms* of target word.

4) The presence of target word in *Ogden's Basic English* list.

5) The target word frequency in language learners *Lang-8* corpus.

Features Selection Using Permutation Importance

Permutation feature importance is a model inspection technique that can be used after a model has been fitted on a dataset. Then the predictions are made on the fitted model for the dataset, although the values of a single column (feature) in the dataset are randomly shuffled, leaving the target and all other columns in place. This process is repeated for each feature in the dataset for the provided number of times. It uses predictions and the true target values to calculate how much the loss function suffered from shuffling. The result is a mean importance score for each input feature.

Figure 6.3 displays the scores achieved by various hand-engineered features by fitting them in the Logistic Regression model using the permutation importance technique. As these features are created independently from the context, I used logistic regression to identify the features having more impact on the binary classification of the tokens. I permuted five times each feature to decide the importance. With the aim of least feature selection, I chose two features: length of the target word and the mean of the cosine similarity of the target word to other words in the same sequence. For further reference, I named this group of features as $Features_{PI}$.

6.7.2 Hybrid Experiment

For the hybrid approach, I perform two experiments based on the group of features selected to introduce in the SEQ_{FTRec} model.

Hybrid Approach Using $\text{Features}_{\text{exp}}$ Features

I trained the hybrid model (SEQ_{hybExp}) by introducing $Features_{exp}$ features in the SEQ_{FTRec} model. The F-score achieved from this experiment is displayed in Table 6.20 along with the F-Score achieved from the SEQ_{FTRec} model for comparison.

The SEQ_{hybExp} model with selected five features performed slightly better for the InScript and OneStop test data. For the Wikipedia testset, it has the almost same performance as from the SEQ_{FTRec} model. For the News, and WikiNews test data

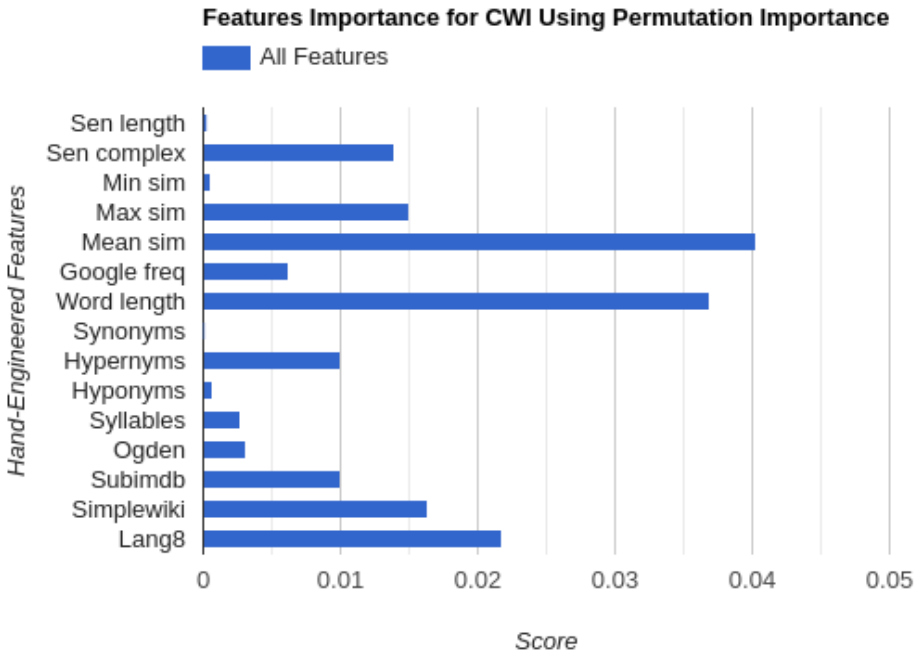

Figure 6.3: Permutation feature importance using Logistic Regression.

Testset	**SEQ$_{\mathbf{FTRec}}$**	**SEQ$_{\mathbf{hybExp}}$**
News	0.8929	0.8897
WikiNews	0.8801	0.8757
Wikipedia	0.8699	0.8701
InScript	0.5945	0.6025
OneStop	0.6444	0.6473

Table 6.20: F-score comparison of the SEQ_{FTRec} model to the SEQ_{hybExp} model introduced with $Features_{exp}$ features.

SEQ_{hybExp} model approach have slightly less score than the SEQ_{FTRec} model. However, the score difference between these models is negligible and could get converged to the same values if the average will be taken for multiple experiment runs.

Hybrid Approach Using $\mathbf{Features_{PI}}$ Features

In this experiment, I introduced $Features_{PI}$ features in the SEQ_{FTRec} model while training to achieve the SEQ_{hybPI} model performance. The F-score achieved from this experiment is displayed in Table 6.21 along with the F-Score achieved from the SEQ_{FTRec} and SEQ_{hybExp} models for comparison.

Testset	$\mathbf{SEQ_{FTRec}}$	$\mathbf{SEQ_{hybPI}}$
News	0.8929	0.8949
WikiNews	0.8801	0.8801
Wikipedia	0.8699	0.8666
InScript	0.5945	0.6000
OneStop	0.6444	0.6477

Table 6.21: F-score comparison of the SEQ_{FTRec} model to the SEQ_{hybPI} model introduced with $Features_{PI}$ features.

The F-Scores achieved from the SEQ_{hybPI} model is similar to the F-Scores achieved from the SEQ_{FTRec} model without much significant difference. Whilst SEQ_{hybPI} model has achieved the higher F-Score for the News testset, it has lower for the Wikipedia testset in comparison to SEQ_{FTRec} model. These smaller changes in scores for these models might converge to the same values if I take an average of more experiments.

Feature engineering, which was the essential part of the traditional machine learning algorithms, did not provide significant improvement on model performance while introduced in the sequential model that incorporates FastText and is recall-oriented, especially for the Shared Task test sets. However, given the deviation of the L2 data from the training data, the hybrid approach achieved a slightly better score for L2 test sets.

7 Tuning Transformers for CWI

Bidirectional long short-term memory networks (BiLSTMs) have been widely used as an encoder for complex word identification (CWI) tasks. Recently, the fully-connected self-attention architecture i.e., Transformer is broadly adopted in various natural language processing (NLP) tasks owing to its parallelism and advantage in modeling the long-range context. The Transformer model relies solely on the use of self-attention, where the representation of a sequence (or sentence) is computed by relating different words in the same sequence. The word embeddings techniques in transformers take into consideration the context of the word and can be seen as dynamic word embeddings techniques, most of which make use of some language model to help modeling the representation of a word. This makes the same word have different representations based on the context where it occurs. In this experiment, I prove that the Transformer-like encoder is just as effective for CWI as other NLP tasks. Experiments on CWI shared task dataset (Yimam et al. 2018) of 3 different genres show that the Transformers achieves superior performance than the prevailing BiLSTM-based models.

7.1 Training Data Setup

I have generated two additional training data sets to be trained with transformers. The additional two training data sets have been generated on the basis of excluding stop words and punctuation. The three training data generated collectively for each genre in the CWI shared task dataset has been described below.

- $Train_{All}$: Training data with all the stop words and punctuation. This data was used previously by almost all the previous systems including winning/state-of-the-art systems. This training data have total $52,476$ tokens, out of that $8,408$ are annotated as complex while $44,068$ are annotated as non-complex (Table 7.1). This makes it a very imbalanced class with approximately 84% of tokens marked as non-complex.

- $Train_{WoP}$: This training data was generated by removing all the punctuation in the original training data. I created this training data with the belief that punctuation does not carry any meaning in the context of sequence and removing them would not degrade the performance of the model. However, it could improve the model performance by reducing the number of majority non-complex tokens as all punctuation were annotated as non-complex. As shown in Table 7.1, after removing punctuation, training data have total $46,260$ tokens, out of that $8,408$ are annotated as complex while $37,852$ are annotated as non-complex. Removing

punctuation does not reduce the number of complex annotated tokens as none of the punctuation has been annotated complex. This training data is still highly imbalanced with approx 82% of the token are annotated as non-complex.

- $Train_{clean}$: This training data was generated by removing all the stop words and punctuation from the original training data using NLTK (Bird and Loper 2004) library. In this training data, along with punctuation I also removed stop words on the same belief that stop words do not contribute to the context of the sentence. Stop words are those words that are unnecessary words and they do not carry meaning to your sentences or corpus. However, it is not always true as some stop words like 'not' have the meaning and can change the context of the sentence that could be very meaningful for the application like sentiment analysis. So removing stop words from data bring down the data size but with a slight loss of information. Also, the definition of what is a stop word may vary. We may consider a stop word a word that has high frequency on a corpus or we can consider every word that is empty of true meaning given a context. So, there is no straight answer for whether we should remove stop words from the corpus or not. The solution is we can try with and without stop words for model training and can decide whether removing stop words is good for my application or not. This makes us create this training data for my application of CWI.

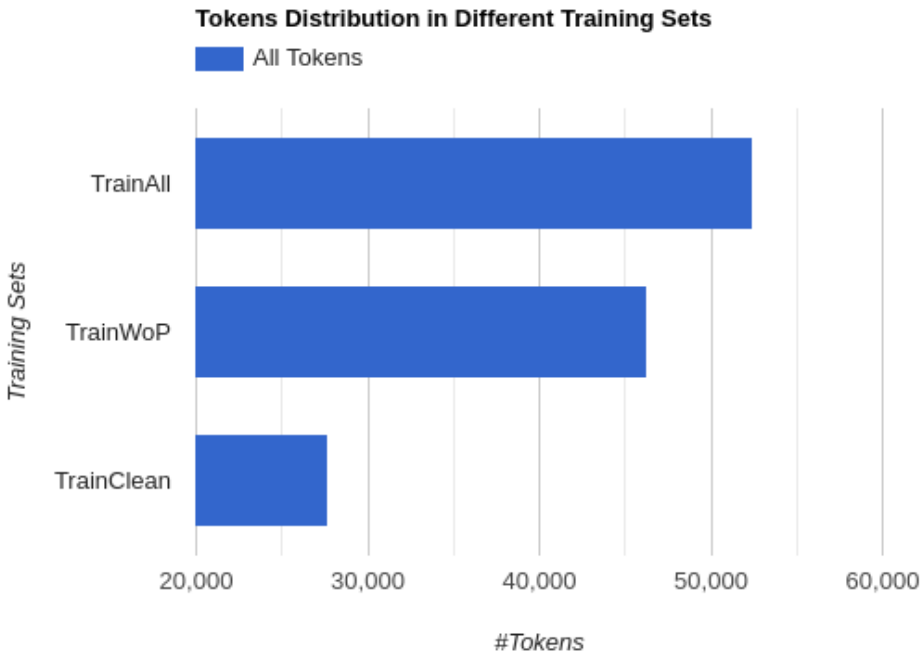

Figure 7.1: Tokens distribution in different training sets.

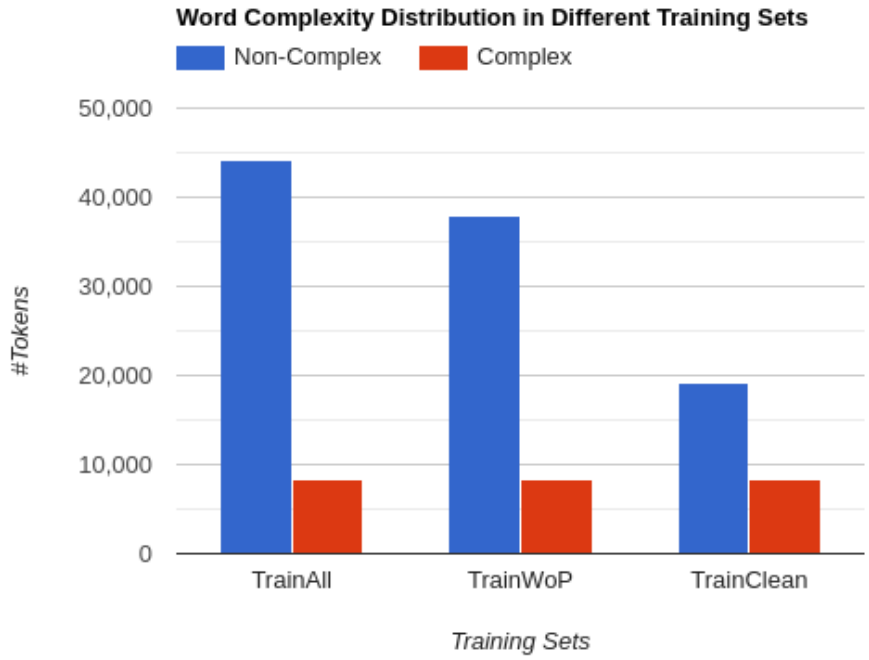

Figure 7.2: Word complexity distribution in different training sets.

Figure 7.2 displays the token distribution in different training sets. Removing stop words from training data has drastically decreased the number of tokens in training data as expected. In Figure 7.2, I displayed the word complexity distribution in different training sets.

Training Data	**#Sen**	**#tokens**	**#C**	**#N**	**N (percentage)**
$Train_{All}$	2,063	52,476	8,408	44,068	83.98
$Train_{WoP}$	2,063	46,260	8,408	37,852	81.82
$Train_{clean}$	2,063	27,667	8,399	19,268	69.64

Table 7.1: Total number of sentence, tokens, complex and non-complex tokens and percentage of non-complex tokens in each training data for transformer

As shown in Table 7.1, after removing punctuation and stop words, training data have a total of $27,667$ tokens, out of that $8,399$ are annotated as complex while $19,268$ are annotated as non-complex. After removing punctuation I have almost 82% of tokens that have been annotated as non-complex. The $Train_{All}$ training data has a $6,216$ number of punctuation and an $18,593$ number of stop words. All punctuation was annotated as non-complex, however, removing stop words from the $Train_{WoP}$ training data reduced

the number of complex words as well by 9. Digging into training data I found that stop words like 'nor', 'haven', 'whom', 'ain', and 'won' were annotated as complex in total 9 times. However, this training data is no more highly imbalanced with approximate 70% of tokens were annotated as non-complex. As these training data have been created to reduce the class imbalance, I also reported accuracy to compare the transformer model performance trained on three different training data.

7.2 Approach

I use the Transformers library provided by Huggingface (Wolf et al. 2019), which allows us to pre-train and fine-tune BERT (Devlin et al. 2018) models with a simplified procedure. BERT model can be fine-tuned on tasks that use the whole sentence (potentially masked) to make decisions, such as sequence classification, token classification, or question answering. For this study, firstly, I fine-tune an English RoBERTa (Liu et al. 2019) large model using shared task training data together for token classification (CWI). As RoBERTa was trained on a ton more data than BERT, RoBERTa outperforms BERT on many benchmark results. RoBERTa has been pre-trained on the BookCorpus, English Wikipedia, CC-News, OpenWebText, and stories amounting to a grand total of 160 GB of text, which ensures that it could be the best fit for CWI for language learners from different genres. The pre-trained RoBERTa model is cased, making it appropriate for fine-tuning on CWI tasks by preserving casing information. Later, I also fine-tune an English BERT (Devlin et al. 2018) base cased model, which has been pre-trained on the BookCorpus, and English Wikipedia, amounting to a total of 16 GB of text (3.3 billion words) on the same training data.

The RoBERTa large model contains 24 transformer blocks, $1,024$ hidden layers, 16 self-attention blocks, and 355 million parameters in total. The BERT Base model contains 12 transformer blocks, 768 hidden layers, 12 self-attention blocks, and 110 million parameters in total. I used the batch size of 16 and the learning rate of 5^{-5} as hyperparameters for the CWI task. I train the fine-tuned RoBERTa model for CWI using the three different training data separately that I prepared based on with or without stop words and punctuation (Section 7.1). Later, I train the fine-tuned BERT model for the training data, for which I achieved the best score using RoBERTa model. Also, keeping the other hyperparameters the same, I train the model for 6 epochs and notice the best number of epochs required to train the model. For each experiment, I have reported the average of three experiments and compared their performances.

7.3 Experiment

The experiment is broadly divided into two parts: firstly, evaluating the RoBERTa model for CWI and finding the best training data from the three then evaluating the BERT model on the training data, for which RoBERTa model performs best. An experiment using the BERT model has been included to confirm whether RoBERTa model that has been trained on a ton of data for many different genres gets the benefit of it for the CWI

system or not. Also, this experiment will confirm whether we need a bigger network for the CWI task or not. The experiment using RoBERTa model is divided into three parts based on three different training data. Firstly, I perform the model training using the training file with all the stop words and punctuation. Later, I will perform the model training using the training file without any punctuation and another experiment using the training file without any stop word and punctuation. Finally, I will compare the model performance achieved by all three experiments and decide the best training file I can use for CWI using transformers.

7.3.1 Fine-Tuning RoBERTa Large Model

Firstly, I have considered the $Train_{All}$ training data for the experiment, which includes all the stop words and punctuation. This training data is similar to the previous researches training data for CWI. In Figure 7.3, I plotted the learning curve from this experiment. The training loss and development loss intersect each other after 2^{nd} epoch. It indicates that the best number of epochs to train this model using shared task train data is 2. Training model for more than 2 epochs will overfit the model on training data.

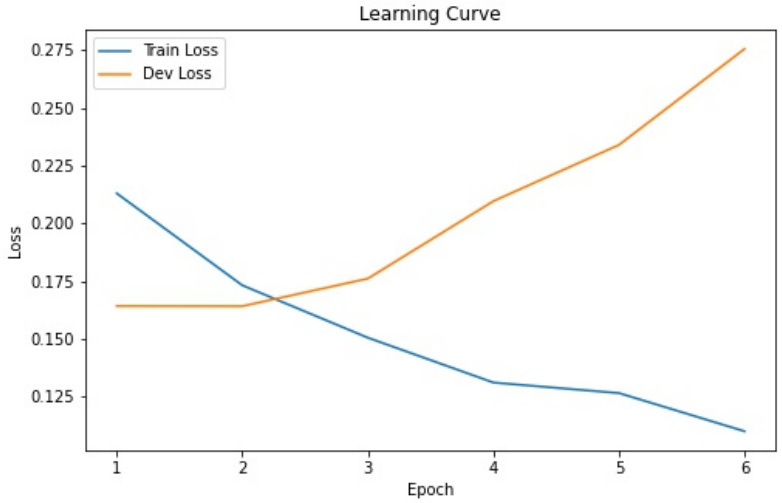

Figure 7.3: Learning curve for training data with all stop words and punctuation ($Train_{All}$).

Table 7.2 compares the F-Scores achieved for the testsets by fine-tuning RoBERTa on $Train_{All}$ data to the baseline results from the $SEQ_{baseline}$ model. The F-Score achieved by fine-tuning RoBERTa model for all testsets has approximately 2.5% better value than the baseline results.

Figure 7.4 demonstrates the learning curve for fine-tuning RoBERTa model on $Train_{WoP}$ training data, which does not include any punctuation. Training loss and development loss line intersects each other after 3^{rd} epoch indicating 3 epochs as the best number of

Testset	$\mathbf{SEQ_{baseline}}$	RoBERTa
News	0.8694	0.8969
WikiNews	0.8544	0.8873
Wikipedia	0.8369	0.8613
InScript	0.5815	0.6069
OneStop	0.6317	0.6588

Table 7.2: Comparison of macro F-Scores achieved by fine-tuning RoBERTa on $Train_{All}$ data to baseline results.

epochs for this model training. However, there is no improvement in development loss after 1^{st} epoch.

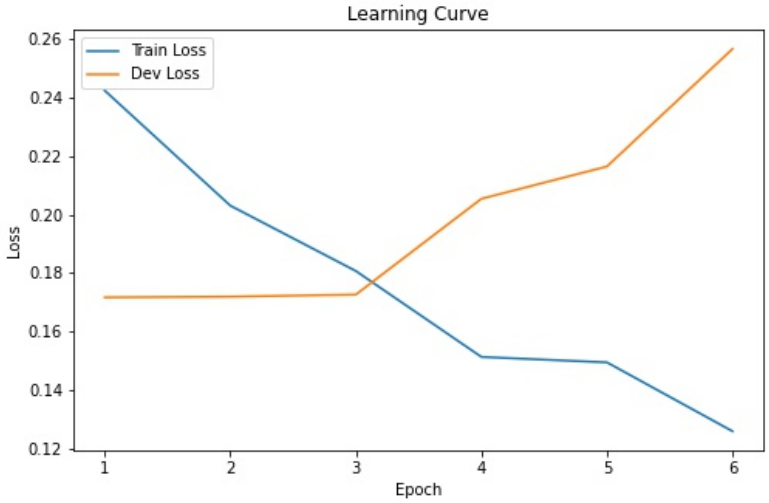

Figure 7.4: Learning curve for train data without punctuation

In Table 7.3 I compare the F-Scores achieved for the various testsets by fine-tuning RoBERTa on $Train_{WoP}$ data to the baseline results. Removing punctuation from training data did not help in model learning and achieved almost similar performance as using $Train_{All}$ data. It has achieved approximately 2% better F-Score value for each testset.

Further, I fine-tuned the RoBERTa model for the cleaned $Train_{clean}$ training data that does not include any stop words and punctuation. This makes this experiment interesting to see if it could improve the accuracy of the testsets. Figure 7.5 demonstrates the learning curve for the model training. Training loss and development loss line intersects each other after 5^{th} epoch, indicating 5 epochs would be a good choice to train this model. However, there is no improvement in development loss after 2^{nd} epoch.

Testset	$\mathbf{SEQ_{baseline}}$	$\mathbf{RoBERTaTrain_{WoP}}$
News	0.8694	0.8926
WikiNews	0.8544	0.8848
Wikipedia	0.8369	0.8580
InScript	0.5815	0.6074
OneStop	0.6317	0.6558

Table 7.3: Comparing macro F-Scores achieved by fine-tuning RoBERTa on $Train_{WoP}$ data to baseline results.

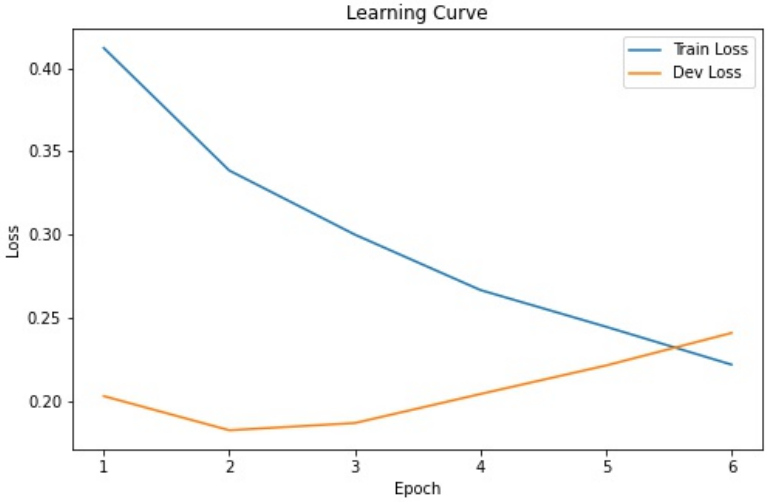

Figure 7.5: Learning curve for train data without stop words and punctuation

The F-Scores achieved by fine-tuning RoBERTa model on $Train_{clean}$ data for various testsets is displayed in Table 7.4 along with the baseline results.

In Table 7.5, I compared macro F-scores and accuracies achieved for testsets by fine-tuning RoBERTa model on different training sets. From the comparison table, we can see that for all testsets except for InScript, using the $Train_{All}$ training file I achieved the best performance. For InScript testset as well, I have approximately the same F-Scores by all three approaches. The F-Score achieved by training RoBERTa model using $Train_{WoP}$ training file has the almost same score as training with $Train_{All}$ training file. And if I take the average of more experiments, the performance achieved by these two files might converge to the same values. However, when I removed the stop words as well from the training file ($Train_{clean}$), the performance of the model is noticeably degraded for all testsets, especially for News and WikiNews testsets. Similar performance pattern we see for the accuracies as well except for InScript testset, where using $Train_{clean}$

Testset	$\mathbf{SEQ_{baseline}}$	$\mathbf{RoBERTaTrain_{clean}}$
News	0.8694	0.8829
WikiNews	0.8544	0.8648
Wikipedia	0.8369	0.8521
InScript	0.5815	0.6066
OneStop	0.6317	0.6503

Table 7.4: Comparing macro F-Scores achieved by fine-tuning RoBERTa on $Train_{clean}$ data to baseline results.

training file have the almost similar performance of using $Train_{All}$ training file. This comparison ensures that removing stop words would not be a good idea for the CWI system. Hence, for the CWI task, I will consider the $Train_{All}$ training file which includes all the punctuation and stop words as a better approach.

	F-Score			Accuracy		
Testset	$\mathbf{Train_{clean}}$	$\mathbf{Train_{WoP}}$	$\mathbf{Train_{All}}$	$\mathbf{Train_{clean}}$	$\mathbf{Train_{WoP}}$	$\mathbf{Train_{All}}$
News	0.8829	0.8926	0.8969	0.9457	0.9495	0.9513
WikiNews	0.8648	0.8848	0.8873	0.9250	0.9356	0.9364
Wikipedia	0.8521	0.8580	0.8613	0.9132	0.9172	0.9187
InScript	0.6066	0.6074	0.6069	0.9245	0.9187	0.9215
OneStop	0.6503	0.6558	0.6588	0.8846	0.8846	0.8892

Table 7.5: Comparison of macro F-Scores and accuracies achieved by fine-tuning RoBERTa model with different training sets.

7.3.2 Fine-Tuning BERT Base Cased Model: Training with $\mathbf{Train_{All}}$ data

As RoBERTa (160 GB) was trained on a ton more data than BERT (16 GB), I created this experiment to clarify whether my CWI system gets the benefit of it or not. Also, I have selected the 'bert-base-cased' model that has a smaller network than the 'bert-large' model. It will ensure us that whether for my task we needed a larger network for training or a smaller network is good enough. I have selected the cased model, as my training data is cased and the complexity of each word will be based on the context of sequence. It will be interesting to see the performance of the 'bert-base' model on my L2 testsets as they are based on different genres than the BERT model was pre-trained, unlike RoBERTa, which was also pre-trained on OpenWebText and stories.

In this experiment, I have considered the $Train_{All}$ training data that includes all the stop words and punctuation as using this training data I achieved the best performance

from the RoBERTa model. Also, the same training data has been used by previous researchers for the CWI task. Figure 7.6 demonstrates the learning curve for the 'bert-base-cased' model training. Training loss and development loss line intersects each other after 2^{nd} epoch indicating 2 epochs as the best number of epochs for this model training.

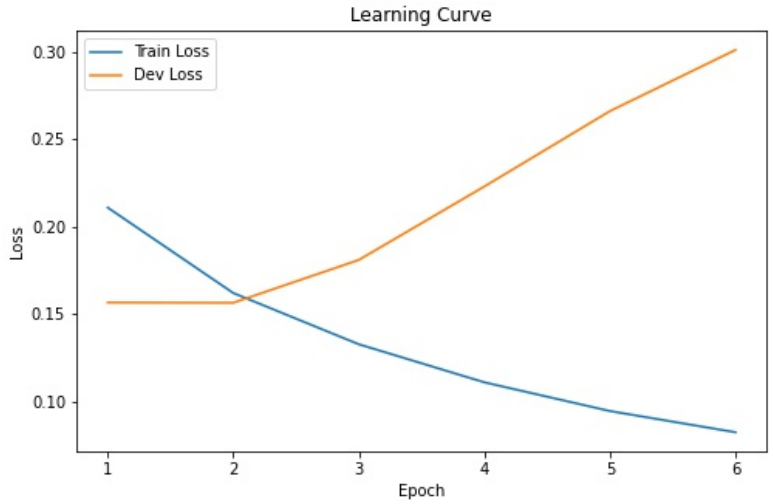

Figure 7.6: Learning curve for train data using BERT model

Table 7.6 shows the comparison between the F-Scores achieved for various testsets by fine-tuning the BERT model using $Train_{All}$ training data to the baseline results. The F-Scores achieved for the Shared Task testsets by fine-tuning the BERT model have almost 3% better value than the baseline results. Especially for the Wikipedia testset, it achieved a 4% better F-Score. For L2 testsets as well BERT model has achieved better performance than baseline results.

Testset	**SEQ$_{\text{baseline}}$**	**BERT**
News	0.8694	0.8975
WikiNews	0.8544	0.8859
Wikipedia	0.8369	0.8784
InScript	0.5815	0.5976
OneStop	0.6317	0.6529

Table 7.6: Comparing macro F-Scores achieved by fine-tuning BERT using $Train_{All}$ data to the baseline results.

8 Results and Error Analysis

In this chapter, I will discuss in detail my experiment results from various approaches and the error analysis.

8.1 Results

The main objective of this thesis is to provide a CWI system, which could help the English language learners in various applications built on top of this system. This makes us collect and annotate the more general text (L2 Data) that language learners come across frequently. In the following section, I will discuss the results achieved from various approaches for Shared Task 2018 data and L2 data.

8.1.1 CWI Shared Task 2018 Dataset

For CWI on the Shared Task 2018 Dataset, I trained the baseline sequential model, hybrid model, and transformer-based models RoBERTa and BERT. I also tried to improve the baseline results in the sequential model by incorporating better word embeddings and tuning F05-Score (6.1), which is the weighted harmonic mean of the precision and recall.

Table 8.1 shows the comparison of my baseline results ($SEQ_{baseline}$) trained on Shared Task 2018 Datasets with the state-of-the-art SEQ model and other approaches in this thesis. The baseline results were achieved while I tried to replicate the results of Gooding and Kochmar (2019) using the sequential architecture by Rei (2017). The $SEQ_{baseline}$ model has almost the same results for News and WikiNews testsets as the SEQ model achieved. However, for the Wikipedia testset, $SEQ_{baseline}$ model has 2.29% better value than the SEQ model. As mentioned before, the reason could be either the way they tokenized and formatted the model acceptance data file or the way they adapted the sequential architecture by Rei (2017).

Further, I tried to improve the $SEQ_{baseline}$ model by incorporating FastText word embeddings instead of GloVe word embeddings. Also, I tuned this model (SEQ_{FTRec}) to improve the recall so that the model better classifies the complex words. The F-Scores achieved by the SEQ_{FTRec} model has 2.35%, 2.57, and 3.3% better values than the $SEQ_{baseline}$ model for News, WikiNews, and Wikipedia testsets respectively. Adding contextualized embeddings to the sequential model could be another interesting experimental setting, but due to technical implementation problems and limited time, this was left out. Along with taking the benefits of FastText word embeddings, the tuning of the model to improve recall has done the trick for better performance of the SEQ_{FTRec} model. As the data has an imbalanced class, improving recall lets the model focus to learn the complex words better without much loss on the precision metric.

	News		WikiNews		Wikipedia	
Models	**F-Score**	**Gain**	**F-Score**	**Gain**	**F-Score**	**Gain**
SEQ	87.63	+0.69	85.40	-0.04	81.40	-2.29
$SEQ_{baseline}$	86.94	0.0	85.44	0.0	83.69	0.0
SEQ_{FTRec}	89.29	+2.35	88.01	+2.57	86.99	+3.3
SEQ_{hybExp}	88.97	+2.03	87.57	+2.13	87.01	+3.32
SEQ_{hybPI}	89.49	+2.55	88.01	+2.57	86.66	+2.97
$RoBERTa$	89.69	+2.75	**88.73**	+3.29	86.13	+2.44
$BERT$	**89.75**	+2.81	88.59	+3.15	**87.84**	+4.15

Table 8.1: Comparison of macro F-scores achieved from state-of-the-art SEQ model to various approaches in this thesis for CWI in News, WikiNews and Wikipedia testsets from Shared Task 2018. The values displayed in this table are in percentage.

I also tried the hybrid approach by identifying thirteen hand-engineered features that could be useful for the CWI and selected a few of them to introduce together in the SEQ_{FTRec} model, resulting in a hybrid model. I created two groups of important features named $Features_{exp}$ and $Features_{PI}$ and introduced them separately in two different experiments. Whilst the $Features_{exp}$ group is a collection of five features that have been selected on the basis of their individual performance in the sequential model as well as providing some external information that could not be extracted from the training data, the $Features_{PI}$ group has two features selected from the permutation feature importance technique. The SEQ_{hybExp} model is trained by introducing the $Features_{exp}$ features, which includes presence of target word in *Ogden's Basic English*, target word frequency in *Lang-8* corpus, and *Syllable* count, number of *Synonyms* and *Hypernyms*. The F-Scores achieved by the SEQ_{hybExp} model has 2.03%, 2.13%, and 3.32% better values than the $SEQ_{baseline}$ model for News, WikiNews and Wikipedia testsets respectively. The $Features_{PI}$ features (target word length and cosine similarity of target to other words in the same sentence) introduced in the SEQ_{hybPI} model training, which achieved 2.55%, 2.57%, and 2.97% better values than baseline. However, both of these models have almost similar performance as I achieved from the SEQ_{FTRec} model, indicating, introducing features in the SEQ_{FTRec} model is not helping the model to improve performance.

Finally, I also tried fine-tuning the transformer models RoBERTa and BERT for the CWI task. Whilst the $RoBERTa$ model has achieved 2.75%, 3.29%, and 2.44% better F-Scores values than the $SEQ_{baseline}$ model for News, WikiNews and Wikipedia testsets respectively, the $BERT$ model has achieved 2.81%, 3.15%, and 4.15% better values. Both models have achieved competitive, state-of-the-art results for News and Wikinews testsets, however, the $RoBERTa$ model fails to get similar performance in the Wikipedia testset. The $BERT$ model has performed exceptionally well for the Wikipedia testset

than all other approaches, achieving 1.71% better F-Score value than the *RoBERTa* model.

8.1.2 L2 Dataset

The L2 dataset is comprised of the InScript and OneStop data, which I have collected as language learners data and annotated on the basis of CEFR levels. The vocabulary complexity of this dataset is different from the Shared Task dataset. However, I did not use the L2 data for model training as it is annotated programmatically and has the most chances of being prone to errors. I trained the model on the Shared Task data and tested the performance on the InScript and OneStop data.

In Table 8.2, I have shown the comparison of results achieved from the $SEQ_{baseline}$ model to the other approaches I have used in this thesis. The macro F-Scores achieved from the $SEQ_{baseline}$ model for InScript and OneStop testsets is 58.15% and 63.17% respectively. The SEQ_{FTRec} model has a difference of 1.3% and 1.27% than the baseline results for InSCript and OneStop data respectively, using the benefits of better word embeddings. Unlike for Shared Task testsets, introducing hand-engineered features helped to achieve better results. Both models based on a hybrid approach, i.e., the SEQ_{hybExp} model and the SEQ_{hybPI} have achieved almost similar performance for L2 testsets. The reason could be that these data have been annotated on the basis of word complexity only, without taking the context of the sentence where they occur into account.

	InScript		**OneStop**	
Models	**F-Score (%)**	**Gain (%)**	**F-Score (%)**	**Gain (%)**
$SEQ_{baseline}$	58.15	0.0	63.17	0.0
SEQ_{FTRec}	59.45	+1.30	64.44	+1.27
SEQ_{hybExp}	60.25	+2.1	64.73	+1.56
SEQ_{hybPI}	60.00	+1.85	64.77	+1.60
$RoBERTa$	**60.69**	+2.54	**65.88**	+2.71
$BERT$	59.76	+1.61	65.29	+2.12

Table 8.2: Comparison of macro F-scores achieved from baseline $SEQ_{baseline}$ model to various approaches in this thesis for CWI in InScript and OneStop testsets from L2 data.

The results achieved for these data by fine-tuning transformer-based language models RoBERTa and BERT were also promising. The $RoBERTa$ model has achieved the highest F-Scores for InScript and OneStop testsets by 2.54% and 2.71% values respectively from the baseline. The $BERT$ model has a difference of 1.61% and 2.12% from the baseline for the InScript and OneStop testsets respectively.

8.2 Error Analysis

In this section, I will investigate the systematic and random erroneous predictions by models in different testsets included in this thesis. The predictions from the $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models will be analyzed for each testset individually.

8.2.1 News Testset

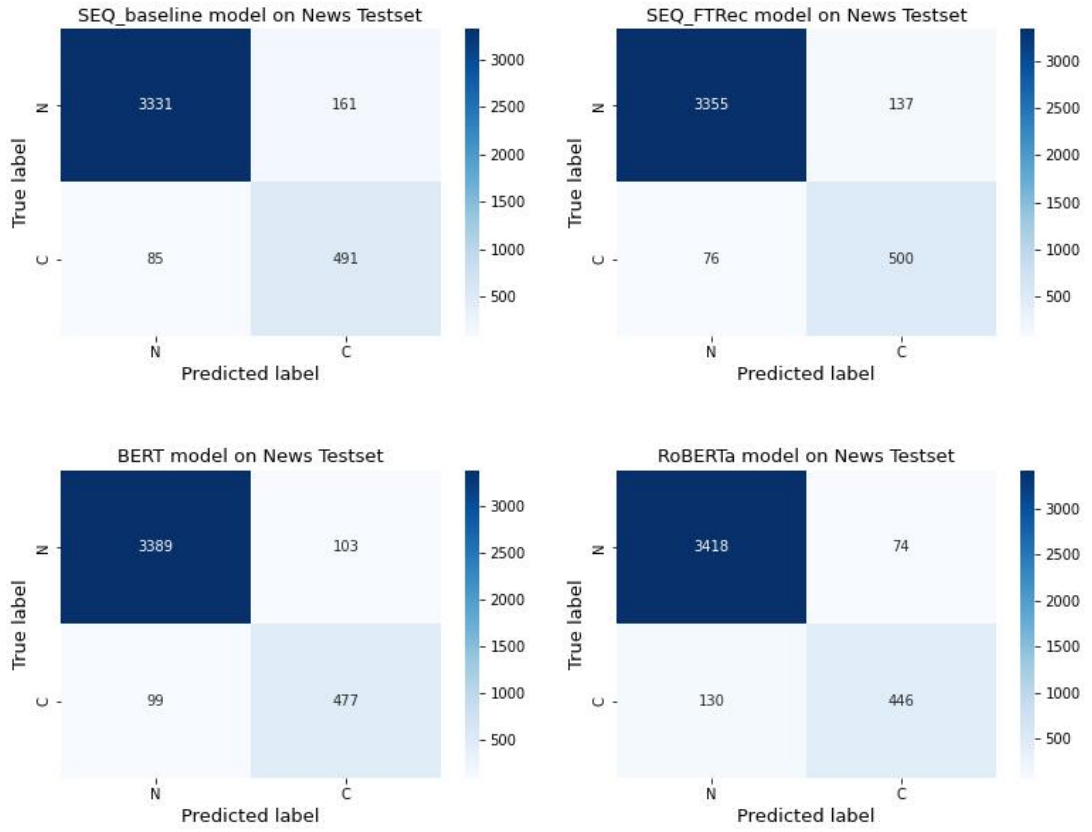

Figure 8.1: Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the News testset.

Figure 8.1 demonstrates the confusion matrix for News testset from the baseline $SEQ_{baseline}$ model and my best approaches in this thesis, i.e., SEQ_{FTRec}, $BERT$, and $RoBERTa$ model for the CWI. The sequential models predicted the complex words better than the transformer-based pre-trained language models. However, the sequential models have more errors in predicting the non-complex words. In Figure 8.2, I have shown the total number of errors made by these models for CWI in the News testset. The $BERT$ and $RoBERTa$ models have performed almost similarly with a difference of 2 errors and better than the sequential models. Figure 8.2 shows that my approaches

have performed much better than the $SEQ_{baseline}$ model.

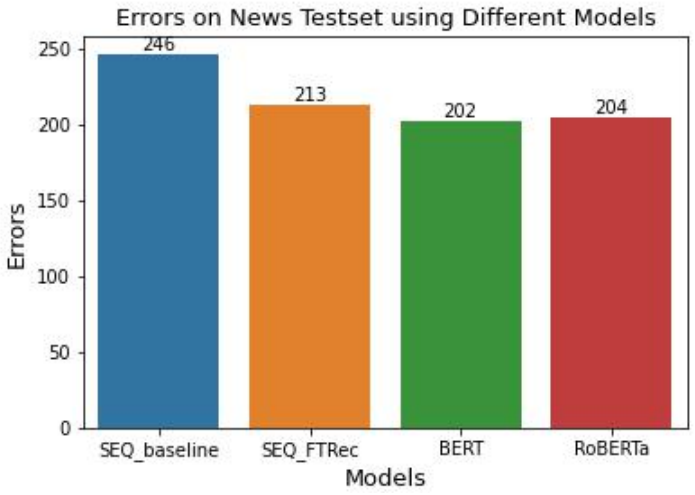

Figure 8.2: Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the News testset.

The $SEQ_{baseline}$ model has incorrectly predicted some of the **puctuation** like :, “, and ’ and almost all **words including hyphen** like *18-year-old* and *last-minute* as complex, which is improved by my sequential framework based approach SEQ_{FTRec} model as well as transformer-based models.

The sequential-based models were also prone to the name of a place, person, or object, which has been correctly predicted by the transformer-based language models. For instance, *Londonderry* and *Orontes*, which are a city in Ireland and a river in Syria respectively are predicted as complex words from the sequential models.

Train Text	**Test Text**	**Word Predictions**
Bellaghy parish priest **Fr** Andrew Dolan said the teenager died trying to protect her sister	Bellaghy **parish** priest Fr Andy Dolan said the teenager died trying to **protect** her sister.	**parish:** $SEQ_{baseline}$,$BERT$ **priest:** SEQ_{FTRec} **teenager:** All models **protect:** $SEQ_{baseline}$, $BERT$

Table 8.3: Similar text (bold words are annotated as complex) from News train and test set and predicted complex words from various models.

It has also been observed that there is some duplicate text in testset, which also exists in the trainset. Table 8.3 shows one of those instances along with the word predicted as complex from different models. I found that many words in a similar context have been annotated as complex as well as non-complex in training data. For instance, the word *teenager* in the given example text in the table has been annotated with both classes. This makes all models for this text to annotate *teenager* as complex even though

it was annotated non-complex in this particular test text. Similarly, SEQ_{FTRec} model has predicted the word *priest* as complex. However, other models predicted it as non-complex. Further from analysis, I found that whenever the word *priest* followed by the word *parish*, it has always been annotated as non-complex in training data. Most of the errors in this testset have been noticed due to the same word annotated as complex as well as non-complex.

8.2.2 WikiNews Testset

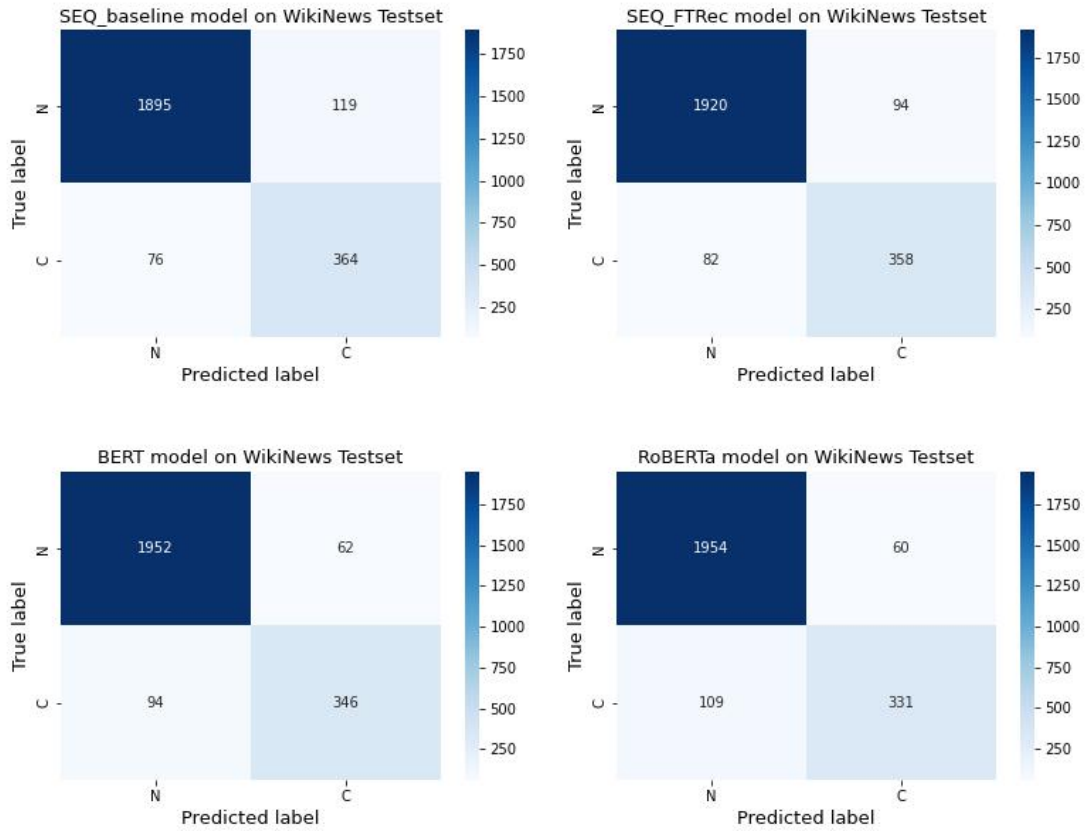

Figure 8.3: Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the WikiNews testset.

The confusion matrix for the WikiNews testset from various models is demonstrated in Figure 8.3. Similar to the News testset, the sequential models predicted the complex words better than the transformer-based pre-trained language models for the WikiNews testset with more errors while predicting the non-complex words. Figure 8.4 demonstrates the total number of errors made by these models for CWI in the WikiNews testset. The CWI using the $BERT$ model has achieved the lowest number of total errors. Also, my approaches have outperformed the $SEQ_{baseline}$ model with a big margin.

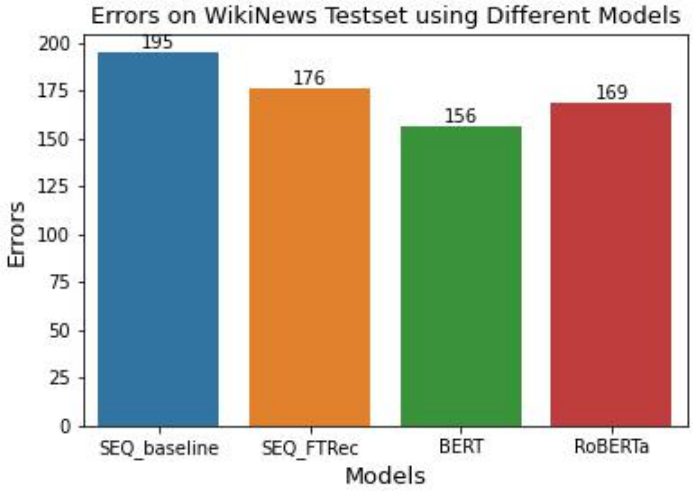

Figure 8.4: Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the WikiNews testset.

In the WikiNews testset as well, all the models have made similar errors as we have seen in the News testset. The sequential models were prone to words including hyphens like *President-Elect, mid-1990s* and *nine-and-a-half* while the $SEQ_{baseline}$ model specifically predicted some punctuation as complex. Also, I found that some of the words like *Apollo* have been incorrectly predicted as complex sometimes in test set, however, the same word was annotated as complex at both times of its existence in training data.

I also noticed that whilst some of the words like *bombers* are annotated as non-complex for all its existence in training data, the SEQ_{FTRec} model correctly predicted it as a complex word in testset. However, the transformer-based models have incorrectly predicted this word as non-complex. The question here this prediction raise is whether the transformer-based models predicted this word wrong or it was the sequential model as the models were trained to predict this word as non-complex. Anyway, this analysis showed that the annotation of some of the words for Shared Task data seems to be problematic.

In training data, I have also found many of the words like *aircraft, crash* and *pilot*, which are annotated as complex as well as non-complex in different places in a similar context. This led to many random predictions from models in testset. Further investigation can be performed to address this issue.

8.2.3 Wikipedia Testset

In Figure 8.3, I demonstrated the confusion matrix for the Wikipedia testset from various models. In this testset, the $RoBERTa$ model has significantly more errors in predicting the complex words. The $BERT$ model has the least number of overall errors with a large difference from other models. Figure 8.6 demonstrates the total number of errors made

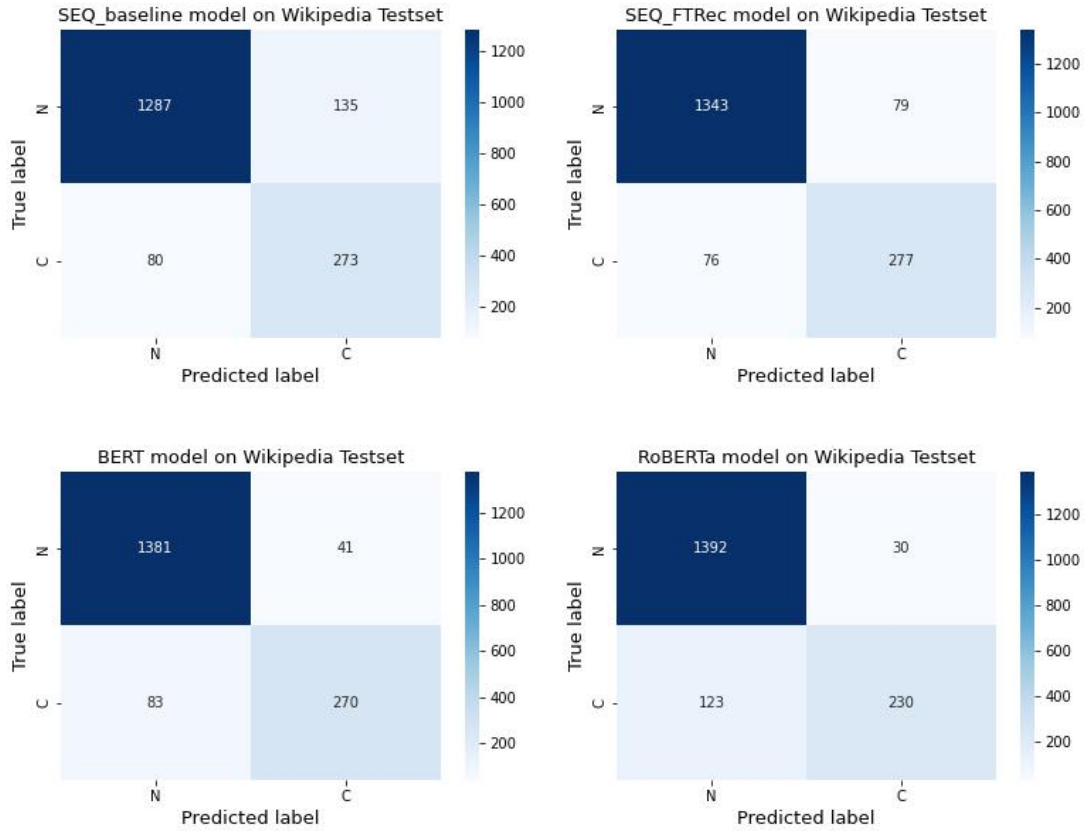

Figure 8.5: Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the Wikipedia testset.

by each model for CWI in the Wikipedia testset.

The sequential models have similar kinds of errors in this testset as well, as they achieved in other Shared Task testsets. The words like *define* and *specific* have been annotated as complex for all of their existence in training data. However, sometimes they have been annotated as non-complex in testset but all models have predicted them as complex. I also found some hyphens containing words like *best-known, late-Cretaceous, soft-blocked* and *light-years* as complex words in testset. The sequential model predicted these words as complex words as it does for most of the words containing a hyphen, even though if words are not complex. However, this time these words are also annotated as complex. Except for the word *light-years*, transformer-based models also predicted these hyphen-based complex words.

I have also found many unseen words on which the model was not trained. Table 8.4 shows two examples from the Wikipedia testset with complex annotated words in bold. All the models have correctly predicted the complex words in these sentences. The SEQ_{FTRec} model has incorrectly predicted the words *prepared*, *acid* and *solution* as complex, instead these words are non-complex in training data as well except for word

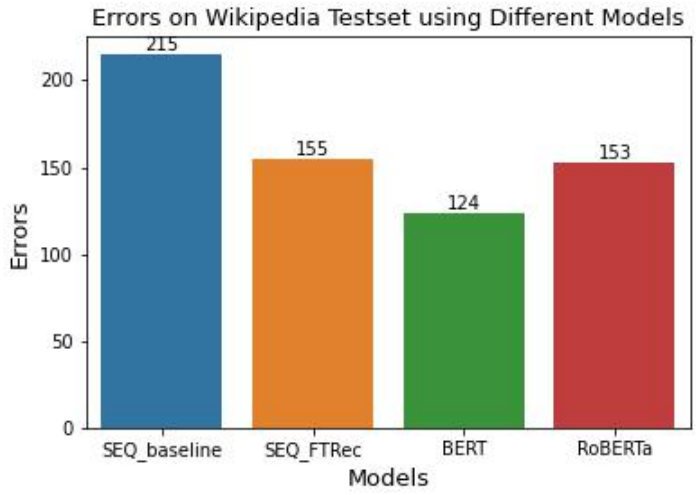

Figure 8.6: Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the Wikipedia testset.

solution, which is sometimes complex. The SEQ_{FTRec} model has been tuned to improve recall due to which it has more errors on predicting non-complex words.

Test Text	**Incorrect Predictions**
It is prepared by a **reaction** of **arsenic trichloride** and **potassium iodide** : **Hydrolysis** occurs only slowly in water forming **arsenic trioxide** and **hydroiodic** acid.	SEQ_{FTRec} : **prepared, acid** $SEQ_{baseline}$: **occurs**
The **aqueous** solution is highly **acidic**.	SEQ_{FTRec} : **solution**

Table 8.4: Text from Wikipedia testset (bold words are annotated as complex) and incorrect predictions from models.

8.2.4 InScript Testset

The confusion matrix for the InScript testset from various models is demonstrated in Figure 8.7. Similar to the previous testsets, the sequential models made more errors while predicting the non-complex words. Figure 8.8 demonstrates the total number of errors made by different models in the InScript testset. The $RoBERTa$ model has achieved the lowest number of total prediction errors. My approaches have made a very less number of errors in comparison to the $SEQ_{baseline}$ model. However, the $SEQ_{baseline}$ model has correctly predicted most of the complex words in comparison to the other models.

I analyzed the annotation of the InScript data and found that most of the simple words like *belong, self, cloth, such, setting, times, simply, found, able, round, sale, safety,*

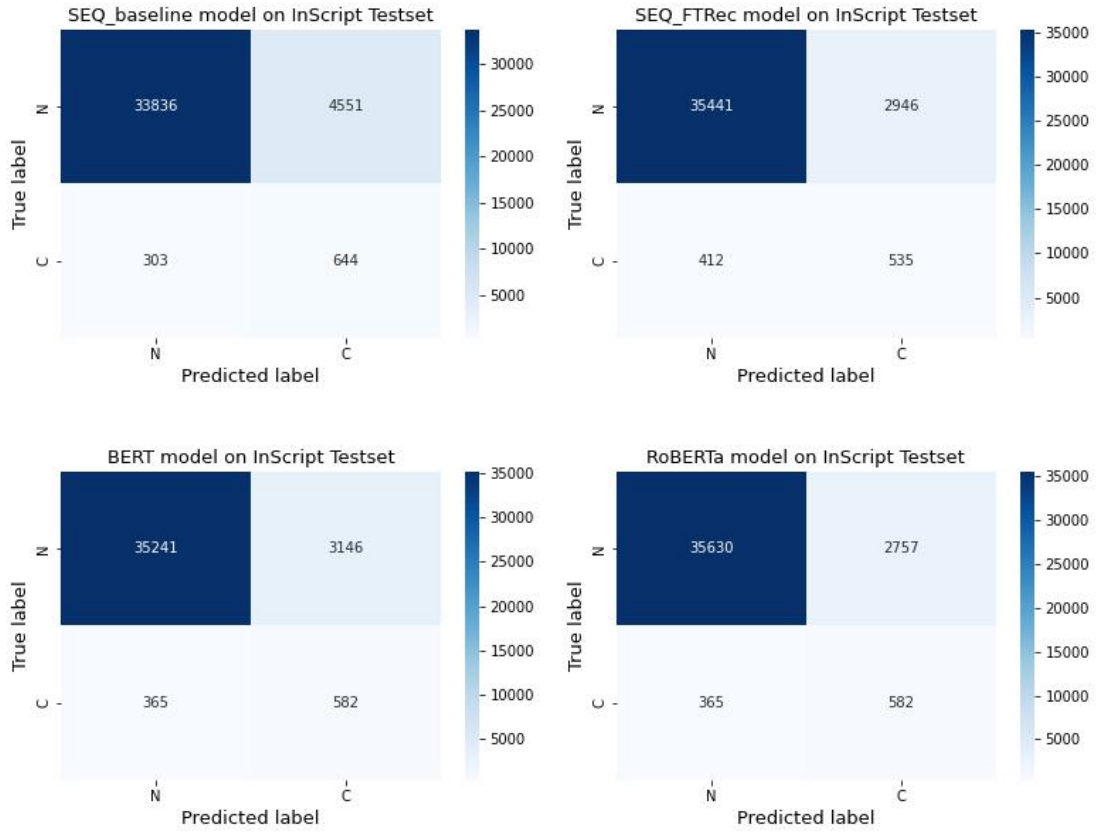

Figure 8.7: Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the InScript testset.

moving, days, deal, middle, and many more has been annotated as complex. I also found that many words which could be complex to second language learners like *sweaty, hamper, gingerly, accumulated, stopper, squirt, dripped, disrobed, faucet, suds, rag, rinse, greased, frosting, sift, blended, flimsy, stews,* and many more has been annotated as non-complex. These many errors on annotation ensure that the number of errors each model made cannot be used to judge the performance. So, I checked each model's prediction and compared them with the other three models' predictions along with self-feedback on the complexity of each word.

I found that most of the simple words and nouns like *bath, soap, tub, hot, finger, ears, toys, whales, bed, dinner, bus, card, hello,* and many more are predicted as complex using the $SEQ_{baseline}$ model. These errors ensure that we cannot use the $SEQ_{baseline}$ model outside the domain it was trained for. However, my approach of the sequential model and transformer-based language models has drastically reduced these kinds of errors and shows promising predictions. Although these models are fitted on Shared Task data, they correctly identified the complex words in this unseen data. For instance, words like *pavement, scraping, wobbly, deflated, sid, hamper, gingerly, squirt, lathered, shard, gauge*

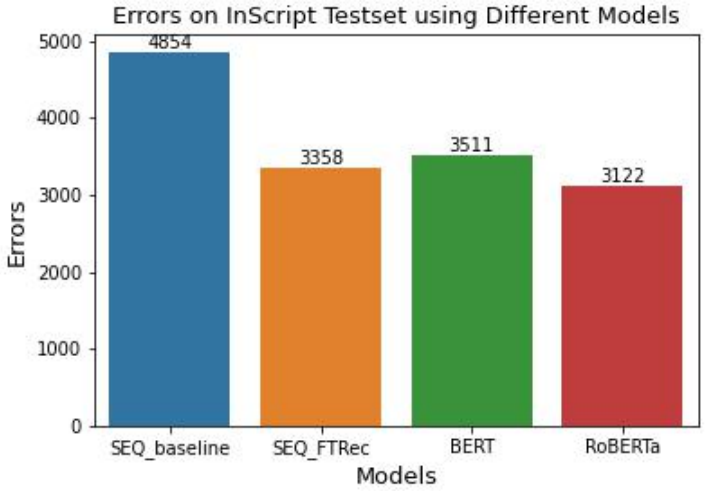

Figure 8.8: Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the InScript testset.

and *haphazardly* are predicted as complex by my approaches. However, the SEQ_{FTRec} model seems to have low performance as it predicted words like *everywhere, correct, pushed, driving* and *difficult* as complex and words like *soak, faucets, chores* and *glue* as non-complex.

8.2.5 OneStop Testset

The confusion matrix for the OneStop testset from various models is demonstrated in Figure 8.9. The sequential models made fewer errors while predicting the complex words and more errors while predicting the non-complex words. The total number of errors made by different models in the OneStop testset is demonstrated in Figure 8.10. The $RoBERTa$ model has achieved the lowest number of total prediction errors. However, it has the most errors while predicting the complex words in comparison to the other models. My approaches have made a very less number of errors in comparison to the $SEQ_{baseline}$ model.

Similar to InScript testset, most of the non-complex words like *largely, days, known, such, moving, planning, March, force, state, opening, base, whoever, remains, found, billion, setting, official, loss, safety, global,* and many more were annotated as complex in OneStop testset. Similarly, many complex words like *rainforest, charted, steered, cartography, telescopic, urbanites, traversing, cartographers, stalkers, sailed, reconnaissance, philanthropy, puddles, arson, coinciding, confederations,* and many more were annotated as non-complex.

The $SEQ_{baseline}$ model incorrectly predicted most of the words as complex. For instance, words like *map, Canadian, chosen, mail, computer, snow, selected, city, fires, bus, thats, changing, cheap, fish, taste, south-east, customers,* and many more are pre-

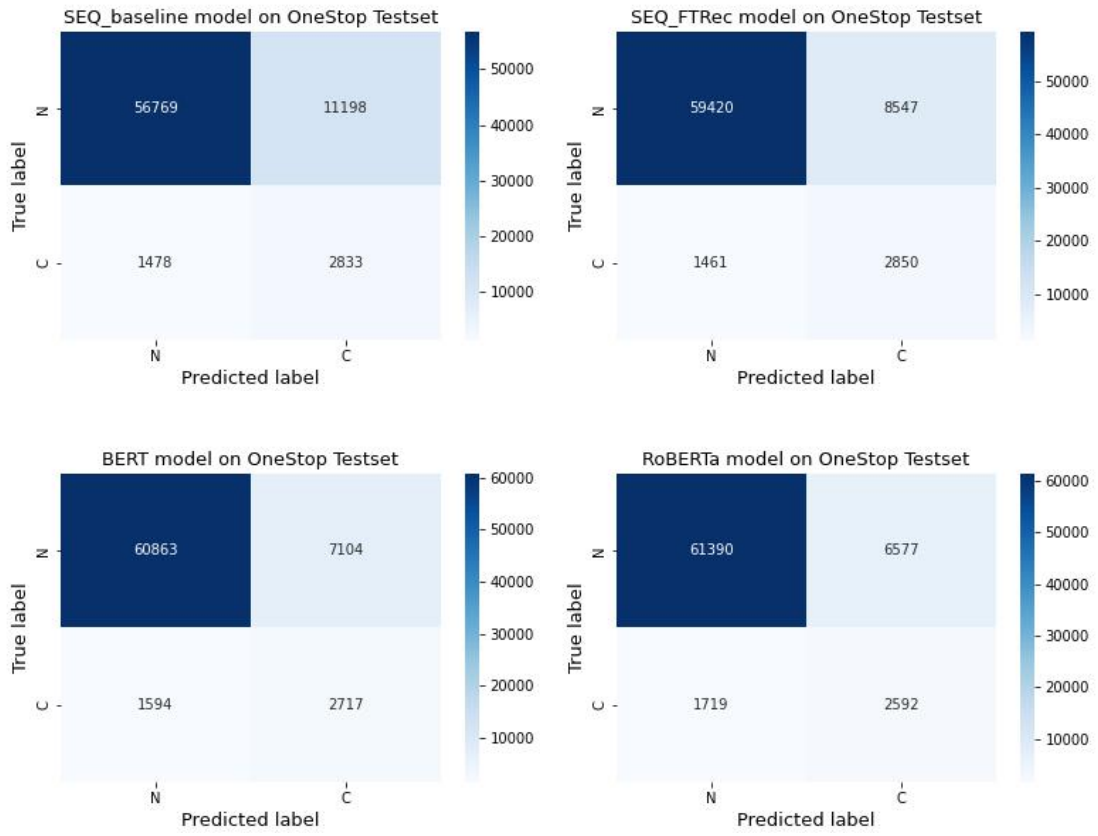

Figure 8.9: Confusion matrix generated from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models (left to right, top to bottom) on the OneStop testset.

dicted as complex using $SEQ_{baseline}$ model. The $SEQ_{baseline}$ model also predicted some punctuation like :, %, (, and) as complex. The SEQ_{FTRec} model does not predict extensive errors like the $SEQ_{baseline}$ model but it also predicted many non-complex words like *largely, application, collecting, project, little-known, understands, address, thinking, brought, sporting, burned, government, rise, completed, mountain, training,* and many more as complex. However, transformer-based pre-trained language models have performed much better on this testset, predicting almost all words complexity correctly.

Table 8.5 demonstrates an example text from the OneStop testset along with the complex words (annotated on the basis of CEFR level) in a bold letter in the text. The right column of the table displays the words which have been predicted as complex from the corresponding models. This is an advanced-level text from the OneStop testset. Most of the words were not annotated as complex, however, almost all models have correctly identified the complex words in this text. The sequential models have incorrectly identified simple words *brought* and *sporting* as complex. This ensures that my models trained on CWI Shared Task 2018 training data could be used for different genres, especially transformer-based pre-trained language models. Definitely, all models have achieved a

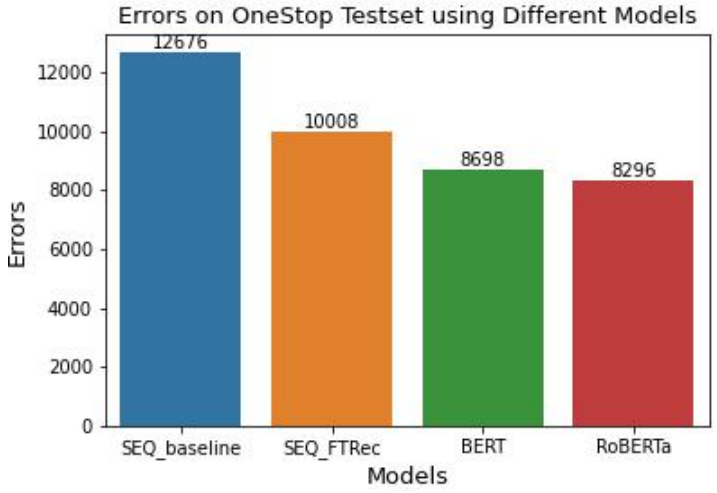

Figure 8.10: Comparison of total classification errors from $SEQ_{baseline}$, SEQ_{FTRec}, $BERT$, and $RoBERTa$ models on the OneStop testset.

low score in L2 testsets but it is due to the poor annotation of the L2 testsets.

Test Text	**Complex Word Predictions**
Coinciding with the start of the Confederations Cup a World Cup test event the rallies brought together a wide coalition of people **frustrated** with the escalating costs and persistently poor quality of public services, **lavish** investment in international sporting events, low standards of health care and wider unease about **inequality** and **corruption**.	*All models:* **Coinciding, Confederations, rallies, coalition, frustrated, escalating, persistently, investment, unease, inequality, corruption, lavish (except** $SEQ_{baseline}$**)** $SEQ_{baseline}$**: brought** SEQ_{FTRec}**: brought, sporting**

Table 8.5: Advanced level text (bold words are annotated complex) from OneStop test set and word complexity predictions from different models.

9 Conclusion and Future Work

In this chapter, I summarize my various approaches for the improvement of the CWI task. Further, I explain in detail the methodologies I adopted to answer the research questions I formulated in the Introduction (Section 1) and present the various challenges faced during the entire process. In Section 9.2, I present future directions that could be done to improve the model performance on the CWI for the second language learners.

9.1 Conclusion

In this thesis, I presented the hybrid approach and the transformers-based approaches for CWI to support English language learners. The hybrid approach was implemented above the sequential model by introducing the hand-engineered features. For all models, I used the training data from Shared Task 2018 datasets to fit the model. For my experiments, I collected the L2 data which is more common for language learners and they come across to them frequently from InScript and OneStop corpus. I considered this dataset as an additional testsets, which I simply annotated on the basis of CEFR levels.

All of my approaches outperform the previous state-of-the-art CWI systems trained on Shared Task 2018 datasets. As a first approach, I included FastText word embeddings instead of GloVe word embeddings and tuned the sequential model by Rei (2017) to improve the recall. The model benefits from the more sophisticated FastText word embeddings as expected but tuning the model to improve recall, improved the overall performance of the model without losing much precision. This could be due to the reason that the data is highly imbalanced with a very less number of complex words in comparison to the non-complex words.

To answer my first research question (RQ1), i.e., whether the hybrid approach outperforms the sequential model for the CWI task, I have identified thirteen hand-engineered features used by previous researchers to provide features based state-of-the-art systems for the CWI. I introduced each feature individually in the SEQ_{FTRec} model and performed the training. The results achieved from these experiments are almost similar to the one I achieved without introducing any features and they might converge to the same values if I take the average of more experiments. Further, from these experiments, I identified five features that have slightly better results for at least three testsets and provide some external information that the model can not see in the training examples. I introduced these five features together in the sequential model while training the model. There was no significant improvement in model performance using this approach as well. Finally, I incorporated the permutation importance technique to identify the most important features. I have selected the top two features from this technique and introduced them in the sequential model training. The results achieved from this experiment as well

did not show any performance improvement of the model. This indicates that the neural network can learn most of these features along with many others in its black box by itself to give better results. ***These approaches answer my first research question as introducing the hand-engineered features to build the hybrid model does not help the sequential model learning to increase performance.*** However, I identified some features, which individually provide slightly better results for some testsets or together provide almost the same performance as from the sequential model. This knowledge could be useful while training the model for bigger datasets where these features could help to boost the learning in initial network training.

To answer my second research question (RQ2), i.e., whether the transformer-based models outperforms the sequential model for the CWI task, I have fine-tuned pre-trained transformer-based language models RoBERTa and BERT on Shared Task 2018 data for CWI. Whilst for RoBERTa I consider the *roberta-large* model, for BERT I consider the *bert-base-cased* model. I also prepared new training data, which does not include any stop words and punctuation, resulting in the deduction of non-complex words and reducing the imbalance data. I fine-tuned the RoBERTa model using both training data and compared the performances. RoBERTa model fine-tuned on the new training data performed worse than the original training data ensuring that removal of stop words would not be a good idea for the CWI system. The reason behind these lower scores could be due to the pre-trained language model that we are using is trained using stop words and removal of them has an adverse impact on the encoded representation of the sequence. As we are feeding full-sentence sequence as input to this model, feeding similar data to this language model on which it has been pre-trained would perform better.

Whilst the RoBERTa model has achieved the highest F-Scores for WikiNews, InScript, and OneStop testsets, the BERT model has achieved the highest F-Scores for the remaining two News and Wikipedia testsets. The BERT model has achieved an exceptionally high score for the Wikipedia testset on which the other approaches from the sequential model and RoBERTa struggles. This could be due to the reason that the BERT model has been pre-trained only on BookCorpus and **English Wikipedia** amounting to a total of 16 GB, which does not make it much diverse like the RoBERTa model, which has been pre-trained on many other genres amounting to a total of 160 GB data. ***The results achieved from these models answer my second research question that transformer-based models outperform the sequential model for CWI task for all testsets, including the L2 target domain.*** The results achieved from the BERT base model for CWI ensure that we do not need a bigger network for the CWI task and using BERT we can achieve the state-of-the-art results for it.

To answer my third research question (RQ3), i.e., whether the models trained on CWI Shared Task data are good enough for second language learners, I tested performance on the L2 dataset from all the models trained using the Shared Task 2018 dataset in this thesis. The F-Scores achieved for this dataset are very low in comparison to the Shared Task testsets. However, the L2 data has not been annotated manually instead annotated on the basis of CEFR levels. All the words from L2 data which has a CEFR level below $B2$ or are not present in CEFR vocabulary are annotated as non-complex words. This also makes many words below $B2$ CEFR level which could be complex for

second language learners, annotated as non-complex by the program. Also, during error analysis, I found that most of the words are incorrectly annotated as complex as well. This ensures us not have high expectations from the model for good performance in the L2 dataset. From the error analysis, I noticed that the transformer-based language models on the L2 dataset have performed better than the sequential models and have precisely identified the complex words on these unseen data. ***Error analysis on the model predictions answers my third research question that the models trained on Shared Task data, specially transformer-based models are good enough for second language learners.*** The RoBERTa model has achieved the best performance for the L2 dataset, significantly better than all the other approaches. This could be due to the reason that the pre-trained RoBERTa model has been trained using many different genres corpus and it might be helping the RoBERTa model to achieve better performance on the L2 dataset.

9.2 Future Work

The L2 dataset collected in this thesis is annotated on the basis of CEFR levels. This technique defined the word complexity without taking the context where it occurred. Also, many words were not present in their EVP word lists and I annotated all of them as non-complex. These limitations with limited time did not allow us to train the model in this dataset. Also, the performance achieved on this dataset from various models is not up to the mark. In the future, the annotation for this dataset could be improved by involving some English language learners using the Amazon Mechanical Turk or some other platform. The results achieved on the L2 dataset from the pre-trained transformer-based RoBERTa language model are exceptionally well in comparison to other models including BERT. The better annotation of the L2 dataset would allow us to conclude whether the RoBERTa model is getting benefit from its pre-trained approach on many different genres corpus amounting to a ton of data in comparison to the BERT model.

From my experiment of fine-tuning RoBERTa model, I found that removing stop words from the training data degraded the model performance. The reason could be that transformer-based language models are pre-trained using stop words and removal of them has an adverse impact on the encoded representation of the sequence. In future work, instead of removing stop words, we can mask them before feeding them to transformer-based models. This way we could achieve the original goal of removing stop words, as well as a transformer will implicitly guess these masked tokens. Also, we will try to use an ensemble of various pre-trained language models fine-tuned for the CWI task.

Bibliography

AbuRa'ed, A. and Saggion, H. 2018. "LaSTUS/TALN at Complex Word Identification (CWI) 2018 Shared Task." In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications,* 159–165. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-0517. https://aclanthology.org/W18-0517.

Bada, M., Eckert, M., Evans, D., Garcia, K., Shipley, K., Sitnikov, D., Baumgartner Jr, W., et al. 2012. "Concept annotation in the CRAFT corpus." *BMC bioinformatics* 13:161. https://doi.org/10.1186/1471-2105-13-161.

Bani Yaseen, T., Ismail, Q., Al-Omari, S., Al-Sobh, E., and Abdullah, M. 2021. "JUST-BLUE at SemEval-2021 Task 1: Predicting Lexical Complexity using BERT and RoBERTa Pre-trained Language Models." In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021),* 661–666. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.semeval-1.85. https://aclanthology.org/2021.semeval-1.85.

Bingel, J., Schluter, N., and Martınez Alonso, H. 2016. "CoastalCPH at SemEval-2016 Task 11: The importance of designing your Neural Networks right." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 1028–1033. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1160. https://aclanthology.org/S16-1160.

Bird, S. and Loper, E. 2004. "NLTK: The Natural Language Toolkit." In *Proceedings of the ACL Interactive Poster and Demonstration Sessions,* 214–217. Association for Computational Linguistics. https://aclanthology.org/P04-3031.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. 2016. "Enriching Word Vectors with Subword Information." *CoRR* abs/1607.04606. arXiv: 1607.04606. http://arxiv.org/abs/1607.04606.

Brooke, J., Uitdenbogerd, A., and Baldwin, T. 2016. "Melbourne at SemEval 2016 Task 11: Classifying Type-level Word Complexity using Random Forests with Corpus and Word List Features." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 975–981. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1150. https://aclanthology.org/S16-1150.

Christodouloupoulos, C. and Steedman, M. 2015. "A Massively Parallel Corpus: The Bible in 100 Languages." *Lang. Resour. Eval.* 49 (2): 375–395. ISSN: 1574-020X. https://doi.org/10.1007/s10579-014-9287-y. https://doi.org/10.1007/s10579-014-9287-y.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. 2020. "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators." *CoRR* abs/2003.10555. arXiv: 2003.10555. https://arxiv.org/abs/2003.10555.

Coster, W. and Kauchak, D. 2011. "Simple English Wikipedia: A New Text Simplification Task." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,* 665–669. Association for Computational Linguistics. https://aclanthology.org/P11-2117.

Council of Europe. 2011. "Common European Framework of Reference for Languages: Learning, Teaching, Assessment," https://www.coe.int/en/web/common-european-framework-reference-languages.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. 2018. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." *CoRR* abs/1810.04805. arXiv: 1810.04805. http://arxiv.org/abs/1810.04805.

Goldberg, Y. and Orwant, J. 2013. "A Dataset of Syntactic-Ngrams over Time from a Very Large Corpus of English Books." In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity,* 241–247. Association for Computational Linguistics. https://aclanthology.org/S13-1035.

Gooding, S. and Kochmar, E. 2018. "CAMB at CWI Shared Task 2018: Complex Word Identification with Ensemble-Based Voting." In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications,* 184–194. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-0520. https://aclanthology.org/W18-0520.

———. 2019. "Complex Word Identification as a Sequence Labelling Task." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 1148–1153. Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1109. https://www.aclweb.org/anthology/P19-1109.

Guo, M., Dai, Z., Vrandečić, D., and Al-Rfou, R. 2020. "Wiki-40B: Multilingual Language Model Dataset" [in English]. In *Proceedings of the 12th Language Resources and Evaluation Conference,* 2440–2452. European Language Resources Association. ISBN: 979-10-95546-34-4. https://aclanthology.org/2020.lrec-1.297.

He, P., Liu, X., Gao, J., and Chen, W. 2020. "DeBERTa: Decoding-enhanced BERT with Disentangled Attention." *CoRR* abs/2006.03654. arXiv: 2006.03654. https://arxiv.org/abs/2006.03654.

Hochreiter, S. and Schmidhuber, J. 1997. "Long Short-term Memory." *Neural computation* 9:1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.

Horn, C., Manduca, C., and Kauchak, D. 2014. "Learning a Lexical Simplifier Using Wikipedia." In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers),* 458–463. Association for Computational Linguistics. https://doi.org/10.3115/v1/P14-2075. https://aclanthology.org/P14-2075.

Kajiwara, T. and Komachi, M. 2018. "Complex Word Identification Based on Frequency in a Learner Corpus." In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications,* 195–199. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-0521. https://aclanthology.org/W18-0521.

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

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. 2017. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." In *Proceedings of the 31st International Conference on Neural Information Processing Systems,* 3149–3157. NIPS'17. Curran Associates Inc. ISBN: 9781510860964.

Koehn, P. 2005. "Europarl: A Parallel Corpus for Statistical Machine Translation." In *Proceedings of Machine Translation Summit X: Papers,* 79–86. https://aclanthology.org/2005.mtsummit-papers.11.

Konkol, M. 2016. "UWB at SemEval-2016 Task 11: Exploring Features for Complex Word Identification." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 1038–1041. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1162. https://aclanthology.org/S16-1162.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. 2019. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations." *CoRR* abs/1909.11942. arXiv: 1909.11942. http://arxiv.org/abs/1909.11942.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. 2019. "RoBERTa: A Robustly Optimized BERT Pretraining Approach." *CoRR* abs/1907.11692. arXiv: 1907.11692. http://arxiv.org/abs/1907.11692.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013. "Efficient Estimation of Word Representations in Vector Space." *CoRR* abs/1301.3781. arXiv: 1301.3781. http://arxiv.org/abs/1301.3781.

Miller, G. A. 1995. "WordNet: A Lexical Database for English." 38 (11): 39–41. ISSN: 0001-0782. https://doi.org/10.1145/219717.219748.

Mizumoto, T., Komachi, M., Nagata, M., and Matsumoto, Y. 2011. "Mining Revision Log of Language Learning SNS for Automated Japanese Error Correction of Second Language Learners." In *Proceedings of 5th International Joint Conference on Natural Language Processing,* 147–155. Asian Federation of Natural Language Processing. https://aclanthology.org/I11-1017.

Modi, A., Anikina, T., Ostermann, S., and Pinkal, M. 2017. "InScript: Narrative texts annotated with script information." *CoRR* abs/1703.05260. arXiv: 1703.05260. http://arxiv.org/abs/1703.05260.

Mosquera, A. 2021. "Alejandro Mosquera at SemEval-2021 Task 1: Exploring Sentence and Word Features for Lexical Complexity Prediction." In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021),* 554–559. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.semeval-1.68. https://aclanthology.org/2021.semeval-1.68.

Mukherjee, N., Patra, B. G., Das, D., and Bandyopadhyay, S. 2016. "JU_NLP at SemEval-2016 Task 11: Identifying Complex Words in a Sentence." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 986–990. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1152. https://aclanthology.org/S16-1152.

Nation, I. 2006. "How large a vocabulary is needed for reading and listening?" *Canadian modern language review* 63 (1): 59–82. https://doi.org/10.3138/cmlr.63.1.59.

Paetzold, G. and Specia, L. 2016d. "Collecting and Exploring Everyday Language for Predicting Psycholinguistic Properties of Words." In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers,* 1669–1679. The COLING 2016 Organizing Committee. https://aclanthology.org/C16-1157.

———. 2016a. "Plumberr: An automatic error identification framework for lexical simplification." In *Proceedings of the first international workshop on Quality Assessment for Text Simplification (QATS),* 1–9. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2016/workshops/LREC2016Workshop-QATS_Proceedings.pdf.

———. 2016b. "SemEval 2016 Task 11: Complex Word Identification." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 560–569. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1085. https://aclanthology.org/S16-1085.

———. 2016c. "SV000gg at SemEval-2016 Task 11: Heavy Gauge Complex Word Identification with System Voting." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 969–974. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1149. https://aclanthology.org/S16-1149.

Paetzold, G. H. and Specia, L. 2017. "A Survey on Lexical Simplification." 60 (1): 549–593. ISSN: 1076-9757.

Palakurthi, A. and Mamidi, R. 2016. "IIIT at SemEval-2016 Task 11: Complex Word Identification using Nearest Centroid Classification." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 1017–1021. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1158. https://aclanthology.org/S16-1158.

Pennington, J., Socher, R., and Manning, C. 2014. "GloVe: Global Vectors for Word Representation." In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 1532–1543. Association for Computational Linguistics. https://doi.org/10.3115/v1/D14-1162. https://aclanthology.org/D14-1162.

Petersen, S. E. and Ostendorf, M. 2007. "Text Simplification for Language Learners: A Corpus Analysis." In *In Proceedings of Workshop on Speech and Language Technology for Education.* Citeseer. https://isca-speech.org/archive_open/archive_papers/slate_2007/sle7_069.pdf.

Quijada, M. and Medero, J. 2016. "HMC at SemEval-2016 Task 11: Identifying Complex Words Using Depth-limited Decision Trees." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 1034–1037. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1161. https://aclanthology.org/S16-1161.

Rei, M. 2017. "Semi-supervised Multitask Learning for Sequence Labeling." *CoRR* abs/1704.07156. arXiv: 1704.07156. http://arxiv.org/abs/1704.07156.

Ronzano, F., Abura'ed, A., Espinosa-Anke, L., and Saggion, H. 2016. "TALN at SemEval-2016 Task 11: Modelling Complex Words by Contextual, Lexical and Semantic Features." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 1011–1016. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1157. https://aclanthology.org/S16-1157.

Rotaru, A. 2021. "ANDI at SemEval-2021 Task 1: Predicting complexity in context using distributional models, behavioural norms, and lexical resources." In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021),* 655–660. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.semeval-1.84. https://aclanthology.org/2021.semeval-1.84.

Shardlow, M. 2014. "Out in the Open: Finding and Categorising Errors in the Lexical Simplification Pipeline." In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14),* 1583–1590. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2014/pdf/479_Paper.pdf.

———. 2013a. "A Comparison of Techniques to Automatically Identify Complex Words." In *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop,* 103–109. Association for Computational Linguistics. https://www.aclweb.org/anthology/P13-3015.

———. 2013b. "The CW Corpus: A New Resource for Evaluating the Identification of Complex Words." In *Proceedings of the Second Workshop on Predicting and Improving Text Readability for Target Reader Populations,* 69–77. Association for Computational Linguistics. https://aclanthology.org/W13-2908.

Shardlow, M., Cooper, M., and Zampieri, M. 2020. "CompLex — A New Corpus for Lexical Complexity Prediction from Likert Scale Data." In *Proceedings of the 1st Workshop on Tools and Resources to Empower People with REAding DIfficulties (READI),* 57–62. European Language Resources Association. https://aclanthology.org/2020.readi-1.9.

Shardlow, M., Evans, R., Paetzold, G. H., and Zampieri, M. 2021. "SemEval-2021 Task 1: Lexical Complexity Prediction." In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021),* 1–16. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.semeval-1.1. https://aclanthology.org/2021.semeval-1.1.

Song, B., Pan, C., Wang, S., and Luo, Z. 2021. "DeepBlueAI at SemEval-2021 Task 7: Detecting and Rating Humor and Offense with Stacking Diverse Language Model-Based Methods." In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021),* 1130–1134. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.semeval-1.158. https://aclanthology.org/2021.semeval-1.158.

Speer, R., Chin, J., and Havasi, C. 2016. "ConceptNet 5.5: An Open Multilingual Graph of General Knowledge." *CoRR* abs/1612.03975. arXiv: 1612.03975. http://arxiv.org/abs/1612.03975.

Taya, Y., Kanashiro Pereira, L., Cheng, F., and Kobayashi, I. 2021. "OCHADAI-KYOTO at SemEval-2021 Task 1: Enhancing Model Generalization and Robustness for Lexical Complexity Prediction." In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021),* 17–23. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.semeval-1.2. https://aclanthology.org/2021.semeval-1.2.

Vajjala, S. and Lučić, I. 2018. "OneStopEnglish corpus: A new corpus for automatic readability assessment and text simplification." In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications,* 297–304. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-0535. https://aclanthology.org/W18-0535.

Wilson, M. 1988. "MRC psycholinguistic database: Machine-usable dictionary, version 2.00." *Behavior Research Methods, Instruments, & Computers* 20. https://doi.org/10.3758/BF03202594.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., et al. 2019. "HuggingFace's Transformers: State-of-the-art Natural Language Processing." *CoRR* abs/1910.03771. arXiv: 1910.03771. http://arxiv.org/abs/1910.03771.

Wróbel, K. 2016. "PLUJAGH at SemEval-2016 Task 11: Simple System for Complex Word Identification." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 953–957. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1146. https://aclanthology.org/S16-1146.

Yimam, S. M., Biemann, C., Malmasi, S., Paetzold, G., Specia, L., Štajner, S., Tack, A., and Zampieri, M. 2018. "A Report on the Complex Word Identification Shared Task 2018." In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications,* 66–78. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-0507. https://www.aclweb.org/anthology/W18-0507.

Yimam, S. M., Štajner, S., Riedl, M., and Biemann, C. 2017. "Multilingual and Cross-Lingual Complex Word Identification." In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017,* 813–822. INCOMA Ltd. https://doi.org/10.26615/978-954-452-049-6_104. https://doi.org/10.26615/978-954-452-049-6_104.

Zampieri, M., Tan, L., and Genabith, J. van. 2016. "MacSaar at SemEval-2016 Task 11: Zipfian and Character Features for ComplexWord Identification." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016),* 1001–1005. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-1155. https://aclanthology.org/S16-1155.

Versicherung an Eides statt

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit im Masterstudiengang Intelligent Adaptive Systems selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.

Ich stimme der Einstellung der Arbeit in die Bibliothek des Fachbereichs Informatik zu.

Hamburg, den March 13, 2022

Ankit Srivastava