Assembly Models for SimpleText Task 2: Results from Wuhan University Research Group

Jianfei Huang[1], Jin Mao[2]

[1]School of Information Management, Wuhan University, Bayi Rd 299, Wuhan, Hubei, 430072, China
[2]Center for Studies of Information Resources, Wuhan University, Bayi Rd 299, Wuhan, Hubei, 430072, China

Abstract

The goal of SimpleText Task 2 is to sort and rank complex terms that are required to be explained, given a passage and a query. To this end, our group applied a pipeline of term recognition and complexity evaluation. Candidate terms are extracted and evaluated according to their similarity with the query and a few rules. We formulate the evaluation of complexity as a classification task. We compile three groups of features for terms, including lexical, syntactic, and semantic features, then, ensemble machine learning models that adopt a soft voting strategy are applied to classify the complexity of the terms. Results of cross-validation on the training set are reported. Potential further improvements about the approach in future are discussed as well.

Keywords

term recognition, lexical features, syntactic features, semantic features, text complexity

1. Introduction

SimpleText Task 2 involves identifying what term is unclear and ranking terms that are crucial for readers to understand scientific text, given a passage and a query. In fact, for ranking terms that bother readers without prior domain knowledge, we need to know which terms should be extracted and explained. Further, evaluating term complexity could be a prior step for text simplification according to Shardlow's proposed approaches[1], as what to do in *SimpleText Task 3*.

Readers who do not understand the background of news articles often need to start with some technical terms. A term may consist of one or many words. It could be a strange word, an uncommon abbreviation, or a phrase. Apparently, a complex term cannot be understood just by its counts in some specific corpus. Its meaning relies on many features and differs according to context. To remove such understanding barriers, the goal of *SimpleText Task 2* is to decide which terms need explanation in a passage concerning a query and to rank them by three-level scores and five-level scores[2]. The task can be divided into two subtasks concerning all the above factors. One is extracting complex terms from a combination of passage and query. The other is evaluating complexity by considering valid influencing factors as much as possible.

In this paper, we extract key phrases and words based on similarity measures and rules, and

CLEF 2022: Conference and Labs of the Evaluation Forum, September 5–8, 2022, Bologna, Italy
hngdoze@gmail.com (J. Huang); danveno@163.com (J. Mao)
0000-0003-1125-4754 (J. Huang); 0000-0001-9572-6709 (J. Mao)

present our submission using two ensemble models to complete the complexity classification tasks. The former considers a large set of linguistics features, such as lexical features, syntactic features, and semantic features. The latter has nothing different but adding the prediction result of the former as a feature. In section 2 we introduce previous relevant works. In section 3 we present the main points of our feature engineering. In section 4 we show the basic flow of our model. Finally, in Section 5 we have some discussions.

2. Related works

2.1. Term Recognition

Terminology recognition methods can mainly be divided into traditional algorithms, classical machine learning, and deep learning models. Different methods have different application scenarios according to the tasks and data. Robertson explained the term-weighting function *TF-IDF* from a theoretical level, which was considered one of the most commonly used baselines for term recognition in information retrieval models[3]. Some studies have applied PageRank to keyword extraction and achieved good performance[4]. In addition, some studies focused on the clustering approach and classic machine learning classifiers, such as the Bayesian and support vector machine approaches[5]. Further, many recent works turned to the black box of deep learning, like using the pre-trained models, e.g., BERT. Deep learning approaches have shown promising results.

2.2. Term Complexity

Terminology complexity is closely related to the study of text complexity. In early studies, computational measures of text complexity have been restricted to some heuristic readability formulations, which mainly focus on some shallow features[6]. The shallow features usually adopt traditional readability metrics by simply counting words and characters[7], such as an average number of syllables per word, an average number of words per sentence, Automated Readability Index[8], and the Flesch-Kincaid score[9]. Later, some studies attempted to dig out deeper and more general features to supplement those shallow features.

In recent years, adopting machine learning or deep learning methods to complete feature learning for text complexity has become a trend. Gooding and Kochmar presented CAMB based on ensemble voting, a system that brings together 27 lexical, morphological, and psycholinguistic features[10]. Although it achieved state-of-the-art results in the *2018 CWI shared task*[11], it dismissed the context of the target words. In the *SemEval-2021 shared task 1*[12], most studies tented to capture extensive information for the target word and its context. Morphosyntax features and pretraining embedding were applied to obtain better feature representation. The model that attained the best performance in the above task, used both token and context features derived from pre-trained models [13]. However, an expanded version of the CAMB system obtained a similar performance[14]. It ranks third and is less than a percentage point below the best result on lexical complexity prediction for single words, which showed some feature engineering-based models can outperform most deep learning-based counterparts. Nonetheless,

combining various features and machine learning models seems to be a consensus in recent studies.

3. Methodology

3.1. Term Recognition

To get candidate terms, we first extracted keywords and phrases in the passages via *KeyBERT*[1]. A few similar algorithms can extract candidate terms, including *TF-IDF*, *Rake*, *YAKE!*. While,*KeyBERT* computes the cosine similarity of sub-phrases and passages internally, which is more in line with the task description. Then, the candidate terms were filtered by calculating the similarity scores between the terms and the query with PhraseSimilarity[2]. And we excluded those starting with a, an, the, or digit in the candidate terms. We also detected the capitalization of terms to extract acronyms. The terms obtained include words, compound words, phrases, etc. We then removed the punctuations and reverted the terms to lowercase except for acronyms.

3.2. Feature Extraction

We designed a few lexical features, syntactic features, and semantic features for the terms.

3.2.1. Lexical Features

These are features based on lexical information about the term:

- `length`: Length of the term.
- `zipf frequency`[3]: To make word frequency norms comparable, Brysbaert Marc et al provide the *Zipf Scale*, which is independent of corpus size[15]. Zipf frequency exactly aims to return the term's frequency on a human-friendly logarithmic scale via that.
- `tf-idf score`: We calculated tf-idf score based on *PhraseFinder*. *PhraseFinder* is a search engine for the Google Books Ngram Dataset (version 2) that features a wildcard-supporting query language and outstanding retrieval performance.
- `acronym`: Check if all letters are uppercase. Because acronyms are often difficult to understand.
- `number of subwords`[4], `syllables`[5], `phonemes`[6]: Morphological awareness is an understanding of how words can be broken down into smaller units[16]. The number of subwords is expected as a complementary feature to the length of the term and we get it via *BPEmb*, which is trained on Wikipedia and using the Byte-Pair Encoding algorithm. Similarly, the other two features are well-represented in speech synthesis and are widely incorporated into measures or feature sets in other studies on lexical complexity.

[1] https://github.com/MaartenGr/KeyBERT
[2] https://github.com/franplk/PhraseSimilarity
[3] https://pypi.org/project/wordfreq/
[4] https://github.com/bheinzerling/bpemb
[5] https://github.com/Kyubyong/g2p
[6] https://pypi.org/project/syllables/

3.2.2. Syntactic Features

Complex terms may have some special syntactic roles in the sentences. We coined a few syntactic features from the syntactic structure of a term's context. We used stanza[7] for part-of-speech recognition and dependency parsing.

- `depth of the term`: It means the distance between the term and the parse tree's root.
- `number of the dependencies`: We count all words that depend on or are depended on by the term, as this feature.
- `part-of-speech`: We use a 17-dimension one-hot vector to represent it, and each dimension represents one kind of part-of-speech tag. Some words have simple meanings, but when combined into phrases their meanings are elusive. Prepositional phrases, verb phrases, noun phrases, and adjective phrases have subtle differences in our understanding of the meaning of phrases. For phrases, what we do is add the vectors together, therefore we put both single words and phrases in the 17-dimensional vectors for comparison.

3.2.3. Semantic Features

- `glove embedding`[8]: We extract 300-dimension embeddings pre-trained on Common Crawl. Further, we use the zero vector to fill missing values and reduce the dimensions to 30 by PCA.
- `fasttext embedding`[9]: Fasttext embedding is considered as an alternative semantic feature. The dimensions are reduced to 30 by PCA as well.

3.3. Model Design

We formulated the complexity evaluation of terms as two classification tasks with 3 classes and 5 classes respectively. For the former, we concatenate all features and get 86 dimensional vector as the input vector. We put the predicted label of the three-classification model and all features together for the latter. Considering a large number of features and the small training set, we trained a few state-of-the-art base models, including *LightGBM*, *CatBoost*, *XGBoost*, *Random Forest*, *Support Vector Machine*, and then assembled these models using a soft voting strategy. On the one hand, the ensemble model consists of multiple classifiers, which improves the accuracy of the classification task. On the other hand, ensemble models reduce the occurrence of special cases, such as predicting difficult terms into simpler ones. Figure 1 gives an overview of the model design. Hyperparameter settings either use grid search or follow default values.

4. Results

The terms provided in the training samples are not independent, in other words, a term can correspond to multiple passages. We deduplicated records and obtained 250 independent sentence-term pairs as the final dataset. Then, we performed five-fold cross-validation on the

[7] https://stanfordnlp.github.io/stanza/
[8] https://nlp.stanford.edu/projects/glove/
[9] https://fasttext.cc/

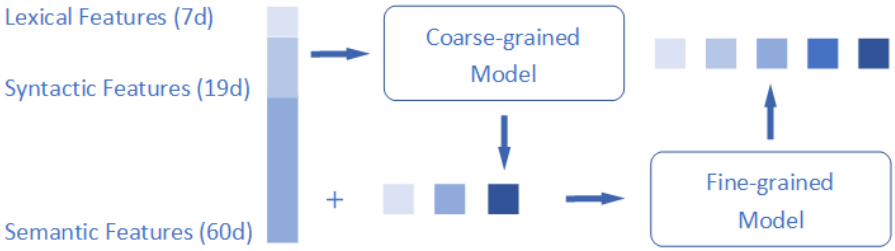

Figure 1: Overall model design: term complexity assessment at simpletext task 2.

dataset. According to the model design, we first verified the models for the three-class task, and the results are shown in Table 1. The star represents our proposed integrated model. It is shown that the integrated model is superior to the base models in terms of accuracy and AUC.

Table 1
Cross-validation results of the three-class task.

Three-classification Model	Accuracy		F1 Score		AUC	
	mean	std	mean	std	mean	std
* (Integrated Model)	**0.684**	0.062	0.583	0.093	**0.635**	0.059
LightBGM	0.652	0.063	0.586	0.089	0.624	0.062
* - LightBGM	0.660	0.083	0.565	0.089	0.607	0.069
CatBoost	0.636	0.069	0.551	0.064	0.615	0.058
* - CatBoost	0.672	0.079	0.583	0.093	0.611	0.069
XGBoost	0.656	0.093	**0.593**	0.112	0.591	0.061
* - XGBoost	0.668	0.084	0.576	0.096	0.631	0.064
RandomForest	0.656	0.097	0.556	0.080	0.590	0.098
* - RandomForest	0.664	0.066	0.581	0.090	0.626	0.055
SVM	0.672	0.079	0.557	0.080	0.576	0.098
* - SVM	0.660	0.072	0.582	0.101	0.621	0.062

Intuitively, five grading scales are more difficult, which require a more precise assessment of complexity. We take the prediction results of the three-class models as the extended input feature, which can improve the performance. We also obtained the accuracy, F1 score, and AUC value for the ensemble models of the five-class task, as shown in Table 2.

The accuracy metrics of the two ensemble models we designed outperform the other base models. On F1 scores and AUC metrics, they also achieved almost the best performance in the experiment. Furthermore, according to the subset of the test set consisting of 592 sentences manually annotated, our submissions are ranked second(2/4) on the scale 1-3 and first(1/4) on the scale 1-5, based on the proportion of successful matches of all participants. In the subset consisting of 167 common sentences, we ranked second in both tasks. [2]

Table 2
Cross-validation results of the five-classification model.

Five-classification Model	Accuracy		F1 Score		AUC	
	mean	std	mean	std	mean	std
* (Integrated Model)	**0.464**	0.064	**0.445**	0.068	0.670	0.025
LightBGM	0.448	0.060	0.414	0.076	0.653	0.023
* - LightBGM	0.428	0.072	0.375	0.085	0.656	0.026
CatBoost	0.428	0.060	0.376	0.084	0.668	0.028
* - CatBoost	0.440	0.067	0.389	0.090	0.667	0.027
XGBoost	0.448	0.057	0.415	0.081	0.657	0.029
* - XGBoost	0.432	0.053	0.389	0.086	0.663	0.028
RandomForest	0.460	0.052	0.402	0.078	0.654	0.047
* - RandomForest	0.428	0.060	0.377	0.086	**0.673**	0.023
SVM	0.440	0.057	0.351	0.061	0.581	0.027
* - SVM	0.432	0.059	0.399	0.081	0.662	0.026

However, the evaluation results of all participating teams performed poorly. One reason for this could be that the term extraction process is not proper. Many terms are manually annotated as requiring no explanation during the evaluation process and assigned a new difficulty score of 0, whereas they are assigned a difficulty score of 1 in our submissions, implying that they belonged to the easiest terms. Admittedly, the values of all these metrics are not high, indicating that the tasks of identifying terms and predicting term complexity are difficult.

5. Discussion

In this paper, we applied a pipeline for the term complexity prediction tasks, which consists of term recognition, feature extraction, training models, and assembling models. The ensemble models show improved performance than the base models.

As a preliminary study, a few limitations have been identified, which could guide our future refinement for our approach. The pre-trained embedding we choose is trained on *Common Crawl*, which is from the public domain. There can be pre-trained word embeddings for technology and medical fields, as are the domains covered by the task corpus. Thus, one work direction is to fine-tune a pre-trained model based on transformer architecture on a specific corpus of the target domain and to extract the learned embeddings as a complement to semantic features. Furthermore, our method takes into account some insignificant features, and there may be some important features that have not been identified. Evaluating the importance of features and emphasizing significant features in the learning models could further improve the approach.

References

[1] M. Shardlow, A survey of automated text simplification, International Journal of Advanced Computer Science and Applications 4 (2014) 58–70.

[2] L. Ermakova, P. Bellot, J. Kamps, D. Nurbakova, I. Ovchinnikova, E. SanJuan, E. Mathurin, R. Hannachi, S. Huet, S. Araujo, Overview of the CLEF 2022 SimpleText Lab: Automatic Simplification of Scientific Texts, Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Thirteenth International Conference of the CLEF Association (CLEF 2022) 13390 (2022).

[3] S. Robertson, Understanding inverse document frequency: on theoretical arguments for idf, Journal of documentation (2004).

[4] J. Wang, J. Liu, C. Wang, Keyword extraction based on pagerank, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2007, pp. 857–864.

[5] D. Isa, L. H. Lee, V. Kallimani, R. Rajkumar, Text document preprocessing with the bayes formula for classification using the support vector machine, IEEE Transactions on Knowledge and Data engineering 20 (2008) 1264–1272.

[6] D. S. McNamara, Y. Ozuru, A. C. Graesser, M. Louwerse, Validating coh-metrix, in: Proceedings of the 28th annual conference of the cognitive science society, 2006, pp. 573–578.

[7] S. Jönsson, E. Rennes, J. Falkenjack, A. Jönsson, A component based approach to measuring text complexity, in: The Seventh Swedish Language Technology Conference (SLTC-18), Stockholm, Sweden, 7-9 November 2018, 2018.

[8] R. Senter, E. A. Smith, Automated readability index, Technical Report, Cincinnati Univ OH, 1967.

[9] G. R. Klare, Assessing readability, Reading research quarterly (1974) 62–102.

[10] S. Gooding, E. Kochmar, 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, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 184–194.

[11] S. M. Yimam, C. Biemann, S. Malmasi, G. H. Paetzold, L. Specia, S. Štajner, A. Tack, M. Zampieri, A report on the complex word identification shared task 2018, arXiv preprint arXiv:1804.09132 (2018).

[12] M. Shardlow, R. Evans, G. H. Paetzold, M. Zampieri, Semeval-2021 task 1: Lexical complexity prediction, arXiv preprint arXiv:2106.00473 (2021).

[13] C. Pan, B. Song, S. Wang, Z. Luo, DeepBlueAI at SemEval-2021 task 1: Lexical complexity prediction with a deep ensemble approach, in: Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), Association for Computational Linguistics, Online, 2021, pp. 578–584.

[14] A. Mosquera, 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), 2021, pp. 554–559.

[15] M. Brysbaert, E. Keuleers, M. Stevens, L. Van der Haegen, A. Verma, M. Callens, W. Tops, V. Khare, P. Mandera, H. Vander Beken, et al., The zipf-scale: A better standardized measure of word frequency, Update (2013).

[16] S. H. Deacon, M. J. Kieffer, A. Laroche, The relation between morphological awareness and reading comprehension: Evidence from mediation and longitudinal models, Scientific Studies of Reading 18 (2014) 432–451.