CS-UM6P at SemEval-2021 Task 1: A Deep Learning Model-based Pre-trained Transformer Encoder for Lexical Complexity

Nabil El Mamoun[1] **Abdelkader El Mahdaouy**[2] **Abdellah El Mekki**[2]
Kabil Essefar[2] **Ismail Berrada**[2]
[1]Faculty of Sciences Dhar EL Mahraz, Sidi Mohamed Ben Abdellah University, Morocco
[2]School of Computer Sciences, Mohammed VI Polytechnic University, Morocco
{firstname.lastname}@um6p.ma

Abstract

Lexical Complexity Prediction (LCP) involves assigning a difficulty score to a particular word or expression, in a text intended for a target audience. In this paper, we introduce a new deep learning-based system for this challenging task. The proposed system consists of a deep learning model, based on a pre-trained transformer encoder, for word and Multi-Word Expression (MWE) complexity prediction. First, on top of the encoder's contextualized word embedding, our model employs an attention layer on the input context and the complex word or MWE. Then, the attention output is concatenated with the pooled output of the encoder and passed to a regression module. We investigate both single-task and joint training on both Sub-Tasks data using multiple pre-trained transformer-based encoders. The obtained results are very promising and show the effectiveness of fine-tuning pre-trained transformers for LCP tasks.

1 Introduction

Text Simplification (TS) is a fundamental task for improving text readability, and presents a wide variety of use cases, including assisting children with reading difficulties and native speakers with low literacy levels (De Belder and Moens, 2010; Aluísio and Gasperin, 2010), increasing accessibility for people with intellectual disabilities (Saggion, 2017), and facilitating certain aspects of language for language learners (Paetzold and Specia, 2016). TS may involve modifications to the syntactic structure of a sentence, its lexical units or both (Shardlow, 2014).

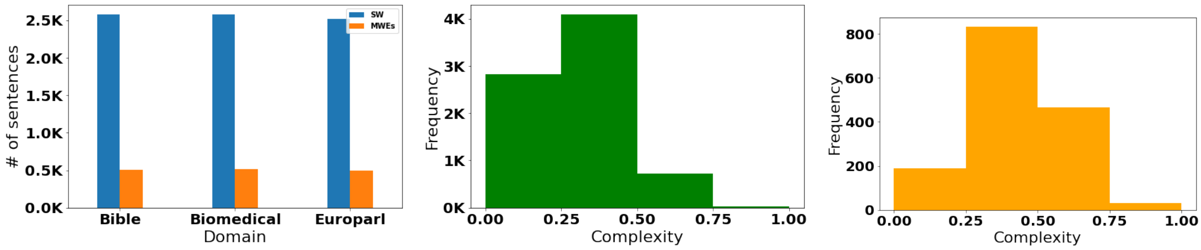
Lexical Simplification (LS), as a sub-task of TS, focuses on simplifying complex words of an input sentence. It first identifies complex words in a sentence, known as Complex Words Identification (CWI) or Lexical Complexity Prediction (LCP) task. Then, it replaces them with other alternatives of equivalent meaning. Those substitutions should be more simplistic while preserving the semantic and the grammatical structure of the input sentence (Paetzold and Specia, 2017; Qiang et al., 2020).

Most of the previous research has modeled LCP as a binary classification task (Paetzold and Specia, 2017; Zampieri et al., 2016; Ronzano et al., 2016). A recent research study has introduced a multi-domain dataset, annotated using a 5-point Likert scale scheme (Shardlow et al., 2020). The aim is to label the complexity of a word or a Multi-Word Expression (MWE), in a more fine-grained manner, from very easy to very difficult. Hence, the lexical complexity of words is expressed on a continuous scale.

In this paper, we introduce our submitted system to the SemEval-2021 LCP 1 and 2 Sub-Tasks (Shardlow et al., 2021). The proposed system consists of a deep learning model for word and MWE complexity prediction. Our model employs a residual attention block and a regression module on top of a pre-trained transformer encoder, as follows:

- The encoder is fed with the concatenation of the context and the complex word or MWE, using the SEP token of the encoder's tokenizer.
- The residual attention block is a layer on top of the encoder's Contextualized Word Embedding (CWE) of the input context (sentence) and the complex word or MWE. The aim is to leverage the encoder's CWE to extract the relevant features of the inputs.
- The attention layer output is concatenated with the pooled output of the encoder and passed to the regression module for complexity prediction.

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 585–589
Bangkok, Thailand (online), August 5–6, 2021. ©2021 Association for Computational Linguistics



(a) Distribution of sentences per domain

(b) Distribution of single words complexity

(c) Distribution of MWEs complexity

Figure 1: Domains and complexity distributions of the datasets sentences

The proposed model is trained to minimize both the Root Mean Square Error (RMSE) and the auxiliary loss associated to the negative Pearson Correlation. The two losses are combined using the uncertainty loss weighting (Kendall et al., 2017). We investigate two pre-trained transformer networks, namely BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Moreover, we evaluate both single-task and joint training of word and MWE complexity prediction sub-tasks. The best performances are achieved using RoBERTa-large encoder while performing joint training on both Sub-Tasks data. The obtained results are very promising and show the effectiveness of our system, which was ranked among the top 10 submitted systems to both LCP 1 and 2 Sub-Tasks.

The rest of this paper is organized as follows. Section 2 describes the dataset and the sub-tasks of SemEval-2021 Task 1. In Section 3, we present our system overview. Section 4 summarizes and discusses the obtained results for both Sub-Task 1 and Sub-Task 2. Finally, Section 5 concludes the paper.

2 Task Description

2.1 Dataset Description

The dataset of the Lexical Complexity Prediction shared task (Shardlow et al., 2021) is an augmented version of the Complex dataset (Shardlow et al., 2020). In addition to complex word annotation, the data also include MWEs along with their context sentences and complexity scores. The dataset is annotated using a 5-point (1-5) Likert scale scheme and covers sentences from three domains: Bible, EuroParl, and Biomedical texts. The dataset is labeled by a group of annotators from English-speaking countries. It is compiled from sentences with at least four valid annotations. The aggregation of annotations is performed ensuring that the normalized complexity is in the interval $[0, 1]$. The complexity scores are on a 5-point Likert scale and correspond to five levels of complexity ranging from ”Very Easy” to ”Very difficult” (Shardlow et al., 2020).

2.2 Sub-tasks Description

The LCP shared task consists of two sub-tasks (Shardlow et al., 2021):

- **Sub-Task 1:** predicting the complexity score of single words.
- **Sub-Task 2:** predicting the complexity score of multi-word expressions.

The training set consists of 7,662 samples for single word complexity prediction (Sub-Task 1), while the training set of MWE sub-task contains 1,517 samples (Sub-Task 2). Figure 1a presents the number of samples per domain. The dataset is almost balanced for all three domains in the two LCP sub-tasks. Figure 1b and 1c show the complexity distribution of single words and MWEs, respectively. The Figures (1b and 1c) illustrate that most single words have a complexity score less than 0.5, whereas for MWEs, the complexity scores are between 0.25 and 0.75.

3 System Overview

The proposed system uses a residual attention block and a regression module on top of the pre-trained transformer encoder network. In the following, we describe each component of our system.

3.1 Transformer Encoder

In order to encode the input context and the complex word or MWE, we employ two state-of-the-art pre-trained transformer encoder networks, namely BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). First, the context (sentence) and the complex word or MWE are concatenated using the special token (SEP or /s) of the encoder's tokenizer, as follows:

- BERT case:

  Input = $[CLS]$ context $[SEP]$ word/MWE

- RoBERTa case:

  Input = $< s >$ context $< /s >$ word/MWE

Then, the tokenizer of the encoder splits the input into wordpieces $[T_1, T_2, ..., T_n]$ and encodes them using its vocabulary. The transformer encoder is fed with these encoded inputs. As a result, it outputs:

- The pooled embedding $h_{pooled} \in \mathbb{R}^{1 \times d}$ (the embedding of $[CLS]$ and $< s >$ tokens for BERT and RoBERTa encoders, respectively).
- The contextualized word embedding (CWE) $H = [h_1, h_2, ..., h_n] \in \mathbb{R}^{n \times d}$ (d is the embedding dimension).

3.2 Attention block

Our model applies an attention layer on top of the CWE, output by the encoder (Bahdanau et al., 2015; Yang et al., 2016). The aim is to reward CWEs according to their relevance to the complexity prediction task. Using the CWE, the attention layer extracts a features vector v, representing the weighted sum of H vectors:

$$C = tanh(HW_a)$$

$$\alpha = softmax(C^T W_\alpha)$$

$$v = \alpha \cdot H^T$$

where $W_a \in \mathbb{R}^{d \times 1}$ and $W_\alpha \in \mathbb{R}^{n \times n}$ are the learnable parameters of the attention layer, $C \in \mathbb{R}^{n \times 1}$ is the context vector of the attention mechanism, and $\alpha \in [0, 1]^n$ weights the CWEs according to their relevance to the task.

3.3 Regression Module

The regression module F consists of one hidden layer and one output layer. F is fed with the concatenation of the encoder's pooled output h_{pooled} and the output attention block v. F outputs the $\hat{y}$, the predicted complexity:

$$\hat{y} = F([h_{pooled}, v])$$

The proposed system is trained to minimize both the Root Mean Square Error (RMSE) and the auxiliary loss associated to the negative Pearson Correlation:

- The RMSE loss:

$$\mathcal{L}_{rmse}(\hat{y}, y) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_j)^2}$$

- The auxiliary loss associated to the negative Pearson Correlation:

$$\mathcal{L}_{aux}(\hat{y}, y) = 1 - \frac{\sum_{i=1}^{N} (y_i - \overline{y})(\hat{y}_i - \overline{\hat{y}})}{\sqrt{\sum_{i=1}^{N} (y_i - \overline{y})^2} \sqrt{\sum_{i=1}^{N} (\hat{y}_i - \overline{\hat{y}})^2}}$$

where N is the number of samples, y is the ground truth complexity, $\hat{y}$ is the predicted complexity, and $\overline{y}$ (resp. $\overline{\hat{y}}$) is the mean of y (resp. $\hat{y}$). In order to combine both $\mathcal{L}_{rmse}$ and $\mathcal{L}_{aux}$, we use the uncertainty loss weighting (Kendall et al., 2017). The latter aims to combine multiple losses according to their uncertainty and to avoid manual tuning of the loss weights. Finally, our model is trained to minimize the total loss, given by:

$$\mathcal{L}_{total} = \frac{1}{2\sigma_1^2} \mathcal{L}_{rmse} + \frac{1}{2\sigma_2^2} \mathcal{L}_{aux} + \log(\sigma_1 \sigma_2)$$

where σ_1 and σ_2 are two parameters for learning the relative weight of $\mathcal{L}_{rmse}$ and $\mathcal{L}_{aux}$.

4 Results

This section describes the experiment settings and the obtained results.

4.1 Experiment Setting

We investigate the performance of our system using both the *base* and the *large* models of BERT and RoBERTa encoders:

- **BERT-base**: 12 transformer blocks, $d = 768$, 12 attention heads, and 110M parameters.
- **BERT-large**: 24 transformer blocks, $d = 1024$, 16 attention heads, and 336M parameters.
- **RoBERTa-base**: 12 transformer blocks, $d = 768$, 12 attention heads, and 125M parameters.
- **RoBERTa-large**: 24 transformer blocks, $d = 1024$, 16 attention heads, and 355M parameters.

We implement a simple text preprocessing pipeline that normalizes the contractions[1]. All models are trained using Adam optimizer (Kingma and

[1] We have employed the package contractions for this purpose. https://github.com/kootenpv/contractions

		Sub-Task 1 (Word complexity)					Sub-Task 2 (MWE complexity)				
Training	**Encoder**	**Pearson**	**Spearman**	**MAE**	**MSE**	**R2**	**Pearson**	**Spearman**	**MAE**	**MSE**	**R2**
Single Task	**BERT-base**	0.7211	0.7005	0.0742	0.009	0.4455	0.8199	0.8157	0.0732	0.0086	0.6703
	BERT-large	0.73	0.7023	0.0816	0.0105	0.3567	0.8214	0.8158	0.0723	0.0087	0.6668
	RoBERTa-base	0.7402	0.7198	0.0871	0.012	0.2654	0.8274	0.8235	0.0752	0.0087	0.6669
	RoBERTa-large	0.7613	0.7309	**0.0728**	**0.0088**	**0.4629**	0.8369	0.8349	0.0749	0.0088	0.6619
Joint Training	**BERT-base**	0.7236	0.7058	0.0827	0.0109	0.3288	0.8256	0.8125	0.0738	0.0088	0.6349
	BERT-large	0.7317	0.6936	0.077	0.0097	0.406	0.8371	0.8391	0.0703	0.0083	**0.7191**
	RoBERTa-base‡	0.7576	0.7318	0.0754	0.0091	0.4374	0.8424	0.8322	**0.0696**	**0.0078**	0.6767
	RoBERTa-large‡	**0.7779**	**0.7366**	0.0803	0.01	0.3813	**0.8489**	**0.8406**	0.076	0.0087	0.638

Table 1: The obtained results using single-task and joint training of both Sub-Tasks 1 and 2. The best performances are highlighted with bold font. The attached superscript ‡ denotes the results of our **two official submissions** to both Sub-Tasks 1 and 2 (TEST).

	Sub-Task 1 (Word complexity)					Sub-Task 2 (MWE complexity)				
	Pearson	**Spearman**	**MAE**	**MSE**	**R2**	**Pearson**	**Spearman**	**MAE**	**MSE**	**R2**
w/o attention	0.7584	0.7316	0.1089	0.0171	**0.485**	0.8323	0.8335	0.0941	0.094	**0.6632**
w/o auxiliary loss ($\mathcal{L}_{aux}$)	0.7597	0.7198	**0.0695**	**0.0082**	0.3176	0.8382	0.8352	**0.0692**	**0.0074**	0.6167
w/o uncertainty loss weighing	0.7694	0.7321	0.0728	0.0088	0.4623	0.8472	0.8401	0.0797	0.0103	0.6071
Model	**0.7779**	**0.7366**	0.0803	0.01	0.3813	**0.8489**	**0.8406**	0.076	0.0087	0.638

Table 2: Ablation study of our model's component using joint training and RoBERTa-large as encoder (symbol w/o denotes without the corresponding component). **w/o uncertainty loss weighing** corresponds to the simple combination of model losses ($\mathcal{L}_{total} = \mathcal{L}_{rmse} + \mathcal{L}_{aux}$).

Ba, 2015) with a learning rate of 1×10^{-5}. The batch size and the number of epochs are fixed to 16 and 5, respectively. We investigate both single-task training and joint training of both Sub-Task 1 and Sub-Task 2 (training a single model on both subtasks data). All models are trained on the full train sets, validated on the trial sets, and evaluated on the test set of each Sub-Task. For evaluation purpose, we use the shared task's evaluation metrics, namely the **Pearson** correlation, the **Spearman** correlation, the Mean Absolute Error **MAE**, the Mean Squared Error **MSE**, and the coefficient of determination **R2**.

4.2 Experiment Results

Table 1 presents the obtained results of our model for both single-task and joint training, using the four transformer-based encoders. The overall results show that training joint models for both Sub-Tasks (1 and 2) outperform their single-task counterparts. The Use of deep encoders (large encoders) in our model yields better correlation performances. The best results for the correlation metrics are obtained using joint training and RoBERTa-large. For Sub-Task 1, the best MAE, MSE and R2 performances are achieved using single-task training and RoBERTa-large encoder. For Sub-Task 2, the best performances of all evaluation measures are obtained using joint training. In accordance with Sub-Task 1, the best correlation performances are attained using RoBERTa-large encoder. Besides, the best R2 is achieved using BERT-large, while the top MAE and MSE performances are obtained using RoBERTa-base.

To sum up, the best performances are obtained by joint training of our model on top of a deep encoder. These results can be explained by the fact that deep encoders yield better input representation for both Sub-Tasks. The joint training helps to leverage signals from both Sub-Tasks.

4.3 Ablation Experiment

In order to assess the effectiveness of each component of our model, we perform an ablation study using joint training and RoBERTa-large encoder. Table 2 illustrates the results of our model's ablation study. The results show that all components in our model improve the system performance. The auxiliary loss improves the performances of correlation measures, while it degrades MAE, MSE, and R2 performances. Combining RMSE and auxiliary losses using uncertainty loss weighting slightly improves the performance of correlation measures.

5 Conclusion

In this paper, we have presented our submitted system to the SemEval-2021 Task 1. The proposed system consists of a deep learning model for word and MWE complexity prediction. Our model employs a residual attention block and a regression module on top of a pre-trained transformer encoder. We have trained the model to minimize the uncertainty weighted loss of the RMSE and the auxiliary loss associated to the negative Pearson correlation. Experiments are performed using the base and the large variants of the pre-trained BERT and RoBERTa encoders. The best performance is obtained using RoBERTa-large encoder while performing joint training on both Sub-Tasks data.

References

Sandra Aluísio and Caroline Gasperin. 2010. Fostering digital inclusion and accessibility: The PorSimples project for simplification of Portuguese texts. In *Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas*, pages 46–53, Los Angeles, California. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jan De Belder and Marie-Francine Moens. 2010. Text simplification for children. In *Prroceedings of the SIGIR workshop on accessible search systems*, pages 19–26. ACM; New York.

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.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2017. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *CoRR*, abs/1705.07115.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

G.H. Paetzold and L. Specia. 2017. A survey on lexical simplification. *Journal of Artificial Intelligence Research*, 60:549–593.

Gustavo H. Paetzold and Lucia Specia. 2016. Unsupervised lexical simplification for non-native speakers. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 3761–3767. AAAI Press.

Jipeng Qiang, Y. Li, Y. Zhu, Yunhao Yuan, and X. Wu. 2020. Lsbert: A simple framework for lexical simplification. *ArXiv*, abs/2006.14939.

Francesco Ronzano, Ahmed AbuRa'ed, Luis Espinosa Anke, and Horacio Saggion. 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@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 1011–1016. The Association for Computer Linguistics.

Horacio Saggion. 2017. Automatic text simplification. *Synthesis Lectures on Human Language Technologies*, 10(1):1–137.

Matthew Shardlow. 2014. A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications(IJACSA), Special Issue on Natural Language Processing 2014*, 4(1).

Matthew Shardlow, Michael Cooper, and Marcos Zampieri. 2020. Complex - A new corpus for lexical complexity predicition from likert scale data. *CoRR*, abs/2003.07008.

Matthew Shardlow, Richard Evans, Gustavo Paetzold, and Marcos Zampieri. 2021. Semeval-2021 task 1: Lexical complexity prediction. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2021)*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Marcos Zampieri, Liling Tan, and Josef van Genabith. 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@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 1001–1005. The Association for Computer Linguistics.