UDC 004.94

M. Bielikov, T. Likhouzova, Y. Oliinyk

MODELS FOR ANALYZING THE COMPLEXITY OF ENGLISH WORDS IN THE TEXT ON THE SCALE FROM A1 TO C2

Abstract: At the current stage of globalization, English plays a key role as the language of international communication. This leads to the fact that more and more people become its carriers at various levels. The work is devoted to the analysis of English words on the scale from A1 to C2, which corresponds to the lowest and highest levels of proficiency according to the CEFR standards. A model that predicts the difficulty of words in a text can be used to improve the educational process. For example, it is possible to find a list of likely unknown and difficult words for the end user in any text depending on his level of English language proficiency. This approach will facilitate the language learning process by providing a personalized list of words to focus on. Also, the model can be useful for analyzing the complexity of texts depending on the number of words of each level of complexity in them. This can help teachers prepare materials that match the level of knowledge of their students, as well as identify words that may be difficult for them to understand.

An application in the Python programming language is proposed, which receives a sample of data from the created storage, displays them graphically, performs intellectual analysis, trains and compares models according to accuracy, precision, recall and f1-score metrics. For data analysis and prediction of the level of complexity of English words, the following models were used: PchipInterpolator, logarithmic model, Gradient Boosting, Random Forest and XGB.

Keywords: intelligent data analysis, text analysis, classification problem, accuracy metrics.

Introduction

In today's world, mastering the English language is considered a key competency that ensures success in various areas of life, including education, work, and communication. This leads to a constant interest in the study and development of language proficiency assessment methods. One of these methods is the scale from A1 to C2 recommended by the European Language Education Standards (CEFR) [12].

Frequency analysis allows you to identify those words that are most characteristic for each level of language proficiency. For example, at the A1 level it can be basic words for everyday communication, while at the C2 level it is specialized vocabulary.

 ISSN 1560-8956

Materials and methods

Five open sources were chosen as raw data:

– information about the frequency of words in texts that were published in periodicals and books collected by Google Books: https://books.google.com/ngrams/info;

– dataset with levels of word complexity set by English language teachers from Zenodo: https://zenodo.org/records/12501;

– a dataset with levels of word complexity from A1 to B2 from Tokyo University of Foreign Studies: https://cefr-j.org/download.html#cefrj_wordlist;

– a dataset with marked levels of word complexity from C1 to C2 from Octanove Labs and RIKEN: https://www.openlanguageprofiles.org/datasets/;

– dataset of English words saved in txt format from the Infochimps company: https://github.com/dwyl/english-words.

Based on the analysis of the subject area, a data storage model of the "snowflake" type was developed (Fig. 1). The center of the snowflake is the word_pos table, which contains information about the part of speech of each word.

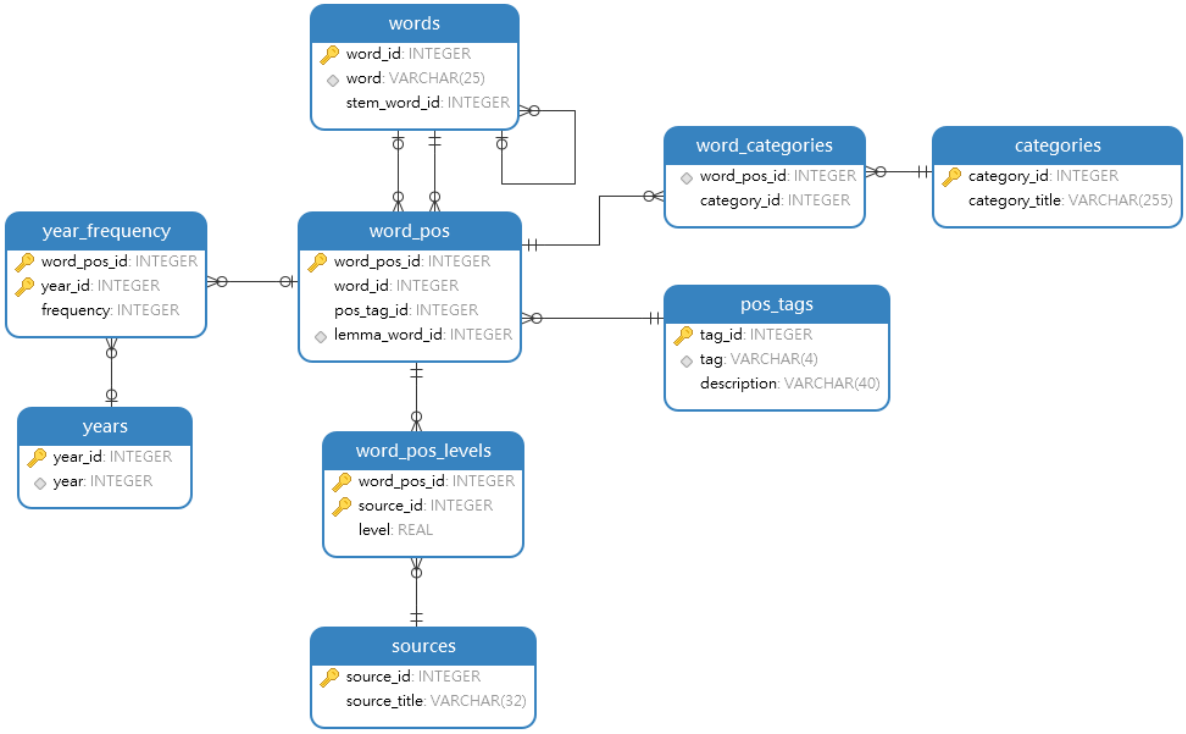


Figure 1. Calculation of stock volatility

The constructed database storage model has a number of important advantages. First, indexed database fields allow you to get data in approximate O(1) time instead of O(log n). Second, the database can be easily expanded. For example, you can add new word difficulty

information from another source by adding the source name to the source table and the data to word_pos_levels. This approach of adding new data will not affect the current results from other datasets and will allow analyzing the data within each dataset separately. Word frequency statistics may also be expanded in the future by adding new entries.

The main disadvantage of such an architecture is the large size of the database. For example, if you analyze data from 1900 to 2019, then for each word as a part of speech there will be up to 120 records in the year_frequency table. But replacing the statistics by year with a single field with the total frequency will make it inconvenient to supplement it for the frequencies of new words in the future and will make it impossible to analyze trends in the use of words over the years.

Based on the designed model, a data store was created using SQLite3.

Data pre-processing

For the design and further work with the data warehouse, it is necessary to take into account a set of methods that implement the process of transferring the initial data into an analytical format that will be maintained in the data warehouse and will not violate the integrity of the system. The process of downloading and preparing data is divided into separate steps: processing and uploading a list of English words to the database, parsing information about their frequency, and downloading data from datasets with levels of difficulty of English words.

The first stage involves preparing a list of English words before uploading them to the database. The importance of this step is that the Google Ngrams word frequency dataset [2] contains a large number of non-existent words. This was due to the use of OCR (optical character recognition) technology for some books during the creation of the dataset. So, it was necessary to prepare a list of valid words, and only then to parse the data for them from the Google Ngrams dataset.

You can also calculate the stem of each word at this step. Stemming is the process of separating affixes (prefixes, suffixes, endings) in words to obtain their basic form. This is necessary in order to set the complexity of words as their base form.

For the English language, there are 3 most popular algorithms for determining the basic form of a word: Porter Stemmer, Lancaster Stemmer and Snowball Stemmer [5]. But none of these algorithms are perfect. They are based on rules and heuristics that do not always reflect all features of word formation. Some words can be incorrectly stemmed, which can lead to the formation of invalid words. Also, the disadvantage is that they cut off only suffixes and endings, not paying attention to prefixes. To fix this, a list of all possible prefixes for words from the Cambridge Dictionary website [8] was used, and in the case when a word begins with one of these prefixes, it was cut off. To avoid mistakes,

 ISSN 1560-8956

requirements have been developed for which the base of the word will be considered correct. First, the word must be present in the list of valid words that we obtained from the previous step. Secondly, in the case of differences in the result between the three algorithms, the form of the word that has the highest frequency of use from the Google Ngrams dataset will be considered. In the theoretically possible case, when the frequency of use of two different versions of the proposed basic forms is the same, we will choose the version of the word that will have a shorter length. Third, if a word has a higher frequency of occurrence than its base form, then the base form will not be considered. In this case, an existing word that is not the basic form is most often formed. For example, for the word "page" the algorithm issued "pag", which is not correct. But valid cases rarely occur. The word "authority" has the base form "author", which is indeed the base form and is less common. In this case, it makes no sense to consider the possibility of determining the level of complexity of a derived word based on the frequency of use of the base form, if it occurs less often and can significantly increase its complexity. Also, in such cases, very often the meaning of the derived word is very different from its basic form. Therefore, the level of complexity of the base form can be higher than that of the derivative. It is worth noting that no attempt was made to determine the complexity of the basic form based on the derivative, because it is not always possible to know the basic form by knowing the derived form.

To take into account all the previous points, it was decided to use the criterion of the validity of the stem-form of the word relative to the derivative, which will be calculated according to the following formula:

$$stemValidProbability = \frac{1}{1 + e^{-\ln\left(\frac{stemFrequency}{wordFrequency}\right)}}$$

This criterion is convenient in that it allows you to reject most of the invalid base forms of words. The logarithm is used in order to take into account the non-linear nature of the dependence between the frequencies of use. The criterion will be close to 1 when the popularity of the current word is significantly less than the base form; close to 0, when the popularity of the current word is much greater than the base word; 0.5 when the word frequencies are the same. If we consider the case with the word "page" and the base form "pag" proposed by the algorithms, the result of this function will be 0.0044. Therefore, the derived form is most likely invalid. For the case of "numbers" and "number" - 0.8523. Therefore, the derived form is probably valid. For the previously described word “authority” and the proposed basic form “author” – 0.3839. This example shows a case where we got a wrong result according to the developed criterion. But such cases are quite few. The view of the distribution of the criterion for words for which the basic form was proposed by the algorithms can be seen in Figure 2. Figure 3 shows a scatter plot of word frequencies and their basic forms. It can be seen that the number of base forms, which are more popular than

the derivative, is significantly higher, because most of the points are in the lower left corner. As a result of experiments with the values of this criterion, at which the basic form can be considered valid, it was noticed: if we accept only probably correct results (the criterion is greater than 0.8), then quite a lot of existing basic forms for unpopular words are rejected. After several tests, it was decided to consider the basic form valid when the value of the criterion is greater than 0.5. That is, this is the case when the base form is more popular than the derivative.

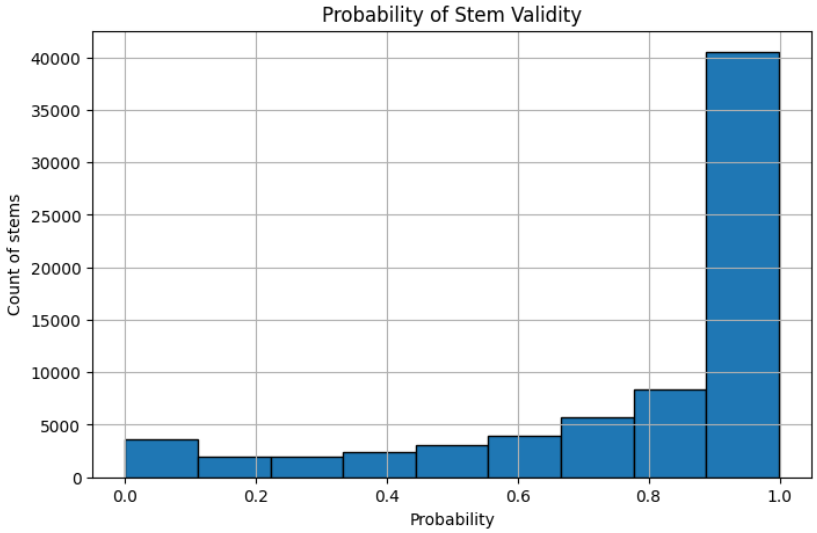


Figure 2. Distribution of values of the validity criterion of basic word forms

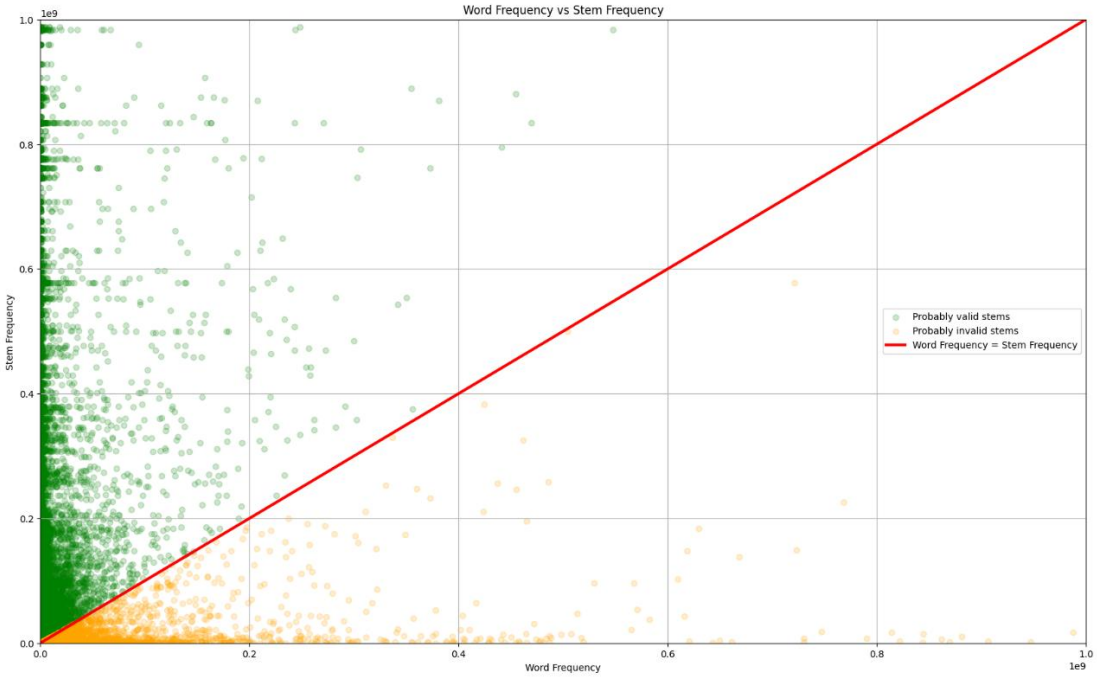


Figure 3. Scatter plot of word frequencies and their basic forms

ISSN 1560-8956

The next step was to create stage zone database tables and download a list of words with their stem forms. Next, it is necessary to parse from the Google Ngrams dataset information about the frequency of words when they appear as certain parts of speech. Each word was assigned one of 10 part-of-speech tags: NOUN, VERB, ADJ, ADV, PRON, DET, ADP, NUM, CONJ, PRT. But the problem is that these tags do not display all possible word forms. It was decided to use the Penn Treebank Project [7] standard, which is used by most NLP libraries, such as NLTK, SpaCy, and CoreNLP. For example, there are not one, but 6 different tags for marking verbs depending on the form of the verb (ending in -ing, past tense, etc.). To move from the current list of tags to the Penn Treebank Project standard, a dictionary of all possible instances of a part-of-speech tag was developed. In the same way, tags from word complexity datasets were brought to the Penn Treebank Project standard.

Another method for determining the basic forms of words - lemmatization - was also carried out at this stage. This process uses language dictionaries, rules of morphology, and a part-of-speech tag to determine the base form of a word. First, it allows you to handle special exceptional cases in English that stemming cannot handle. For example "mice" and "mouse". Secondly, the obtained results are true with a high probability. The LemmInflect library is responsible for all of this. All its results will be considered valid if the word exists in the list of words collected in the previous stages. The library was tested on the Automatically Generated Inflection Database (AGID), which contains 119,194 test examples. On average, it has an accuracy of 95.6% for all parts of speech [1].

Primary analysis of input data

The next step is to upload data from datasets with levels of word complexity to the stage zone of the database. The number of words of each difficulty level of each dataset can be seen in Figure 4. However, the difficulty of some words in the datasets was ranked differently. In such cases, the average word complexity among all datasets was chosen. The final word count statistics of each difficulty level can be seen in Figure 4 on the right. There were significantly more B1 and B2 level words than others. This is due to the fact that there are much more complex words in the language than simple ones. However, for levels C1 and C2 there is quite little data. It so happened that one of the largest CEFR-J datasets does not contain records for levels of difficulty above B2. And the dataset for complex words from Octanove Labs is quite small, because it contains only the main ones.

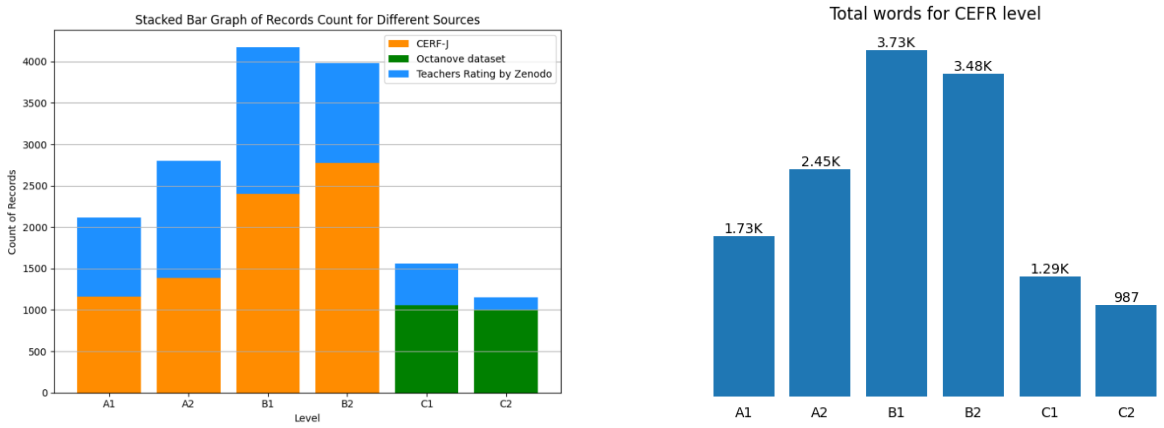


Figure 4. The number of words of each difficulty level

If you calculate the average frequency of words for each level of complexity relative to the parsed datasets, then in Figure 5 you can see the inverse dependence of the number of occurrences on the level of complexity. The more often a word is found in the English language, the lower the level of complexity it has.

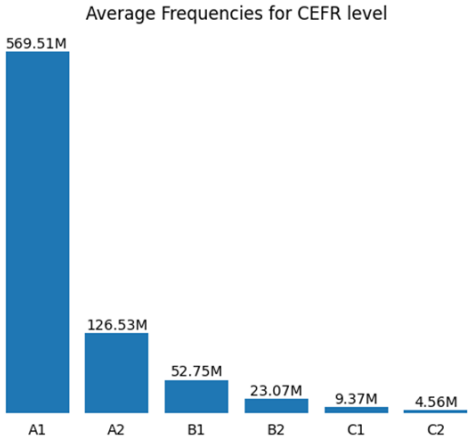


Figure 5. Mean word frequency for each difficulty level

At this stage, data preprocessing can be considered complete.

Results and discussion

The goal is to predict the difficulty level of English words on the CEFR scale based on their frequency in the English language. All models will be compared according to the following metrics: Accuracy, Precision, Recall and F1-score. The result of prediction will be word complexity from 1 (A1) to 6 (C2). For metric evaluations, the model output will be rounded to an integer within these limits. The dataset is divided into a training (80%) and a test sample (20%).

The first selected model is the PCHIP Interpolator (Piecewise Cubic Hermite Interpolating Polynomial) [6] – a model that provides a smooth connection between data

 ISSN 1560-8956

points, allowing for a more continuous and understandable function for prediction. To train the PCHIP Interpolator model, average word frequencies for each difficulty level will be provided as input. At the output of the model, we get a smooth function that passes through these points and provides continuous interpolation of values between them.

The second selected model is the logarithmic model. It is often used to approximate data where there is an exponential growth or decline. In our case, when the value of the argument (word frequency) increases, we expect a decrease in the level of complexity, which can be displayed using a logarithmic function.

After a more detailed analysis of the data, it can be understood that the first model will not accurately predict the data. This is because the average values of word frequencies for each level will be inflated because there are very popular words in each level. For example, the word "the" occurred 92805722256 times as part of the Determiner (DT) language between 1900 and 2019. These and other words will overestimate the average frequency for word difficulty levels.

The third model is identical to the first, but the best word frequency values for each difficulty level will be selected using the Simulated Annealing method.

The fourth, fifth and sixth models are improved versions of the previous three models, which will have a different function depending on the part of speech of the word.

The seventh model is Gradient Boosting Regressor [3], which will take into account word frequency and part of speech as a categorical predictor.

However, all previous models will give incorrect results for the unlikely case that a word of a certain difficulty level is too popular or not popular compared to other words. Then, accordingly, the predicted level will be higher or lower than the actual one. To reduce the impact of such cases, the meaning of the word must be taken into account. To improve the results, you can use the model of the SpaCy library “en_core_web_md” [10]. This model was trained on data from Common Crawl (web archives with data from web pages on various topics), OntoNotes 5 (corpus of annotated texts together with annotations of parts of speech, word dependencies and named entities) and WordNet (a large lexical database of the English language in which words grouped into sets of synonyms: synsets). The input of the model will be a word, and the output will be a 300-dimensional vector of this word (Word2Vec). Words that are similar in meaning have close vector representations in multidimensional space. The idea was to use the following predictors to predict the level of word complexity: word frequency, part of speech, word length and a vector obtained from the NLP model “en_core_web_md”.

Considering all the conditions above, 3 more models will be trained: Gradient Boosting Regressor [3], Random Forest Regressor [9] and XGB Regressor (eXtreme Gradient Boosting Regressor) [13].

The best model was chosen based on the F1-score metric, because it is the key to analyzing words in any text according to the CEFR scale. Due to the fact that the English language has an exhaustive list of words and parts of languages that each of them can represent, it is enough to predict the complexity of each word together with its part of speech once and enter the resulting complexity into the database. Then, during text analysis, it is not necessary to have a huge dataset with the frequency of each word, but only a compact dataset that will contain only 3 parameters: the word, its part of the language and the level of complexity from 1 to 6, which can be translated into the CEFR scale. Therefore, such a dataset will be very compact and more convenient to use. Also, this approach is faster, because the time of obtaining the value of the level of complexity from the database is much shorter than the time of forecasting a complex model, the input of which is given 303 predictors.

It should be noted that the model will predict the level of complexity only for basic word forms. That is, it is necessary to implement methods by which we can determine the complexity of derived words. At the beginning, it is worth taking words for which the correct level of difficulty is guaranteed. These are words from datasets that were chosen for model training. However, there are cases when the level of complexity is defined for a word of one part of the language, but not for the same word as another part of the language. For example, the word "warming" as a noun has a difficulty level of A2 (for example, in the sentence "The polar ice caps melt due to global warming"). However, the complexity value for the same word as a verb is absent (for example, in the sentence “She is warming her hands by the fireplace”). Therefore, it was decided to set the difficulty level as the average between the known cases of difficulty when the word appears as another part of speech.

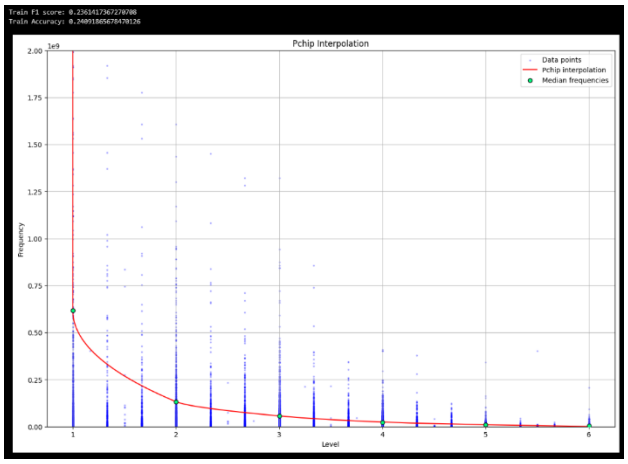
Application of models

Due to the fact that the number of words of each difficulty level is different, the metrics were calculated as a macro average - for each word difficulty level, precision, recall and f1-score are calculated, and then the average value of these metrics is calculated for all six classes. Thus, the macro average gives equal weight to each difficulty level.

The first model was a simple interpolation through the midpoints of word frequencies for each level. However, because some words are too popular, the values of these points are higher than they really should be. An f1-score of 0.2361 and an accuracy of 0.24 were obtained (Fig. 6).

The logarithmic model showed better results according to the accuracy metric than the first model, but the f1-score is still low (Fig. 7). According to the inconsistency matrix, it can be seen that it was concentrated only on levels B1 and B2, which were the most in the datasets. This model is also not suitable for predicting the difficulty level of words based on their frequency due to low metric scores.

 ISSN 1560-8956



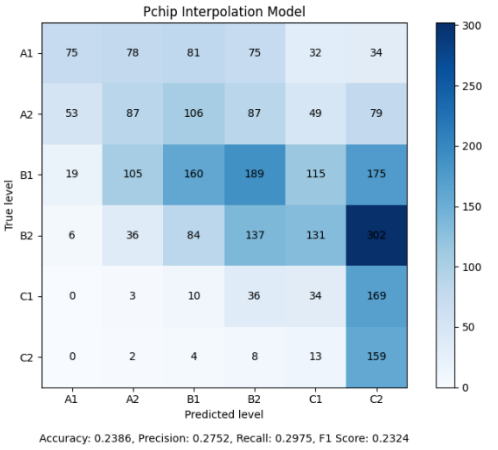
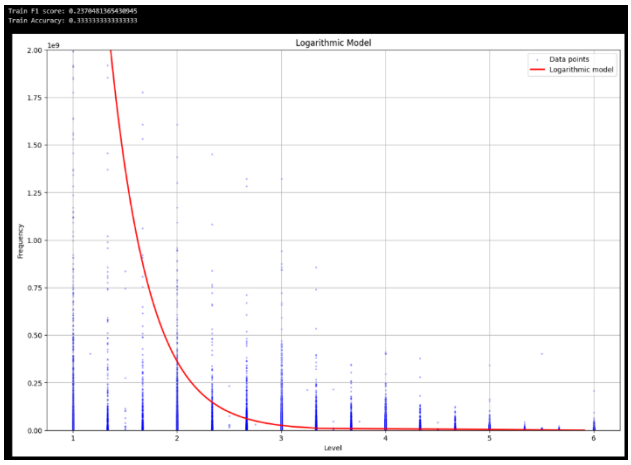


Figure 6. Results of training the PchipInterpolator model



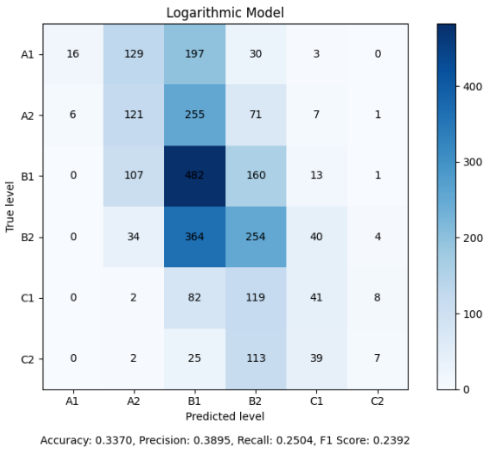


Figure 7. Results of training the Logarithmic model

For the next model, the same approach is used as for the first, but the points for interpolation were selected by the Simulated Annealing method. You can see that the f1-score is better than the previous models, but still very low. It is interesting that the points of levels 4.5 and 5.5 hardly differ in terms of values, and on the inconsistency matrix, the model almost did not predict level C1 (Fig. 8).

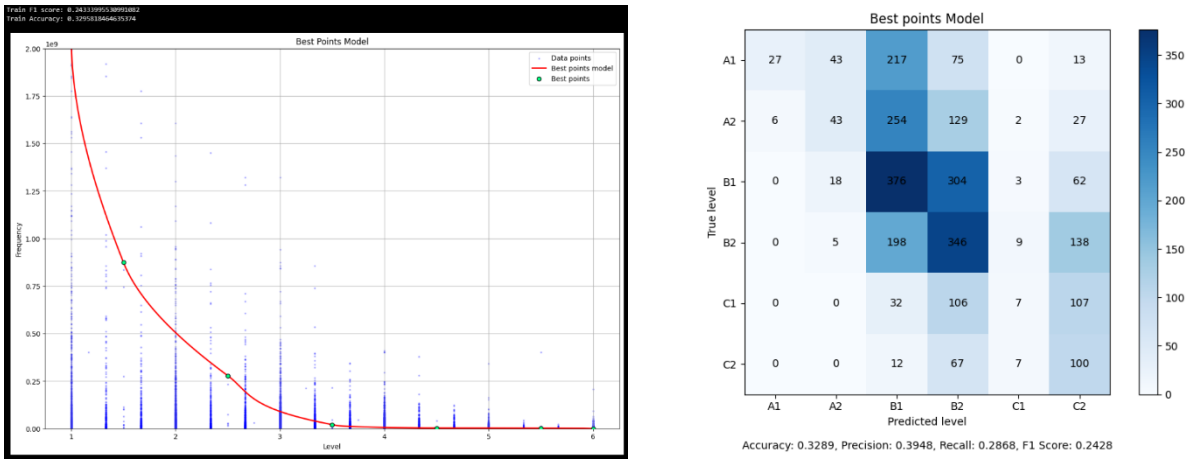


Figure 8. Results of training the Best Points model

After training the previous 3 models, you can understand that it is impossible to accurately predict the level of complexity of English words based only on the frequency of their occurrence. So, it is worth considering another predictor that affects the level of complexity of words - their part of speech. This can be argued by the fact that we have data with the frequency of use of each word as a specific part of speech, but in general some parts of speech occur more often than others. For example, in the English language, nouns occur 4 times more often than adjectives. So, for the previous 3 models, another predictor was added - the part of speech of the word.

The first model was extended by adding one more predictor. But the accuracy on the test sample almost did not change. The model began to predict level A1 better, but it also makes almost all predictions for level C2.

The training results of the improved logarithmic model were only marginally better. According to the inconsistency matrix, the overall result turned out to be very similar, despite the significant difference in the logarithmic curves.

The model that finds points using the Simulated Annealing method showed the best result - the value of f1-score on the test sample was 0.2759 and the accuracy was 35.64%. According to the discrepancy matrix, the model is highly concentrated at levels B1 and B2 and poorly predicts other levels of complexity.

We also tried to train the Gradient Boosting Regressor model and see how it copes with this task. However, it performed significantly worse than the previous models - the model made only 1 prediction for the C2 level, which was incorrect.

If you look at the overall obtained statistics for the models, you can see that it is unlikely to be able to train a model with an accuracy higher than 36% and an f1-score of 0.28. Improved models give only a small increase. These predictors alone cannot predict the level of word complexity.

 ISSN 1560-8956

Previous results have shown that word frequencies and part-of-speech information are fairly weak predictors. However, there is one of the most important predictors that people now use for this prediction: the difficulty of a word directly depends on its meaning. For example, specialized vocabulary will have a much higher level than simple words in terms of meaning. Therefore, it is worth applying NLP and also providing a word value vector obtained by the already trained “en_core_web_md” model of the SpaCy library.

The trained Gradient Boosting Regressor model showed an accuracy of 41.13% accuracy and 0.3338 f1-score. If you look at the discrepancy matrix, it is much better than in previous models. However, as in the previous case with the Gradient Boosting Regressor, there is only 1 forecast for level C2. Also, the model often confuses the level of complexity of a word with neighboring words.

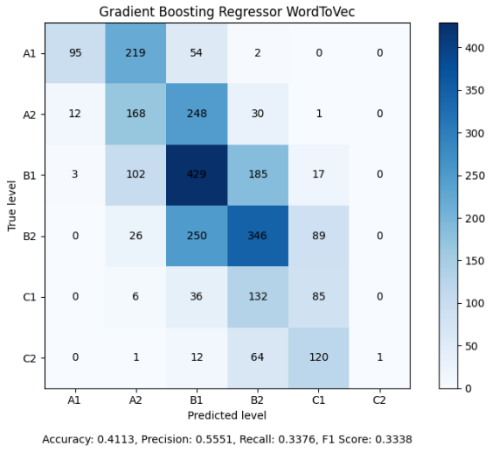


Figure 9. Results of training Gradient Boosting Regressor with SpaCy WordToVec

The Random Forest Regressor model showed a result of 56.79% accuracy and 0.5403 f1-score. From the correspondence matrix, we can see that the model predicts word difficulty levels quite well, but again there is a problem with C2 words, where the majority were predicted as C1. So far, this is the best model among all others (Fig. 10).

The XGB Regressor model showed a result of 56.57% accuracy and 0.5422 f1-score. From the discrepancy matrix, we can conclude that we predict C2 level words just as poorly. In general, the model is wrong within one level of complexity (Fig. 11).

The XGB Regressor performed best, so it was decided to select the best parameters for it using GridSearchCV. We obtained an accuracy of 58.54% and an f1-score of 0.58. Accuracy increased by 2% and f1-score by 0.04 compared to the model without parameter selection and without cross-validation.

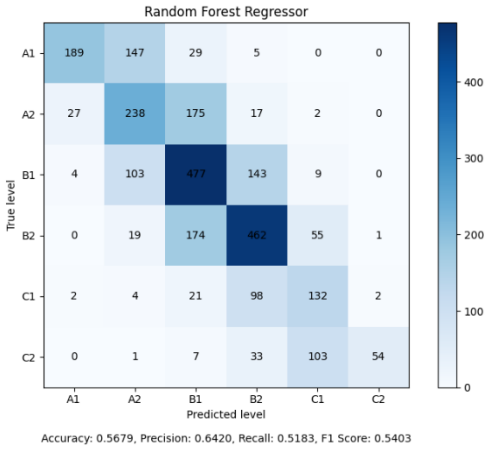


Figure 10. Results of training Random Forest Regressor with SpaCy WordToVec

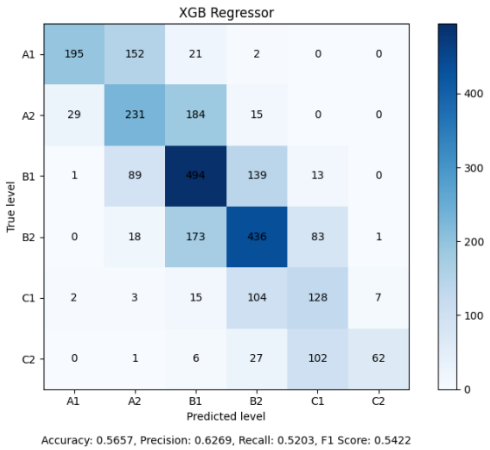


Figure 11. Results of training XGB Regressor with SpaCy WordToVec

Model comparison

General statistics based only on the basic metrics of accuracy and average macro precision, recall, f1-score for all models are shown in Fig. 12. The best models according to each of the metrics are highlighted in green.

Title	Accuracy	Precision	Recall	F1
Pchip Interpolation Model	0.2386	0.2752	0.2975	0.2324
Logarithmic Model	0.3370	0.3895	0.2504	0.2392
Best points Model	0.3289	0.3948	0.2868	0.2428
Advanced Pchip Interpolation Model	0.2309	0.2482	0.2950	0.2236
Advanced Logarithmic Model	0.3414	0.4034	0.2547	0.2435
Advanced Best Points Model	0.3564	0.3822	0.2847	0.2759
Gradient Boosting Regressor	0.3454	0.3673	0.2558	0.2402
Gradient Boosting Regressor WordToVec	0.4113	0.5551	0.3376	0.3338
Random Forest Regressor	0.5679	**0.6420**	0.5183	0.5403
XGB Regressor	0.5657	0.6269	0.5203	0.5422
Best XGB Regressor	**0.5854**	0.6182	**0.5636**	**0.5800**

Figure 12. General statistics on the main metrics for all models

 ISSN 1560-8956

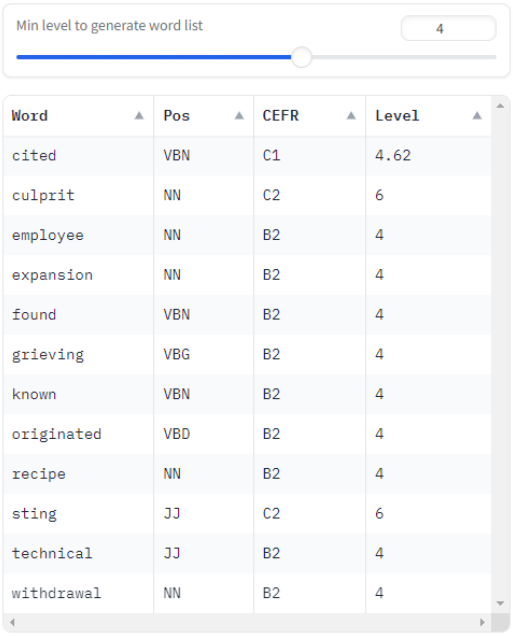
So, from the general statistics, it can be seen that according to the Precision metric, the Random Forest Regressor model with a value of 0.642 turned out to be the best. However, the Best XGB Regressor model with the best fitted parameters and cross-validation is significantly better in other metrics accuracy, recall and f1 score with values of 0.5854, 0.5636 and 0.58 respectively. Best XGB Regressor was selected as the best model for further prediction of word difficulty levels. Although the f1 score metric is not very high, it meets the accuracy requirements for the given task. The model error of about 1 level of complexity, which we can see in the discrepancy matrices, is not critical. If we consider the datasets with already set levels of difficulty, then even there different levels were set for some words and it was necessary to average the results between them.

The last step is to set the levels of difficulty for all words of the English language according to the developed algorithm. In total, 238,423 difficulty levels were set for words (not including already set values from datasets), and of these, 125,838 values for basic forms were predicted by the model.

In order to visually check the results of predicting word difficulty levels separately, data with words, parts of speech and assigned word difficulty levels were exported through Navicat 16 database software. Based on this, a Python package has been developed, which will process the input text and, receiving values from the dataset, set the appropriate level of complexity for each word (direct link https://pypi.org/project/cefrpy/ so that everyone can download it using the pip command). Also developed a visual web interface using gradio (direct link https://huggingface.co/spaces/Maximax67/cefrpy-demo).

The result of predicting the word complexity for the text can be seen in Fig. 13 and 14. Proper and geographical names were omitted with the help of NER from SpaCy.

Figure 13. The result of predicting the complexity of English words



Word	Pos	CEFR	Level
cited	VBN	C1	4.62
culprit	NN	C2	6
employee	NN	B2	4
expansion	NN	B2	4
found	VBN	B2	4
grieving	VBG	B2	4
known	VBN	B2	4
originated	VBD	B2	4
recipe	NN	B2	4
sting	JJ	C2	6
technical	JJ	B2	4
withdrawal	NN	B2	4

Figure 14. Words from the text of level B2 and above

Conclusion

The PchipInterpolator, logistic model, Gradient Boost Regressor, Random Forest Regressor, and XGB Regressor models were used to predict the difficulty levels of English words. At first, there was an attempt to predict only by the frequency of words. The best was the PchipInterpolator model with the best points found by the Simulated Annealing method (Best Points Model). According to the metrics, the accuracy was 32.89% and the f1-score was 0.2428. This approach showed very low results.

The next step was to try to predict using two predictors: word frequency and part of speech as a categorical variable. The Advanced Best Points Model turned out to be the best with an accuracy of 35.64% and an f1-score of 0.2759. The value of the metrics has become larger, but not so much that the model can be used for prediction in real problems.

The “en_core_web_md” model of the SpaCy library was applied to obtain NLP WordToVec vectors that were provided as input to the model along with word frequency, part of speech and word length. The Best XGB Regression model is trained with the selection of the best parameters and cross-validation. Its result is 0.58 according to the f1-score metric and the accuracy is 58.54% on the test sample. Based on this model, difficulty levels were predicted for all base word forms for which there was no value in the datasets. The derived forms of words have the same complexity as for the basic ones.

The analysis of the text in Figures 13 and 14 was carried out, from which it can be seen that the intended final task was successfully completed.

 ISSN 1560-8956

REFERENCES

1. Accuracy - LemmInflect. LemmInflect. URL: https://lemminflect.readthedocs.io/en/latest/accuracy/ (date of access: 25.05.2024).

2. Google Books Ngram Viewer. Google Books. URL: https://books.google.com/ngrams/info (date of access: 25.05.2024).

3. GradientBoostingRegressor. scikit-learn. URL: https://scikit-learn.org/stable/ modules/generated/sklearn.ensemble.GradientBoostingRegressor.html (date of access: 25.05.2024).

4. Navicat. Navicat DB Admin Tool for MySQL, Redis, PostgreSQL, MongoDB, MariaDB, SQL Server, Oracle & SQLite client. URL: https://www.navicat.com/en/products (date of access: 25.05.2024).

5. NLTK stem package. NLTK :: Natural Language Toolkit. URL: https://www.nltk.org/api/nltk.stem.html (date of access: 25.05.2024).

6. PchipInterpolator – SciPy v1.13.1. Numpy and Scipy documentation. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.PchipInterpolator.html (date of access: 25.05.2024).

7. *Penn Treebank* P.O.S. Tags. Department of Linguistics - Home | Department of Linguistics. URL: https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html (date of access: 25.05.2024).

8. Prefixes. Cambridge Dictionary | English Dictionary, Translations & Thesaurus. URL: https://dictionary.cambridge.org/grammar/british-grammar/prefixes (date of access: 27.05.2024).

9. RandomForestRegressor. scikit-learn. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html (date of access: 25.05.2024).

10. SpaCy Models Documentation. spaCy. URL: https://spacy.io/models/en#en_core_web_md (date of access: 25.05.2024).

11. SQLite. SQLite Home Page. URL: https://www.sqlite.org/index.html (date of access: 25.05.2024).

12. The CEFR Levels - Common European Framework of Reference for Languages (CEFR). URL: https://www.coe.int/en/web/common-european-framework-reference-languages/level-descriptions (date of access: 25.05.2024).

13. XGBoost Documentation. XGBoost Documentation. URL: https://xgboost.readthedocs.io/en/stable/index.html (date of access: 25.05.2024).