

# NLPSangh at SemEval 2021: Task 1 - Lexical Complexity Prediction

## Team Name : iitkgp.cs60075.team4

**Satyam Porwal**  
17CS10048

**Prakhar Bindal**  
17CS10036

**Akash Tiwari**  
17CS10003

**Pravesh Jain**  
17CS10040

**Prabhpreet Singh**  
17CS10035

### Abstract

This paper presents the systems we submitted to SemEval 2021, Shared Task-1, Lexical Complexity Prediction (LCP). We will describe our best performing systems' implementation and discuss our key findings from this research. We will also present other incremental works we did to reach this model. Our best performing model had an Pearsonr score of 0.7081 and MSE of 0.0082 in Subtask-1 and 0.7393 and 0.0110 on Subtask-2 respectively.

## 1 Individual Contributions

The research has been efforts of the mentioned members of the groups. The group tried two different approaches for the first task and divided itself into groups of two:

### Group 1. Handcrafted + Pre-trained Sentence Transformers (Model Submitted):

**Satyam Porwal and Prakhar Bindal** : The group performed literature survey on various approaches that had been taken for similar tasks. The findings pointed towards using a sentence transformer and using handcrafted features.

**Prakhar Bindal** performed extensive literature survey on finding handcrafted features which have shown high information gain for similar tasks and implemented multiple features which had good performance based on our use-case.

**Satyam Porwal** based on the findings of his literature survey of sentence transformers, wrote the python-based code for training the data and performed various experiments to improve the performance.

### Group 2. CharLSTM + BERT Transformer

**Pravesh Jain and Akash Tiwari** : The group

performed literature survey for CharLSTM and different ways BERT has been used for similar tasks. The major drawback was too much time to train and often caused time-out while using Google Colab. Results obtained were not as good as the first approach.

**Pravesh Jain** performed literature survey on charLSTM and it's usage in generative models. Wrote the code and performed training in sync with Akash.

**Akash Tiwari** did a similar literature survey and collaborated to train the model. The model had slow training and the model was not performing well given the technical difficulties faced.

Both Akash and Pravesh then collaborated with the first group and performed various experiments to get better results using the first model. Some of the ideas from their approach was then integrated in the model.

For the second sub-task they literature survey was done by the four of us. Since the work in this was limited, we only found a few features which had not been previously used, hence we decided to use the previous approach and extend it. An equal collaboration among the five was done for training and coding in this sub-task.

## 2 Introduction

With the rapid growth in digital pedagogy, English has become an extremely popular language. Although English is considered an easy language to learn and grasp, a person's choice of words often affects texts' readability. The use of difficult words can potentially lead to a communication gap, thus hampering language efficiency. Keeping these issues in mind, many Natural Language Processing tasks for text simplification have been recently proposed.(1; 2) Our task of lexical complexity prediction is an important step in the process of simpli-

fying texts. The SemEval 2021 Task 1 (3) focuses on lexical complexity prediction in English. Given a sentence and a token from it, we have to predict the complexity score of the token. The task has two Sub-Tasks:

Sub-Task 1: complexity prediction of single words

Sub-Task 2: complexity prediction of multi word expressions (MWEs). A word might seem complex because of 2 major factors

a) The word is less common or complex in itself.

b) The context in which the word is used makes it hard to comprehend. We tried to replicate both these factors in our implementation, which is available on Github (4). The various datasets used have been made available on Google Drive (5).

### 3 Related Works

#### 3.1 Complex Word Identification

Given the interest of the community in Complex Word Identification(CWI) two shared tasks have been organized so far on this topic.

The first edition of CWI shared task was SemEval-2016 Task 11 (6). In this task the complexity was defined as whether or not a word is difficult to understand for non-native English speakers. In the CWI 2016 dataset, the annotation followed the binary approach wherein the English words in context were tagged as complex or noncomplex. The organizers labeled a word as complex in the dataset if the word has been assigned by at least one of the annotators as complex. All words that have not been assigned by at least one annotator as complex have been labeled as non-complex. The task was to use this dataset to train classifiers to predict lexical complexity assigning a label 0 to non-complex words and 1 to complex ones.

In Zampieri et al. (7) oracle and ensemble methods have been used to investigate the performance of the participating systems. The study showed that most systems performed poorly due to the way the data was annotated and also due to the fact that lexical complexity was modelled as a binary task, a shortcoming addressed by CompLex (3).

Finally, a second iteration of the CWI shared task was organized at the BEA workshop 2018. In CWI 2018, a multilingual dataset was made available containing English, German, and Spanish training and testing data for monolingual tracks, and a French test set for multilingual predictions. It featured two sub-tasks: a binary classification task, similar to the CWI 2016 setup, where participants

were asked to label the target words in context as complex (1) or simple (0); and a probabilistic classification task where participants were asked to assign the probability that an annotator would find a word complex.

#### 3.2 Text Simplification

Text simplification evaluation is an active area of research, with recent efforts focussing on evaluating the whole process of text simplification in the style of machine translation evaluation. Whilst BLEU score (8) has been used for text simplification evaluation, this is not necessarily an informative measure, as it only measures similarity to the target. It does not help a researcher to understand whether the resultant text preserves meaning, or is grammatical.

To overcome some of these shortcomings, Xu et al. (9) introduced the SARI method of evaluating text simplification systems. SARI comprises parallel simplified-unsimplified sentences and measures additions, deletions and those words that are kept by a system. However, SARI is still an automated measure and optimising systems to get a good SARI score may lead to systems that do well on the metric, but not in human evaluations. Recently, EASSE (10) has been released to attempt to standardise simplification evaluation by providing a common reference implementation of several text simplification benchmarks

### 4 Electronically-available resources

We used the following resources for various handcrafted features:

- [Wikipedia Basic English Word List](#)
- [Ogden Word Set](#)
- [SubtlexUS](#)
- [SubIMDB](#)
- [Datamuse API](#)
- [MWE-CWI Dataset](#)
- [CWI 2018 Winner Github Code](#)

### 5 Our Approach

#### 5.1 Handcrafted Features with Pre-trained Sentence BERT

In this approach we used a combination of handcrafted features with pre trained sentence BERT for our task.

### 5.1.1 Handcrafted Features

Upon analysing the systems and datasets used in previous related works we noticed several intuitive explanations as to why a word may be judged as complex, or not:

- Length: Length of the word
- Syllables: Syllable count of the word
- SUBTLEX frequency: SubtlexUS is database containing word frequencies based on English-US movies and TV series subtitles.
- Uppercase Count: Number of uppercase letters in the word
- Stemmed Length: Length of suffix from the stemmed word
- Degree of Polysemy: A word is said to be polysemous if a word has multiple meanings. This feature denotes the number of meanings a word has which was implemented using wordnet and synsets.
- Degree of Hyponymy: Hyponym is a term used to designate a particular member of a broader class. This feature denotes the number of Hyponyms a word has which was implemented using wordnet and synsets.
- Degree of Hypernymy: Hypernymy is the semantic relation of being superordinate or belonging to a higher rank or class. This feature denotes the number of Hypernyms a word has which was implemented using wordnet and synsets.
- Presence in Ogden word set: This is a list of the 850 words in the Basic English core vocabulary. These words all denote simple concepts commonly used in everyday life. This feature is a boolean flag indicating whether the word is present in the word set or not.
- Presence in Simple wiki word set: This is the maximum Basic English combined wordlist. This feature is a boolean flag indicating whether the word is present in the word set or not.
- Presence in SubIMDB word set: SubIMDB is a structured corpus of subtitles that captures everyday language. This feature is a boolean

flag indicating whether the word is present in the word set or not

- Singularity/Plurality of the word: This feature is a list of three boolean features:
  1. Whether the word is in singular or not
  2. Whether the word is in plural or not
  3. Whether the word is in both singular and plural form

In case of multi-word expressions, we used the features described above in the following manner. For all the numerical features we took average of the vectors obtained for both words in the multi-word expression and for binary features we only used the vector obtained from the second word.

### 5.1.2 Word Embedding

Word embedding have been used in various previous works in field of text analysis. For our purpose we used these such that the neural networks learns the target word and in case of multi-words a concatenation of the two embedding was performed. For these word embedding we tried using GloVe (11) and FastText (12; 13). We were not able to use the original FastText implementation (14), due to RAM restrictions. Moreover the results showed GloVe embeddings performed better than the FastText embeddings.

### 5.1.3 Sentence-BERT

Sentence Transformers is a Python framework for state-of-the-art sentence, text and image embeddings. Sentence-BERT (SBERT) (15) is a modification of the pretrained BERT network that use Siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity. The results obtained in the original work showed similar accuracy to one obtained using BERT but with reduced efforts. We used 'stsb-distilbert-base' for our experiments. Various performance comparison (16) show it has a high STS benchmark score alongside a high processing speed. We also tried to use 'stsb-roberta-large', but there was not much performance improvements compared to 'stsb-distilbert'.

### 5.1.4 Neural Network

The Neural Network used had 3 fully connected layer followed by a Sigmoid activation function. We also tried NN using ReLu activation and also without any activation function. The results showed a drop in performance hence were discarded.

## 5.2 CharLSTM + Pre-trained BERT

In this Approach, we fine tuned the existing pre-trained BERT model with our data-set.

### 5.2.1 Pre-processing

Stemming of the tokens were done along with converting sentences into one hot encoding with padding before passing them into the model.

### 5.2.2 Bert and LSTM

We took the pre-trained embeddings from the "bert-base-uncased" model. We took the existing LSTM neural network and then modified it to fine tune the embeddings by running it on our data-set.

### 5.2.3 Neural Network

This was followed by trying various Neural network combinations along with the suggested models. We used a combination of dropout layers before each linear layer to prevent over-fitting and ReLu as the activation function. The factors tweaked were- learning rate, number of neurons per layer, dropout probability, number of layers.

### 5.2.4 Takeaway

After multiple tweaking, the loss was not meeting our expectations and stagnated. This prompted us to look for other approaches such as increasing the features for better results. This was then done using the handcrafted features in the approach discussed earlier, which showed better results.

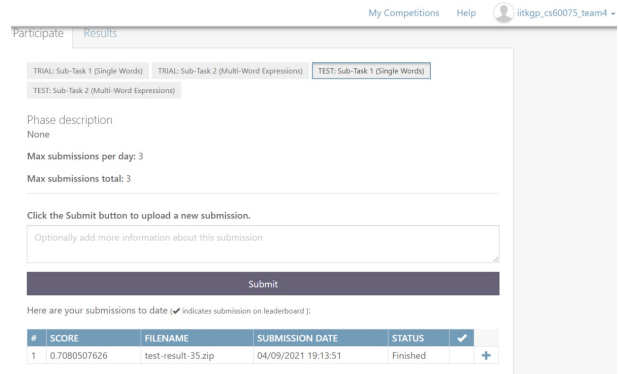
## 6 Results

The final submission have been made to [SemEval Task-1 Codalab](#) under the team name "iitkgp\_cs60075\_team4". The results are shown in Table 1.

Subtask-Phase	Pearson	MSE
Subtask 1 - Trial	0.7505	0.0083
Subtask 1 - Test	0.7081	0.0082
Subtask 2 - Trial	0.7177	0.0120
Subtask 2 - Test	0.7393	0.0110

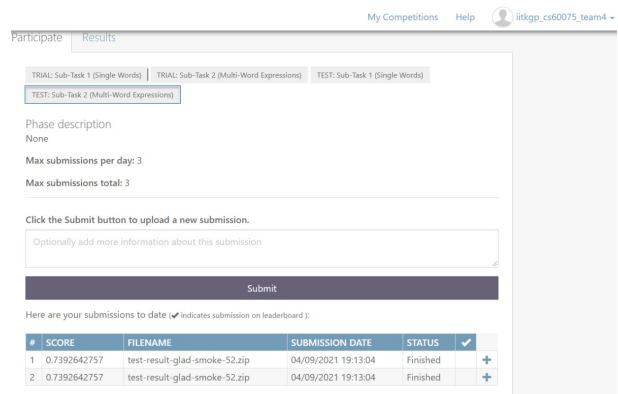
Table 1: Results obtained in various phases

The screenshots of the results obtained have been provided as verification. Figure 1 and Figure 2 shows the scores obtained in the Test Phase of Subtask-1. and Subtask-2 respectively. The photos are also available on the github page (4).



#	SCORE	FILENAME	SUBMISSION DATE	STATUS	✓
1	0.7080507626	test-result-35.zip	04/09/2021 19:13:51	Finished	+

Figure 1: Screenshot of codalab submission for Test Phase Subtask 1



#	SCORE	FILENAME	SUBMISSION DATE	STATUS	✓
1	0.7392642757	test-result-glad-smoke-52.zip	04/09/2021 19:13:04	Finished	+
2	0.7392642757	test-result-glad-smoke-52.zip	04/09/2021 19:13:04	Finished	+

Figure 2: Screenshot of codalab submission for Test Phase Subtask 2

## 7 Conclusion

It was already shown that previous datasets were insufficient for the task of Complex Word Identification. In fact, the very definition of the task — identifying complex words in a binary setting is at fault. In this task we were presented with CompLex, a new dataset for lexical complexity prediction which proposed a new 5-point Likert scale annotation scheme to annotate complex words in texts from three sources: the Bible, Europarl, and biomedical texts. In this paper we presented a system for lexical complexity prediction in the form of a prediction task. We came up with two approaches for the task one with Handcrafted features and Pre-trained sentence BERT and other one with CharLSTM and Pre-trained BERT. The second approach didn't meet our loss expectations and hence the first approach was primarily adopted for the task.

## References

- [1] Gustavo H. Paetzold and Lucia Specia, “A survey on lexical simplification,” *J. Artif. Int. Res.*, vol. 60, no. 1, pp. 549–593, Sept. 2017.
- [2] Punardeep Sikka and Vijay Mago, “A survey on text simplification,” 2020.
- [3] Matthew Shardlow, Michael Cooper, and Marcos Zampieri, “Complex: A new corpus for lexical complexity prediction from likert scale data,” 2020.
- [4] “Nlp term project spring 2021 team 4,” <https://github.com/prakharmath/CS60075-Team-4-Task-1>.
- [5] “Data folder containing all the data set and cached files used,” <https://drive.google.com/drive/folders/1VcfuOHGu9OMn0qpjFw70fEmJ4pAjSfIa?usp=sharing>.
- [6] “Complex word identification (cwi) shared task 2018,” <https://sites.google.com/view/cwisharedtask2018/>.
- [7] Marcos Zampieri, Shervin Malmasi, Gustavo Paetzold, and Lucia Specia, “Complex word identification: Challenges in data annotation and system performance,” in *Proceedings of the 4th Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2017)*, Taipei, Taiwan, Dec. 2017, pp. 59–63, Asian Federation of Natural Language Processing.
- [8] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, “Bleu: a method for automatic evaluation of machine translation,” in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, Pennsylvania, USA, July 2002, pp. 311–318, Association for Computational Linguistics.
- [9] Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch, “Optimizing statistical machine translation for text simplification,” *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 401–415, 2016.
- [10] Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia, “EASSE: Easier automatic sentence simplification evaluation,” 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): System Demonstrations*, Hong Kong, China, Nov. 2019, pp. 49–54, Association for Computational Linguistics.
- [11] Jeffrey Pennington, Richard Socher, and Christopher Manning, “GloVe: Global vectors for word representation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1532–1543, Association for Computational Linguistics.
- [12] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov, “Enriching word vectors with subword information,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.
- [13] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, H erve J egou, and Tomas Mikolov, “Fasttext.zip: Compressing text classification models,” *arXiv preprint arXiv:1612.03651*, 2016.
- [14] “Fasttext english word vectors,” <https://fasttext.cc/docs/en/english-vectors.html>.
- [15] Nils Reimers and Iryna Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” 2019.
- [16] “Performance comparison of various models,” <https://www.sbert.net/#performance>.