RS_GV at SemEval-2021 Task 1: Sense Relative Lexical Complexity Prediction

Regina Stodden
Dept. of Computational Linguistics
Heinrich Heine University
Düsseldorf, Germany
regina.stodden@hhu.de

Gayatri Venugopal
Symbiosis Institute of
Computer Studies and Research
Symbiosis International (Deemed University)
Pune, Maharashtra, India
gayatri.venugopal@sicsr.ac.in

Abstract

We present the technical report of the system called RS_GV at SemEval-2021 Task 1 on complexity prediction of English words. RS_GV is a neural network using hand-crafted linguistic features in combination with character and word embeddings to predict the target words' complexity. For the generation of the hand-crafted features, we set the target words in relation to their senses. RS_GV predicts the complexity well of biomedical terms but it has problems with the complexity prediction of very complex and very simple target words.

1 Introduction

Text simplification is the process of modifying a text so that it becomes easy for the reader to understand the meaning of the text without any loss of information. A main part of text simplification is lexical simplification. In lexical simplification, complex words are replaced with easier or more frequent synonyms. Following Shardlow (2014), the process of lexical simplification can be split as follows: I.) identification of complex words in a given text, II.) substitution generation, III.) word sense disambiguation, IV.) synonym ranking, V.) substitution of complex word with the best synonym in correct morphological form.

Following Shardlow (2014), the most common errors in lexical simplification are that the words are not identified as complex or that words are incorrectly identified as complex. One reason might be the approach to predict complex words. So far, in the task called *complex word identification* (CWI), a word in a sentence was labeled as either complex or simple without any range in between. Shardlow et al. (2020) criticize this approach because there is no clear threshold when a word starts to be complex. Hence, they propose a new task called *lexical complexity prediction* (LCP). The aim of LCP is to predict the complexity of a single word or a multi-word expression on a scale of 0 to 1.

This paper proposes RS_GV , a model for LCP in the context of the SemEval-2021 task 1 (Shardlow et al., 2021a). RS_GV uses hand-crafted features relative to their WordNet senses, Flair embeddings and a neural regressor in a cross-domain and within-domain setting.

2 Related Work

Lexical complexity prediction is a new sub-task of lexical text simplification. The aim is to predict the complexity of a single word or a multiword expression on a scale of 0 to 1. The most similar task is CWI. In contrast to LCP, CWI aims at binary classification that determines whether a word is complex or not. As LCP has been mentioned for the first time in the context of this shared task (Shardlow et al., 2020, 2021a,b), no other related work exists yet. Hence, we outline the state of the art in CWI.

SemEval-2016 Task 11: CWI Paetzold and Specia (2016) collated 9200 sentences from the CW Corpus (Shardlow, 2013), the LexMTurk Corpus (Horn et al., 2014), and the Simple Wikipedia corpus (Kauchak, 2013). All these corpora were based on the Simple English Wikipedia (SEW). CWI was treated as a binary classification task, wherein 400 non-native speakers annotated content words in English text. It was observed from the annotations that complex words were shorter, less ambiguous and had a low occurrence in SEW. F-score and G-score were used as the evaluation metrics. The features incorporated by the submitted systems can be seen in Figure 1.

It is shown that the word frequency, lexical, semantic and morphological features play a dominant role in CWI. Besides these, n-gram features were also experimented with by a few systems. Word embeddings were not used extensively.

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 640–649
Bangkok, Thailand (online), August 5–6, 2021.

CWI Shared Task 2018 Another shared task on complex word identification was organized in 2018 (Yimam et al., 2018). Yimam et al. (2018) collected data from three sources, i.e., professionally written news, WikiNews and Wikipedia, and in four languages, i.e., English, German, French, and Spanish. The shared task was composed of two sub-tasks. Sub-task 1 approached the problem as a binary classification problem and sub-task 2 treated it as a probabilistic classification problem, wherein the score between 0 and 1 indicated the proportion of annotators who considered a word as complex. Native as well as non-native readers annotated the dataset created by Yimam et al. (2017). A word was deemed to be complex if at least one out of twenty annotators labeled it as complex. Based on annotations, it was observed that the systems might perform better when trained on domain-specific data. It was also found that traditional feature engineering-based approaches performed better than neural network and word embedding based approaches. The features incorporated by the submitted systems of 11 teams can be seen in Figure 1.

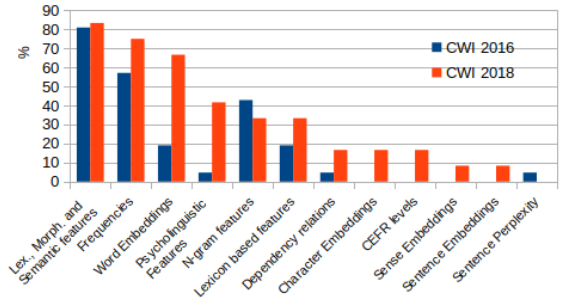

Figure 1: Features incorporated by the systems submitted to CWI Shared Task 2016 (Paetzold and Specia, 2016) and 2018 (Yimam et al., 2018).

The graph reinstates the fact that frequencies, lexical, semantic and morphological features play a key role in CWI. However, it was observed that as compared to 2016, in 2018, word embeddings were more commonly used.

3 Experimental Setup

3.1 Data

The corpus (Shardlow et al., 2020, 2021b) contains 9,476 annotated instances in three new CWI/LCP domains, i.e., bible, political and biomedical texts. For every instance, one target word, its target complexity value and its containing sentence are given. The complexity value is based on crowd-sourced human ratings of at least 4 and at most 20 persons with residence in the UK, USA, or Australia. Each instance was rated on a 5-point Likert scale from 1 (very easy) to 5 (very difficult). Afterwards, the ratings were averaged and normalized on a continuous scale between 0 and 1, where 0 is easy and 1 is complex.

Each target word occurs in multiple instances and may capture different senses so that each word can be assigned to different complexity values in different instances. For example, *vision* occurs in all sub-domains with different meaning, e.g., ability to see, supernatural experience, and foresight.

Following the corpus description (Shardlow et al., 2020), a target word should only occur in a different sentence but not in the same sentence twice. Unfortunately, in our corpus analysis, we found a few doubled instances but with varying complexity values. For example, *body* is rated within in the same sentence in the biomedical part of the set with complexity values of 0.05 and 0.32 (see Appendix C, Table 9). This variation underlines that LCP is a subjective task, and, hence, a difficult NLP task (see section 5.3).

More details regarding the data, including the data split in training, trial, and test can be found in the shared task paper (Shardlow et al., 2021a).

As a preprocessing step we tokenized the sentences and annotated the tokens with their lemma, part-of-speech, and morphological information using spaCy (Honnibal and Montani, 2017). This linguistic information is the basis of our features.

3.2 Evaluation

The lexical complexity prediction is evaluated, following the shared task instructions (Shardlow et al., 2021a), with e.g., Pearson's correlation (r, mainly reported here) and Mean Absolute Error (MAE).

3.3 Baselines

We use the baseline results reported by the organizers[1] as comparative results. They use linear regression models with the following features, complexity-average, word length, log word frequency from SUBTLEX and log word frequency combined with word length.

[1] https://competitions.codalab.org/competitions/27420#learn_the_details-evaluation

4 System Description

Our system's main characteristics are a combination of hand-crafted features, contextualized character embeddings (see subsection 4.1), a sense relative normalization (see subsection 4.2), and a neural network for regression (see subsection 4.4).

4.1 Features

Based on the survey of features previously used for complexity estimation of words (see section 2), we decided to combine hand-crafted features and contextualized embeddings. A list of all language resources used for feature generation is provided in Appendix B (see Table 7).

4.1.1 Word and Character Embeddings

Similar as proposed in Gooding and Kochmar (2019); Hartmann and dos Santos (2018), and De Hertog and Tack (2018) we use word and character embeddings. We compare pre-trained non-contextualized word embeddings, i.e., `GloVe` (Pennington et al., 2014), pre-trained contextualized word embeddings, i.e., `ELMo` (Peters et al., 2018) and `BERT` (Devlin et al., 2019), with pre-trained contextualized character embeddings, i.e., stacked `Flair` (Akbik et al., 2018, 2019a) –a combination of GloVe and Flair– and `PooledFlair` (Akbik et al., 2019b).

We suggest that the contextualized embeddings perform better on LCP as the context of the target word and the meaning of the sentence are important for words' complexity. To the best of our knowledge contextualized character embeddings have not been used for CWI or LCP before.

The embedddings are extracted using FLAIR (Akbik et al., 2019a). Details regarding the settings of the word and character embeddings are provided in Appendix B (see Table 8).

4.1.2 Hand-crafted Features

An overview of all hand-crafted features used is visualized in Table 1.

Readability Assessment Features. We use the sentence's readability as a feature because we assume that a token would be perceived as more complex if the entire sentence is complex. We implemented the readability using readability scores which are mainly applicable on texts such as Kincaid et al. (1975), Gunning (1952), Coleman and Liau (1975), Dale and Chall (1948) and Senter and Smith (1967) using textstat (Bansal and Aggarwal, 2014). We do not consider readability scores that are applicable on sentences as we could not reproduce certain sentence-level readability methods.

category	feature	category	feature
Readability Assessment	flesch kincaid grade	Morphological	proper noun
	gunning fog		singular
	coleman liau index		plural
	dale chall readability score		famsize
	automated readability index		HAL frequency
	difficult words		number morphems
Lexical	∗frequency		number prefixes
	∗word length		number roots
	∗number consonants		number suffixes
	∗number vowels		suffix length
	∗number syllables		prefix length
WordNet	∗number hypernyms	Psycholinguistic	∗familiarity
	∗number hyponyms		∗concreteness
	∗number senses		∗imagery
Lexicon	in wordlists		∗m.fullness colerado
Other	named entity		∗m.fullness pavio
	word position		∗age of acquisition

Table 1: List of all used features sorted by category. An asterisk (∗) indicates whether the feature is normalized relative to its senses or not.

Lexical Features. Word length, word frequency and number of syllables are included in the set of lexical features following the methodology explained in Shardlow et al. (2020). The word frequency values are obtained from Sharoff (2006) and the GoogleWeb1T resource (Brants, Thorsten and Franz, Alex, 2006). Besides these, the number of consonants and vowels are also calculated.

WordNet Features. Paetzold and Specia (2016) use the number of senses, synonyms, hypernyms and hyponyms among other features to identify complex words. In our study, the number of hypernyms, hyponyms and senses are retrieved from the English WordNet (Fellbaum, 1998).

Psycholinguistic Features. Similarly as proposed in Davoodi and Kosseim (2016), we generate psycholinguistic features, e.g., word familiarity and age of acquisition, using the Medical Research Council (MRC) Psycholinguistic Database version 2.0 (Wilson, 1988).

Morphological Features. As seen in the survey of CWI shared task, morphological features are often used for this task. Hence, we also use a few morphological features derived by the morphological database MorphoLex-EN (Sánchez-Gutiérrez et al., 2018), e.g., number of prefixes, morphemes, and suffixes. We assume the more morphological rich, the more complicated the word.

Lexicon-based Features. As, for example, proposed in AbuRa'ed and Saggion (2018), and Wani et al. (2018), we check if the target word is con-

tained in the Oxford 3000 word list (Dictionaries, 2021) with commonly used words. We assume the more common a word is, the simpler it would be.

Other Features. Since it is expected that the corpus contains a lot of named entities, such as person names in the bible subcorpus, we check if a target word is a named entity, as also suggested in Gooding and Kochmar (2018).
The last feature is the position of the target word in the sentence. If a target word occurs more than once in a sentence, we consider the word's last occurrence. In contrast to AbuRa'ed and Saggion (2018), who normalize the word position by the sentence length, we use the absolute word position because we normalize all features afterwards.

4.2 Normalization

The hand-crafted features described above all range on different scales, hence, normalizing is required. The normalization is performed as follows: I.) the synsets of the target word are identified, II.) the values of features for every word in the synset are calculated, III.) the values are normalized using min-max normalization. This is being done to compare words that are related to each other, rather than comparing, for instance, frequencies of unrelated words (glee and joyous as opposed to glee and table). In this manner, we are normalizing all the values within a range of 0-1, but by comparing each word with a related word in the synset in which it is present. For words that appear in multiple synsets, we take an average of the normalized values.

As not all features could be normalized relative to their sense (see Table 1), e.g., readability features, we normalized them using scikit-learn's MinMaxScaler (Pedregosa et al., 2011).

4.3 Feature Sets

We create different feature sets considering the normalizing strategies in combination with all character and word embeddings. For the hand-crafted features, we either used the 14 `sense relative` features, all 34 `minmax` normalized features, or the 14 sense relative features combined with the missing 20 features minmax normalized (`both`). All feature sets are listed in Table 2.

4.4 Model

RS_GV's structure is a more simple version of the structure proposed in De Hertog and Tack (2018), containing linear layers instead of convolutional layers. Our model is a simple feed-forward neural network with two input layers –one for the hand-crafted features and one for the embedding features–, both followed by a linear hidden layer. Both feature layers are concatenated in another hidden linear layer. It is finally followed by a linear output layer which is activated using the rectified linear unit function (ReLU). We also use stochastic gradient descent (SGD) optimization function. L1Loss as implemented in scikit-learn (Pedregosa et al., 2011) or another mean absolute error loss function seems best for our purpose of predicting continuous labels in a regression task. Following easy stopping, we chose 250 epochs for our model. All hyperparameters with which our model performs best are listed in Appendix A (see Table 5).

RS_GV can be trained either across all domains at once (*cross-domain*) or on each domain separately (*within-domain*).

4.5 Implementation

The system is implemented in Python 3.8 and PyTorch 1.6 (Paszke et al., 2019) using the packages listed in Appendix B (see Table 6). The code of the system is available in our GitHub repository: `https://github.com/gayatrivenugopal/SharedTask-LPC2021`.

5 Results

5.1 Ablation Tests / Error Analysis

In this section, we report on different approaches made during developing RS_GV . We compare the results on the trial data using the different feature sets, and a within and a cross domain approach. In the following, we report the average of Pearson correlation on 10 system runs.

5.1.1 Feature Sets

The system's performance considering all different feature sets is summarized in Table 2.

Embed.	HCF	r	SD	Embed.	HCF	r	SD
GloVe	sense rel.	0.7654	0.0123	Flair	sense rel.	0.8002	0.0056
GloVe	minmax	0.7721	0.0114	Flair	minmax	0.8007	0.0039
GloVe	both	0.7689	0.0073	**Flair**	**both**	**0.8027**	0.0051
ELMo	sense rel.	0.7648	0.0103	PooledFlair	sense rel.	0.7331	0.0050
ELMo	minmax	0.7667	0.0119	PooledFlair	minmax	0.7685	0.0068
ELMo	both	0.7752	0.0118	PooledFlair	both	0.7537	0.0051
BERT	sense rel.	0.7204	0.0085				
BERT	minmax	0.7260	0.0088				
BERT	both	0.7178	0.0134				

Table 2: Results of all feature sets reporting Pearson correlation r (average of 10 runs) on the trial data set. The standard deviation is provided in the last column.

Hand-crafted Feature Sets. Considering all embedding feature sets (see Table 2), RS_GV performs often best and with a comparative low standard deviation (see Table 2) with the hand-crafted feature set `both` (e.g., r_{flair}=0.8027, ±0.0051) compared to `sense relative` (e.g., r_{flair}=0.8002, ±0.0056) and `minmax` (e.g., r_{flair}=0.8007, ±0.0039). Hence, in the following, we report the results only on the hand-crafted feature set `both`.

Embedding Feature Sets. A comparison between the character embeddings and the word embeddings (see Table 2) shows that `PooledFlair` (r=0.7537, ±0.0051) could outperform `BERT` (r=0.7178, ±0.0134) but `ELMo` could also outperform `PooledFlair` (r=0.7752, ±0.0118). `Flair` (r=0.8027, ±0.0051) could outperform all other embedding feature sets.

Surprisingly, RS_GV with the non-contextualized embeddings feature set (`GloVe`, r=0.7689, ±0.0073) could outperform all systems with contextualized embeddings except `Flair` (r=0.8027, ±0.0051) and `ELMo` (r=0.7752, ±0.0118). It seems that the impact of the contextualization of the embeddings is not as high as expected.

As a compromise of contextualized vs non-contextualized and character vs word embeddings, we use stacked `Flair` embeddings. They combine the forward and backward versions of `Flair` contextualized character embeddings with `GloVe` non-contextualized word embeddings.

5.1.2 Cross-domain vs. within-domain

In contrast to the insight of Yimam et al. (2018), RS_GV performs on average better using the cross-domain approach (r=0.8027, ±0.0051) than the within-domain approach (r=0.7823, ±0.0235). The standard deviation of the within-domain approach implies that the model is not as robust as the cross-domain approach. Roughly 3000 instances per domain might be too less to train a robust LCP model with a neural network.

5.1.3 Deep Learning vs. Machine Learning

We compare the results of our deep learning approach of RS_GV with a machine learning regression, i.e., linear regression of scikit-learn. As a result, the neural network and `Flair` (r=0.8027, ±0.0051) significantly improve LCP compared to the machine learning regression (r=0.6945) using only hand-crafted features. Hence, we can confirm the results of the CWI shared task 2018, character embeddings and neural networks do improve LCP.

5.2 Submitted Results

Following the previously described ablation tests, we chose to submit the results of the cross-domain approach and the within-domain approach. Both use a deep learning regressor and stacked `Flair` embeddings in combination with the hand-crafted feature set `both`.This section presents the official results of our system RS_GV on the test set at SemEval 2021 Task 1 sub-task 1 (see also Table 3).

With a Pearson correlation coefficient of r=0.7478 our system with the within-domain outperforms the cross-domain approach on the test data (r=0.7316). Officially, RS_GV ranks on place 34 of 54. The best system proposed by the team *JUST BLUE* achieved r=0.7886.

Comparing our submitted results with the results on an average of 10 runs (see Table 3), the cross-domain approach can outperform the within-domain approach on the test and trial data.

Overall, both approaches achieve better results than each of the baselines.

Setting or Team	**Version**	**trial** r	**test** r
within-domain	submission	**0.8156**	**0.7478**
cross-domain	submission	0.7978	0.7316
within-domain	average	0.7823	0.7287
cross-domain	average	**0.8027**	**0.7408**
Complexity-average	baseline	-	
Length	baseline	0.1589	
Log Frequency	baseline	0.5287	
Log Frequency & Length	baseline	**0.5376**	
JUST BLUE ()	best team	**0.8340**	**0.7886**

Table 3: Results using the trial (3rd) and test dataset (4th column) using Pearson correlation r for evaluation. The first block contains our submitted and averaged results of 10 runs using `Flair` and `both`. The second block reports the results of the baselines and the third block the results of the best performing system.

5.3 Error Analysis

The submitted results reveal that RS_GV cannot stick with the shared task's best performing systems. This section presents insights regarding the problems and strengths of RS_GV on the test data.

Domain-specific Results. The subcorpora differ regarding their lexical complexity: The biomed subcorpus has the highest average of lexical complexity in the single word dataset (0.325) and the europarl subset the lowest average (0.286). When we train and predict the lexical complexity per domain,

we can observe the same ranking of the complexity prediction per domain as in Shardlow et al. (2020): The lexical complexity of the europarl domain is most difficult to predict for RS_GV , whereas the biomedical subcorpus is most easy (see Table 4).

Domain	Feature set	r (n=10)	SD
all	Flair + both	0.7823	0.0235
bible	Flair + both	0.7177	0.0182
biomed	Flair + both	0.8585	0.0042
europarl	Flair + both	0.7444	0.0089

Table 4: Results of within-domain approach and results per domain using the trial dataset for evaluation. The Pearson correlation r is an average of 10 system runs. The standard deviation is provided in the last column.

High vs. Low Complexity. It seems that our system can predict complex words better than easy words. However, when splitting the test dataset by complexity value and not by domain, RS_GV performs poorly on very complex words (complexity value $>$ 0.666, r=0.0125, $\pm$=0.0542, n=12), which might be again due to too less training samples (n=105) for the neural network.

Furthermore, the system performs poorly for very easy words (complexity value $<$0.2, r=0.0873, $\pm$=0.0272, n=188) although roughly 20% of the training samples (n=1600) are in this complexity area. We have not found a reason for it yet.

Homonym-specific Results. This SemEval task aims at predicting word complexity of tokens in different context including different meanings. Looking more closely on homonyms, on the one hand, different complexity values are assigned to different meanings of a homonym, e.g., vision, but on the other hand, similar complexity values are assigned to a homonym, e.g., resolution. Hence, there is no clear interpretation of how to predict their complexity. This problem is reflected in RS_GV , our system predicts only slightly different complexity values per homonyms. It seems, that RS_GV can somehow differentiate the different meaning of the words but overall it differentiate not good enough to perform well in their lexical complexity prediction.

The examples also show the importance of the multi-word LCP task, hence "account" is part of light verb constructions as "to give account" and "to take into account".

Context-specific Results. A few samples contain the same token in the (nearly) same sentence, but the complexity values of them are varying (see Appendix C, Table 9). Removing these 6 out of overall 917 samples of the test data, the system output already improve from 0.7316 to 0.7334. This underlines that LCP is a subjective task and, hence, difficult to predict for machines.

Linearity. We tested the data for linearity in order to justify the usage of linear regression. We could not find any linearity between the individual features and the complexity value. The missing linearity might be a reason why RS_GV could not keep with other systems of the shared task.

6 Discussion and Conclusion

We described our model named RS_GV which was submitted to SemEval Task 2021: Task 1 regarding lexical complexity prediction. We propose a neural network with a combination of hand-crafted and word/character embeddings to approach the task. Our analysis shows that normalization of hand-crafted features using WordNet senses achieves better results than using only a minmax normalization. Furthermore, we figured out that RS_GV predicts lexical complexity best using a combination of non-contextualized word embeddings and contextualized character embeddings.

In contrast to other shared tasks results, our cross-domain approach achieves better results than the domain-specific approach. A domain-specific approach may need more data to perform reliably.

Furthermore, our neural regressor seems problematic, since it shows some variance in the results on average and the current dataset might be too small for regression with neural networks.

7 Future Work

In future works, we plan to improve the character and word embeddings. We could fine-tune the embeddings on our data or use domain-specific pre-trained embeddings, which fits the datasets' domains, e.g., BioFlair (Sharma and Jr, 2019).

Furthermore, we could calculate more hand-crafted features or edit the current ones. For example, the implementation of sentence readability formulas seems more promising than the misuse of text readability formulas on sentences.

The current neural network contains only a few linear layers, an extension using, e.g., convolutional layers for feature selection seems promising.

Acknowledgements

We thank the SemEval-2021 Task 1 organizers for preparing and organizing the shared task.

This research is part of the PhD-program "Online Participation", supported by the North Rhine-Westphalian (German) funding scheme "Forschungskolleg".

We gratefully acknowledge NVIDIA Corporation's support with the donation of the Titan Xp GPU used for this research.

References

Ahmed AbuRa'ed and Horacio Saggion. 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*, pages 159–165, New Orleans, Louisiana. Association for Computational Linguistics.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019a. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019b. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shivam Bansal and Chaitanya Aggarwal. 2014. Textstat. Last updated: 2020-11-20, Last accessed: 2021-02-22.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Brants, Thorsten and Franz, Alex. 2006. Web 1t 5-gram version 1.

Meri Coleman and Ta Lin Liau. 1975. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60(2):283.

Edgar Dale and Jeanne S Chall. 1948. A formula for predicting readability: Instructions. *Educational research bulletin*, pages 37–54.

Elnaz Davoodi and Leila Kosseim. 2016. CLaC at SemEval-2016 task 11: Exploring linguistic and psycho-linguistic features for complex word identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 982–985, San Diego, California. Association for Computational Linguistics.

Dirk De Hertog and Anaïs Tack. 2018. Deep learning architecture for complex word identification. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 328–334, New Orleans, Louisiana. Association for Computational Linguistics.

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

Oxford Learner's Dictionaries. 2021. Oxford 3000 and 5000. Last accessed: 2021-02-22.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Sian Gooding and Ekaterina Kochmar. 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*, pages 184–194, New Orleans, Louisiana. Association for Computational Linguistics.

Sian Gooding and Ekaterina Kochmar. 2019. Recursive context-aware lexical simplification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4853–4863, Hong Kong, China. Association for Computational Linguistics.

Robert Gunning. 1952. *Technique of clear writing*. McGraw-Hill, New York, USA.

Nathan Hartmann and Leandro Borges dos Santos. 2018. NILC at CWI 2018: Exploring feature engineering and feature learning. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 335–340, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Colby Horn, Cathryn Manduca, and David Kauchak. 2014. Learning a lexical simplifier using Wikipedia. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 458–463, Baltimore, Maryland. Association for Computational Linguistics.

David Kauchak. 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)*, pages 1537–1546, Sofia, Bulgaria. Association for Computational Linguistics.

J. Peter Kincaid, Robert P. Fishburne Jr., Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch.

Gustavo Paetzold and Lucia Specia. 2016. SemEval-2016 task 11: Complex Word Identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 560–569, San Diego, California. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Claudia H Sánchez-Gutiérrez, Hugo Mailhot, S Hélène Deacon, and Maximiliano A Wilson. 2018. Morpholex: A derivational morphological database for 70,000 english words. *Behavior research methods*, 50(4):1568–1580.

Toby Segaran and Jeff Hammerbacher. 2009. *Beautiful data: The stories behind elegant data solutions*, 1 edition. O'Reilly Media, Inc.

RJ Senter and Edgar A Smith. 1967. Automated readability index. Technical report, Cincinnati University, Cincinnati, USA.

Matthew Shardlow. 2013. 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*, pages 69–77, Sofia, Bulgaria. Association for Computational Linguistics.

Matthew Shardlow. 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)*, pages 1583–1590, Reykjavik, Iceland. European Language Resources Association (ELRA).

Matthew Shardlow, Michael Cooper, and Marcos Zampieri. 2020. CompLex: A new corpus for lexical complexity predicition from Likert Scale data. In *Proceedings of the 1st Workshop on Tools and Resources to Empower People with REAding DIfficulties (READI)*, pages 57–62, Marseille, France. European Language Resources Association.

Matthew Shardlow, Richard Evans, Gustavo Paetzold, and Marcos Zampieri. 2021a. SemEval-2021 Task 1: Lexical Complexity Prediction. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2021)*.

Matthew Shardlow, Richard Evans, and Marcos Zampieri. 2021b. Predicting lexical complexity in english texts. *arXiv preprint arXiv:2102.08773*.

Shreyas Sharma and Ron Daniel Jr. 2019. BioFLAIR: Pretrained pooled contextualized embeddings for biomedical sequence labeling tasks. *arXiv preprint arXiv:1908.05760*.

Serge Sharoff. 2006. Open-source corpora: Using the net to fish for linguistic data. *International Journal of Corpus Linguistics*, 11(4):435–462.

Rachael Tatman. 2017. English word frequency: 1/3 million most frequent english words on the web. Last updated: 2018-09-28; Last accessed: 2021-02-22.

Nikhil Wani, Sandeep Mathias, Jayashree Aanand Gajjam, and Pushpak Bhattacharyya. 2018. The whole is greater than the sum of its parts: Towards the effectiveness of voting ensemble classifiers for complex word identification. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 200–205, New Orleans, Louisiana. Association for Computational Linguistics.

Michael Wilson. 1988. Mrc psycholinguistic database: Machine-usable dictionary, version 2.00. *Behavior research methods, instruments, & computers*, 20(1):6–10.

Seid Muhie Yimam, Chris Biemann, Shervin Malmasi, Gustavo Paetzold, Lucia Specia, Sanja Štajner, Anaïs Tack, and Marcos Zampieri. 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*, pages 66–78, New Orleans, Louisiana. Association for Computational Linguistics.

Seid Muhie Yimam, Sanja Štajner, Martin Riedl, and Chris Biemann. 2017. CWIG3G2 - complex word identification task across three text genres and two user groups. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 401–407, Taipei, Taiwan. Asian Federation of Natural Language Processing.

A Hyperparameter

Hyperparameter	Final Value	Fine-tuning Values
classifier	Neural Lin. Regession	sklearn LinearSVR, sklearn LinearRegression, Neural Lin. Regression
learning rate	0.075	0.01, 0.05, 0.075, 0.1, 0.2
epochs	250	100, 250, 500
# input layer	2	1, 2
# hidden layer	3	1, 2, 3
hidden size (HCF)	128	128, 256, 512
hidden size (EM)	256	256, 512, 1024
hidden size (concat)	128	128, 256, 512
criterion	L1Loss	L1Loss, MSELoss, SmoothL1Loss
optimizer	SGD	SGD, ADAM
dropout	-	0.1, 0.05, 0.01

Table 5: Hyperparameter during fine tuning and the final chosen hyperparameter of the proposed system.

B Resources

B.1 Python Packages

Package	Usage	Package	Usage
pandas	Data Import	torch	Regression
xlrd	Data Import	scikit-learn	Regression
spacy	Preprocessing	interpret	Regression
stanza	Preprocessing	numpy	Ablation Study &
nltk	WordNet Feature		Error Analysis
syllables	Syllable Feature	seaborn	Data Visualization
textstat	Readability Feature	yellowbrick	Data Visualization
Flair	Embedding Feature	visdom	Data Visualization
torch	Model		
scikit-learn	Evaluation		

Table 6: Python packages used for the implementation of the proposed system.

B.2 Language Resources

name	usage	type	access	reference
CompLex	data	corpus	https://github.com/MMU-TDMLab/CompLex	Shardlow et al. (2020, 2021b)
Textstat	readability features	python package	https://pypi.org/project/textstat/	Bansal and Aggarwal (2014)
internet-en-forms	frequency feature	word list	http://corpus.leeds.ac.uk/list.html#frqc	Sharoff (2006)
GoogleWeb1T, unigram_freq.csv	frequency feature	word list	https://www.kaggle.com/rtatman/english-word-frequency	Tatman (2017)
GoogleWeb1T, count_1w.txt	frequency feature	word list	https://norvig.com/ngrams/	Segaran and Hammerbacher (2009)
NLTK	lexical feature	NLP library	https://www.nltk.org/	Bird et al. (2009)
spaCy	lexical feature	NLP library	https://github.com/explosion/spaCy	Honnibal and Montani (2017)
stanza	lexical feature	NLP library	https://stanfordnlp.github.io/stanza/	Qi et al. (2020)
syllapy	lexical feature	python package	https://github.com/mholtzscher/syllapy	
syllable	lexical feature	python package	https://github.com/prosegrinder/python-syllables	
WordNet	WordNet feature	database	https://www.nltk.org/api/nltk.corpus.reader.html?highlight=wordnet#module-nltk.corpus.reader.wordnet	Fellbaum (1998)
Oxford 3000	lexicon feature	word list	https://github.com/gokhanyavas/Oxford-3000-Word-List/blob/master/Oxford%203000%20Word%20List%20No%20Spaces.txt	Dictionaries (2021)
MorphoLex-EN	morphological feature	database	https://github.com/hugomailhot/MorphoLex-en	Sánchez-Gutiérrez et al. (2018)
MRC Psycholing. Database	psycholinguistic feature	database	https://github.com/samzhang111/mrc-psycholinguistics/raw/master/mrc2.dct	Wilson (1988)
FLAIR	embedding feature	NLP framework	https://github.com/flairNLP/flair	Akbik et al. (2019a)
GloVe	word embedding feature	pretrained embeddings	http://nlp.stanford.edu/data/glove.6B.zip	Pennington et al. (2014)
FLAIR	character embedding feature	pretrained embeddings	https://github.com/flairNLP/flair	Akbik et al. (2018, 2019a)
PooledFlair	character embedding feature	pretrained embeddings	https://github.com/flairNLP/flair	Akbik et al. (2019b)
BERT	word embedding feature	pretrained embeddings	https://github.com/google-research/bert	Devlin et al. (2019)
ELMo	word embedding feature	pretrained embeddings	https://allennlp.org/elmo	Peters et al. (2018)

Table 7: All used language resources listed with usage, access and reference.

B.3 Word and Character Embeddings

embedding name	type	context	specification	domain	corpora	dimensions
Flair	character	x	Mix-forward, mix-backward, glove	web, wikipedia, subtitles	1 Billion Word Benchmark	4196
PooledFlair	character	x	Mix-forward	web, wikipedia, subtitles	-	4096
BERT	word	x	bert-base-uncased	Fiction, news, wikipedia	BooksCorpus, Wikipedia, 1 Billion Word Benchmark	3072
ELMO	word	x	original	news	1 Billion Word Benchmark	3072
GloVe	word		glove.6B.300d	wikipedia, news	Wikipedia 2014, Gigaword 5	300

Table 8: Settings of the word and character embeddings.

C Detailed Results

C.1 Context-specific Results.

ID	Sentence	Token	Complexity	Predicted
39HYCOOPKOL434K1UCPA8CBZRO4DMM	Arguably, since the body pools and plasma sitosterol levels in the knockout mice are so considerably elevated, perhaps the biliary sitosterol levels could be considered to be inappropriately low.	body	0.0499	0.2049
3ZZAYRN1I6RZKW1ATI425KIQPA7TO0	Arguably, since the body pools and plasma sitosterol levels in the knockout mice are so considerably elevated, perhaps the biliary sitosterol levels could be considered to be inappropriately low.	body	0.3173	0.2049
3KWGG5KP6J2UYCENUGUZO6TH6QDCMA	Fishing opportunities and financial contribution provided for by the EU-Seychelles Fisheries Partnership Agreement (	Fisheries	0.3088	0.3387
341H3G5YF0EA3RIQXPR917NPC4EZ0Z	Fishing opportunities and financial contribution provided for in the EU-São Tomé and Príncipe Fisheries Partnership Agreement (	Fisheries	0.1875	0.3405
3LCXHSGDLT6CT5B6A4WGQ3SQJNDSES	Therefore thus says Yahweh of Armies concerning the prophets: Behold, I will feed them with wormwood, and make them drink the water of gall; for from the prophets of Jerusalem is ungodliness gone forth into all the land.	wormwood	0.7321	0.4117
3DWNFENNE3V120VNY4BPPGPCAHX4JD	therefore thus says Yahweh of Armies, the God of Israel, Behold, I will feed them, even this people, with wormwood, and give them water of gall to drink.	wormwood	0.4843	0.4170

Table 9: Samples of the test set with the same token in the same sentence but different complexity values. The last column contains the predicted values of RS_GV .