Russian Journal of Linguistics
ISSN 2687-0088 (print), ISSN 2686-8024 (online)

2022 Vol. 26 No. 2 342–370
http://journals.rudn.ru/linguistics

https://doi.org/10.22363/2687-0088-30145

Research article

ReaderBench: Multilevel analysis of Russian text characteristics

Dragos CORLATESCU[1]✉, **Ștefan RUSETI**[1]
and Mihai DASCALU[1,2]

[1]University Politehnica of Bucharest, Bucharest, Romania
[2]Academy of Romanian Scientists, Bucharest, Romania
✉dragos.corlatescu@upb.ro

Abstract
This paper introduces an adaptation of the open source ReaderBench framework that now supports Russian multilevel analyses of text characteristics, while integrating both textual complexity indices and state-of-the-art language models, namely Bidirectional Encoder Representations from Transformers (BERT). The evaluation of the proposed processing pipeline was conducted on a dataset containing Russian texts from two language levels for foreign learners (A – Basic user and B – Independent user). Our experiments showed that the ReaderBench complexity indices are statistically significant in differentiating between the two classes of language level, both from: a) a statistical perspective, where a Kruskal-Wallis analysis was performed and features such as the “nmod” dependency tag or the number of nouns at the sentence level proved the be the most predictive; and b) a neural network perspective, where our model combining textual complexity indices and contextualized embeddings obtained an accuracy of 92.36% in a leave one text out cross-validation, outperforming the BERT baseline. ReaderBench can be employed by designers and developers of educational materials to evaluate and rank materials based on their difficulty, as well as by a larger audience for assessing text complexity in different domains, including law, science, or politics.

Keywords: *ReaderBench framework, text complexity indices, language model, neural architecture, multilevel text analysis, assessing text difficulty*

For citation:
Corlatescu, Dragos, Ștefan Ruseti & Mihai Dascalu. 2022. ReaderBench: Multilevel analysis of Russian text characteristics. *Russian Journal of Linguistics* 26 (2). 342–370. https://doi.org/10.22363/2687-0088-30145

Научная статья

ReaderBench: многоуровневый анализ характеристик текста на русском языке

Драгош КОРЛАТЕСКУ[1]✉, **Штефан РУСЕТИ**[1], **Михай ДАСКАЛУ**[1,2]

[1]*Политехнический университет Бухареста, Бухарест, Румыния*
[2]*Академия румынских ученых, Бухарест, Румыния*
✉dragos.corlatescu@upb.ro

Аннотация

В статье представлена новая версия платформы ReaderBench с открытым исходным кодом. В настоящее время Readerbench поддерживает многоуровневый анализ параметров текстов на русском языке, интегрируя при этом как индексы текстовой сложности, так и современные языковые модели, в частности, BERT. Оценка предлагаемого алгоритма обработки проводилась на корпусе русских текстов двух языковых уровней, используемых при обучении русскому языку как иностранному (А – базовый пользователь и В – независимый пользователь). Наши эксперименты показали, что (а) индексы сложности текстов различных уровней по Общеевропейской шкале, рассчитываемые при помощи ReaderBench, статистически значимы (по критерию Краскела-Уоллиса), при этом количество существительных на уровне предложения оказалось наилучшим предиктором сложности; б) а наша нейронная модель, сочетающая индексы сложности текста и контекстуализированные вложения, при перекрестной валидации достигла точности 92,36 % и превзошла базовый уровень BERT. ReaderBench может использоваться разработчиками учебных материалов для оценки и ранжирования текстов в зависимости от их сложности, а также более широкой аудиторией для оценки сложности восприятия текста в различных областях, включая юриспруденцию, естествознание или политику.

Ключевые слова: *фреймворк ReaderBench, индексы сложности текста, языковая модель, нейронная архитектура, многоуровневый анализ текста, оценка сложности текста*

Для цитирования:

Corlatescu D., Ruseti Ș, Dascalu M. ReaderBench: Multilevel analysis of Russian text characteristics. *Russian Journal of Linguistics.* 2022. Vol. 26. № 2. P. 342–370. https://doi.org/10.22363/2687-0088-30145

1. Introduction

The Natural Language Processing (NLP) field focuses on empowering computers to process and then understand written or spoken language texts in order to perform various tasks. The performance of Artificial Intelligence or Machine Learning approaches on common NLP tasks has increased over the years, but there are still many tasks where computers are far from human performance. Nonetheless, the processing speed of computer programs is not to be neglected, and the current tradeoff between the response time of an algorithm and its errors is shifting the balance towards automated analyses – for example, a human invests tens of hours to correctly extract all the parts of speech from a novel, while the computer can perform the same task in a couple of minutes, with an error of only 1–5% mislabeled words. As such, NLP tools are becoming more widely used to provide valuable inputs to further develop and test various hypotheses.

Tailoring reading materials for learners is a practical and essential field where NLP tools can have a high impact. Designing such materials can prove to be a difficult task since texts below readers' level of understanding will make them lose interest, while texts too difficult to comprehend will demotivate learners. Automated NLP frameworks provide valuable insights in those situations, especially the ones that focus on identifying the complexity of a text. One such tool is the ReaderBench (Dascalu et al. 2013) framework, which previously supported other languages besides English, namely French (Dascalu et al. 2014), Dutch (Dascalu et al. 2017), and Romanian (Gifu et al. 2016), and has now been adapted to also support Russian.

The new version of ReaderBench[1] is a Python library that extracts multilevel textual characteristics from texts in multiple languages. These characteristics, named also textual complexity indices, provide valuable insights of text difficulty on multiple levels, namely surface, word, morphology, syntax, and semantics (i.e., cohesion), all described in the following sections. The purpose of this study is to present the adaptation process of ReaderBench to support the Russian language, starting from the computation of Russian complexity indices, and followed by the integration of new methods for building the Cohesion Network Analysis (CNA, Dascalu et al. 2018) graph using state-of-the-art language models, namely Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. 2019).

Various neural network architectures and statistical analyses were employed to assess the performance of our processing pipeline. Our experiment uses Russian texts from two language level groups that reflect an individual's language proficiency: A (Basic User) and B (Independent User). The corpus is a part of the Russian as a Foreign Language Corpus (RuFoLC) compiled by language experts from the "Text Analysis" laboratory, Kazan Federal University. Our goal is to build an automated model and to perform statistical analyses of the texts in order to differentiate between the two classes, while assessing the importance of the textual complexity indices in making this decision.

2. Assessing Russian text complexity

The manner in which people understand and study languages has changed during the last two centuries; Russian is no different. A brief history of the approaches used to analyze textual complexity in Russian texts is presented by Guryanov et al. (2017). The authors documented that such analyses were conducted by linguists mostly by hand in the beginning of the 20th century. Even though the key terms such as readability or text complexity were not completely defined, the general understanding of the concepts existed and simple indices, such as word

[1] https://github.com/readerbench/ReaderBench

length or number of words, were considered. Moving on to the end of the 20th century – beginning of the 21st century, researchers started to include semantic features, such as, for example: the polysemy of the words. In recent times, additional features were introduced, detailed in the next subsection dedicated to automated measures of text complexity.

One important part of research on text complexity revolves around its educational theme: are texts appropriate in terms of complexity for the students reading or studying them? Linguists can provide an expert opinion to this question; however, this requires a lot of resources, including a considerable amount of time. Thus, a system that can provide meaningful insights into the difficulties encountered when reading a text is desirable.

McCarthy et al. (2019), who are language experts, developed a Russian language test to assess text comprehension. The test was conducted on approximately 200 students (~100 fifth graders, ~100 ninth graders) and the results showed that they struggle to understand the ideas of the texts. Additionally, the paper provided an overview of the entire evaluation process in the Russian educational system, and it offered a viable evaluation alternative designed by linguists in the form of a test.

One of the initial papers on the same matter, but written from a more statistical perspective, was the work by Gabitov et al. (2017). In their study, the problem of text complexity in Russian manuals was addressed. Specifically, the investigation focused on the 8th grade manual on social studies made by Bogolyubov. All analyses were performed mostly manually, starting from selecting 16 texts from the book and then computing readability indices formulas, such as Flesch-Kincaid, Coleman-Liau, Dale-Chale readability formula, Automated Readability Index, and Simple Measure of Gobbledygook (SMOG). The unevenness of those indices across the texts raised questions whether the texts were suitable for students and represented the underlying reason for further research in this domain.

The syntactic complexity of social studies texts was explored by Solovyev et al. (2018). The authors used ETAP-3 (Boguslavsky et al. 2004), a syntactic analyzer for Russian grammar, to compute the dependency parse tree for each sentence. Fourteen indices were extracted based on the dependency tree that looked at key components of the Russian sentence structure in order to deduct its complexity, namely the length of the path between two nodes and various counts of nodes, leaves, verbal participles, verbal adverb phrases, modifiers in a nominal group, syndetic elements, participial constructs, compound sentences, coordinating chains, subtrees, and finite dependent verbs. Their statistical analyses showed high correlation between the extracted features and grade level; however, syntactic features were less correlated than the lexical ones.

Solovyev et al. (2020) also explored how predictive specific quantitative indices were in ranking academic Russian texts and in determining their complexity. Their corpus was composed of texts from the field of Social Studies grouped by grade level, i.e. 5th–11th grades. The texts were extracted from manuals written by two authors (Bogolyubov and Nikitin) used at that time for teaching social studies. The corpus required a preprocessing step, where the parts of speech were extracted using TreeTagger (Schmid et al. 2007) for Russian, the texts were split into sentences, and outliers (i.e., sentences that were even too short or too long) were eliminated. The following indices were used in their analysis: Flesch-Kincaid Grade, Flesch Readability Ease, frequency of content words, average words per sentence, average syllables per word, and additional features based on the part of speech tags (such as the number of nouns or verbs). The authors performed a statistical analysis using both Pearson (1895) and Spearman (1987) coefficients to inspect the correlation between the indices and the complexity of the texts (i.e., their grades level). All features proved to be statistically significant, except for "average words per sentence" and "average syllables per word". Additionally, the authors proposed slightly modified formulas for the Flesch-Kincaid Grade and Flesch Readability Ease that better reflect the field of Social Studies.

Further studies of quantitative indices on the corpus containing texts from Social Sciences manuals, Churunina et al. (2020) introduced new indices such as type-token ratio (TTR), abstractness index, and words frequency based on Sharoff's dictionary (Sharoff et al. 2014) that proved to be statistically significant in differentiating the grades of the texts. Out of the specified indices, abstractness, was proven to be closely related to textual complexity. In fact, the study by Sadoski et al. (2000) claimed that the concreteness (which is the opposite of abstractness) is the most predictive feature for comprehensibility. As a follow-up, Solovyev et al. (2020) provided an in-depth analysis of the abstractness of words in the Russian Academic Corpus (RAC, Solnyshkina et al. 2018) and in a corpus containing students recalls of academic texts. The core of the experiments was the Russian dictionary of concrete/abstract words (RDCA, Akhtiamov 2019). A notable result was obtained in terms of students recall, where texts provided by students used more concrete words than the original ones, underlining the idea that abstract terms are harder to digest.

Quantitative indices provide significant insights into the textual complexity of writings, but they are not the only concept that can be applied to analyze text difficulty. One example can be topic modelling, as applied in an experiment performed by Sakhovskiy et al. (2020) on the Social Studies corpus. The authors implemented Latent Dirichlet Allocation with Additive regularization of topic models (ARTM, Vorontsov & Potapenko 2015). Topics were extracted at three granularity levels: paragraph, segment (i.e., sequences of 1000 words maximum), and full text level. The topics were manually verified by linguist experts, and they were further used in an experiment to determine the correlations between topics and grades of the texts, in four different ways: a) correlation between grade and topic weight, b) correlation between grade and the distance between topic words in a semantic space, c) correlation between grade and topic coherence, and d) correlation between topic properties and complexity-based topic proportion growth. The conclusion of their study highlighted that topic models can be successfully used to assess text complexity.

3. Textual complexity as a Natural Language Processing task

Readability reflects the level of easiness in understanding of a text. Extracting features using NLP techniques is a common approach, when exploring the readability of a given text. There are multiple tools readily available; however, most of them support only English. Nevertheless, the underlying ideas can be extrapolated to other languages, as well. We further describe recent tools that cover the most frequently integrated textual complexity indices and that are also present to some extent in the Russian version of ReaderBench.

One of the first freely available systems is Coh-Metrix (Graesser et al. 2004) which is at its 3rd version at present. Coh-Metrix provides 108 textual complexity indices from eleven categories: descriptive, text easability principal components scores, referential cohesion, LSA, lexical diversity, connectives, situation model, syntactic complexity, syntactic pattern density, word information, and readability. The framework can be freely accessed on a website, but the code is not open-sourced. Coh-Metrix offers support for other languages than English, namely Traditional Chinese, while adaptations for other languages exist – for example, Coh-Metrix-Esp (Quispesaravia et al. 2016) for Spanish.

The Automatic Readability Tool for English (ARTE, Choi 2020) is a Java library available on all platforms that processes plain text files and outputs a CSV file with all the computed indices. The list of indices includes the Flesch Reading Ease Formula (Flesch 1949), Flesch Kincaid Grade Level Formula (Kincaid et al. 1975), and Automated Readability Index (Senter & Smith 1967), which take into consideration the average number of words per sentence, the average number of syllables per word, and the difference between them consisting of the weights for each parameter. Other examples of indices are SMOG Grading (Mc Laughlin 1969) and the New Dale-Chall Readability Formula (Dale & Chall 1948). Lastly, there are multiple "crowdsourced" indices that are computed by aggregation of different counts and statistics from other libraries.

The following library is the Constructed Response Analysis Tool (CRAT, Crossley et al. 2016) which provides over 700 indices that also take into consideration text cohesion. The indices are grouped in specific categories, namely: a) indices that count or compute percentages for words, sentences, paragraphs, content words, function words, and parts of speech, or b) indices based on the MRC Psycholinguistic Database (Coltheart 1981), the Kuperman Age of Acquisition scores (Kuperman et al. 2012), the Brysbaert Concreteness scores (Brysbaert et al. 2014), the SUBTLEXus corpus (Brysbaert et al. 2012), the British National Corpus (BNC, BNC Consortium 2007), the COCA corpus (Davies 2010). Complementary, the Custom List Analyzer (CLA, Kyle et al. 2015) is a library written in Python that computes various occurrences of text sequences (i.e., a word, an n-gram, or a wildcard) in a corpus.

The Grammar and Mechanics Error Tool (GAMET, Crossley et al. 2019) is a Java library that identifies errors in a plain text file from the perspective of grammar, spelling, punctuation, white space, and repetitions. The core of the library integrates two packages, one from Java, Java LanguageTool (LanguageTool 2021), and one from Python, language-check (Myint 2014). The GAMET project was also tested and evaluated on two datasets (Crossley et al. 2019): a) a TOEFL-iBT corpus containing 480 essays written by English as a Second Language Learners, and b) 100 essays written by high school students in the Writing Pal Intelligent Tutoring System project (Roscoe et al. 2014). The errors reported by GAMET were evaluated by two expert raters, and the results showed that GAMET offered relevant feedback throughout the experiments.

Next, we explore a collection of four tools (TAACO, TAALEED, TAALES and TAASC) that cover a wide spectrum of analysis levels. All the tools have a graphical interface that accepts plain text files as input to produce CSV files with all indices as outputs. First, the Tool for the Automatic Analysis of Cohesion is a framework that focuses on text cohesion. The indices are separated into multiple categories: a) TTR and Density, where TTR stands for type-token ratio computed as the number of unique words/lemmas in a category, divided by the total number of words/lemma in the same category; b) Sentence overlap, where statistics regarding the repetition of the same word with certain properties in the following sentences are computed; c) Paragraph overlap, which is similar to the sentence overlap, only that the metrics are computed at paragraph level; d) Semantic overlap, where the scores of similarity between adjacent blocks (sentences and paragraphs) are computed on three methods: Latent semantic analysis (Landauer et al. 1998), Latent Dirichlet allocation (LDA, Blei et al. 2003), and word2vec (Mikolov et al. 2013); e) Connectives, where statistics are computed based on the types of the English connectives (e.g. conjunctions, disjunctions); f) Givenness, which is a measure of new information in the context of previous information, based on pronouns counts and repeated content lemmas. Second, the Tool for the Automatic Analysis of Lexical Diversity (TAALED, Kyle et al. 2021) provides 9 indices for measuring the language diversity of a text.

Third, the Tool for the Automatic Analysis of Lexical Sophistication (TAALES, Kyle et al. 2018) offers 484 indices addressing lexical sophistication divided into 4 major categories: a) Academic Language containing wordlists and formulas based on counts and percentages of words, b) indices based on the COCA corpus, c) indices based on other corpora (BNC, MRC, SUBTLEXus), and d) other types of indices, such as Age of Exposure or Contextual Distinctiveness. Fourth, the Tool for the Automatic Analysis of Syntactic Sophistication and Complexity (TAASSC, Kyle 2016) focuses on analyzing the sentence components and the relations between them. It provides statistics at the clause and noun phrase level for measuring complexity. The syntactic sophistication is computed based on indices that focus on verbs and lemmas.

Textstat (Bansal 2014) is a Python library available online on the pypi archives which provides textual complexity indices for multiple languages. Textstat includes 16 indices, out of which most are English readability formulas: Flesch Reading Ease, Flesch Kincaid grade, SMOG, Coleman Liau, Automated Readability, and Dale Chall.

ReaderBench (Dascalu et al. 2013) is an open-source framework that offers multiple natural language processing tools. ReaderBench was initially developed in Java, but the library migrated to Python given that all major NLP frameworks, including Tensorflow (Abadi et al. 2016), Scikit learn (Pedregosa et al. 2011), spaCy (Honnibal & Montani 2017), and Gensim (Rehurek & Sojka 2010) are written in Python to enable Graphics Processing Unit (GPU) optimizations. ReaderBench is grounded in Cohesion Network Analysis (CNA, Dascalu et al. 2018), a method similar to Social Network Analysis, but instead of representing relations between people or entities, the CNA graph contains links between text elements. The weights of the links are given by the semantic similarity between the components using different semantic models, such as LSA, LDA, or word2vec. Both local and global cohesion are computed based on the strength of intra and inter-paragraph edges extracted from the CNA graph. The library comes with a demo website (Gutu-Robu et al. 2018), making it available to multiple audiences. On one hand, the Python library can be installed and used by machine learning/NLP developers using the Pip library archives[2]). On the other hand, the website provides multiple interactive interfaces, where linguists or any other person interested in studying text can perform their own analysis using the capabilities of the library, without having any programming knowledge – demos include for example: Multi document CNA (i.e., a detailed analysis and visualization of multiple documents grounded in Cohesion Network Analysis), Keywords extraction (i.e., a list and a graph of the keywords from a text), AMoC (Automated Model of Comprehension, a model that simulates reading comprehension), Sentiment Analysis (i.e., extracting the polarity of a text in terms of expressed sentiments), and Textual Complexity (i.e., provide an export of the complexity indices applied on the input text). All publicly available analyses cover multiple languages, not just English, and all the additional information required for each experiment is also present on the website. In this study, we focus on the extension of the framework to also accommodate textual complexity indices and prediction models for Russian texts.

It is important to note that ReaderBench provides a viable alternative to all other previously mentioned software for text analysis. ReaderBench leverages state of the art NLP models to explore the semantics of texts and was effectively employed in various comprehension tasks, in multiple languages including English, French, Dutch, and Romanian. The project is open sourced under an Apache 2 license, the library can be easily integrated into multiple Python projects, whereas the presentation website can be used freely for the remote processing of texts.

4. Current study objectives

Our study focuses on an in-depth multilevel analysis of Russian texts by employing textual complexity indices and the CNA graph updated with language models, together with neural network models and statistical analyses, all integrated

[2] https://pypi.org/project/rbpy-rb/

into the ReaderBench framework. As such, we assess to what extent the Russian textual complexity indices integrintoed into ReaderBench are predictive of the differences between Russian texts from two language levels (i.e., A – Basic User and B – Independent User). We perform this analysis to explore the predictive power of our models and underline the most predictive features for this task.

5. Method

Corpus

This study considers Russian texts from two language levels for foreign learners (A-Basic User and B-Independent User) with the aim to predict a text's difficulty class. The selection of texts in terms of complexity assessment was performed by Russian linguists, members of the "Text Analytics" Laboratory from the Kazan Federal University. The corpus used in the follow-up experiments is a subpart of the Russian as a Foreign Language Corpus (RuFoLC). The initial corpus was in a raw format containing texts from 3 language levels A1 (Breakthrough or beginner), A2 (Waystage or elementary), and B1 (Threshold or intermediate). However, since only 3 texts were available for the A1 level, we decided to merge the A1 and A2 together (see Table 1 for corpus statistics). Since the overarching number of examples was too low for a neural network to learn meaningful representations, we decided to use paragraphs as input in order to ensure an increased number of samples.

Table 1. **Language levels corpus statistics.**

Class	# Documents	# Paragraphs	# Sentences	# Words
A	37	465	1663	18,307
B	48	333	1105	13,741

The ReaderBench Framework adapted for Russian

A specific set of resources is required for a new language to be integrated into ReaderBench. From this list, part are mandatory, while others are nice to have. One mandatory requirement is to have a language model available in spaCy (Honnibal & Montani 2017), an open-source library written in Python that offers support for NLP pre-processing tasks, such as part of speech tagging, dependency parsing, and named entity recognition. SpaCy offers a unified pipeline structure for any language and, at the moment of writing, spaCy reached version 3.1 with support for 18 languages, including Russian which has been integrated for reproducibility reasons. Additionally, spaCy includes a multi-language model that can be used for any language, but with lower performance. All languages have multiple models (i.e., small, medium, and large) available to address memory or time constraints. Smaller models are faster to run and require fewer resources, but yield lower performance.

Semantic models are a key component for the ReaderBench pipeline and for building the CNA graph. All indices that are calculated based on the meaning of the words, the relations between words, sentences and paragraphs need a semantic language model. ReaderBench generally uses word2vec as a language model because it is available for most languages from multiple sources. During the development of this paper, we also considered it fit to align the semantic models across all the languages available in ReaderBench. Thus, we added support for the MUSE (Conneau et al. 2018) version of word2vec, where the semantic spaces are similar across the three languages.

Previous versions of ReaderBench used to compute similarity scores between textual elements from the CNA graph using LSA, LDA, and word2vec; however, these models have been outperformed by BERT-based (Devlin et al. 2019) derivatives. The Transformer architecture introduced by Vaswani et al. (2017) obtained state of the art results in most NLP tasks, especially with its encoder component, namely the Bidirectional Encoder Representations from Transformers (BERT). The original BERT was trained on two tasks: language modeling (where 15% of the tokens were masked and the model tried to predict the best word that fitted the mask, given the context) and next sentence prediction (given a pair of sentences, the model tried to predict if the second sentence made sense to follow the first sentence). The language modeling component is used to represent words in a latent vector space.

Nowadays, almost all languages have a custom BERT model available, and Russian is no exception. The ReaderBench library now integrates the DeepPavlov rubert-base-cased (Kuratov & Arkhipov 2019) BERT-base model to compute contextualized embeddings. It is important to note that this is the first study in which ReaderBench indices are computed using BERT-based embeddings.

Besides the above-mentioned libraries and models, ReaderBench can also benefit from specific word lists which were adapted for Russian, including: list of stop words (i.e., words with no semantic meaning ignored in preprocessing stages), list of connectives and discourse markers, and list of pronouns grouped by type and person; all previous word lists were provided by Russian linguists.

Additional improvements were made to the ReaderBench Python codebase, including performance optimizations and a refactoring to provide a more efficient and cleaner implementation of the textual complexity indices. New cohesion-centered textual complexity indices were added in ReaderBench, as well as a new aggregation function on top of them – the maximum value at a certain granularity level (more details are presented in the next section).

Textual Complexity Indices for Russian

The textual complexity indices provided by ReaderBench ensure a multilevel analysis of text characteristics and are grouped by their scope (see dedicated Wiki page[3]). Table 2–6 present the names of the indices, their description, what component or components from the above enumeration are used, as well as availability in terms of granularity. Note that, as previously mentioned, all indices

[3] https://github.com/readerbench/ReaderBench/wiki/Textual-Complexity-Indices

require the spaCy pre-processing pipeline to be executed; thus, SpaCy does not appear as a dependency. The "Granularity" column reflects four possible levels on which the index is calculated: Document (D), Paragraph (P), Sentence (S), or Word (W). In general, the value of one level of granularity is computed recursively as a function of values coming from one level below. For example, word counts are calculated at the sentence level by considering word occurrences from each sentence; follow-up at paragraph level, we report the count of the words from all sentences belonging to a targeted paragraph. The final values presented as indices are the results of three aggregation functions: mean (abbreviated "M"), standard deviation (abbreviated "SD"), and maximum (abbreviated "Max"). Thus, an index can look like "M(Wd / Sent)", which can be translated as the mean value of words per sentence in a text. In terms of consistency across languages, all ReaderBench indices, their acronyms and descriptions, are provided in English.

The surface indices available in ReaderBench are presented in Table 2. These indices are computed using simple algorithms that involve counting appearances of words, punctuations, and sentences. Starting from the Shannon's Information Theory (Shannon 1948), the idea of entropy at word level is also included as an index; the hypothesis is that a more varied vocabulary (i.e., higher entropy) may result in a more difficult text to understand.

Table 2. **ReaderBench Surface indices**

Abbreviation	**Description**	**Dependencies**	**Granularity**			
			D	**P**	**S**	**W**
Wd	Words	-	X	X	X	
UnqWd	Unique words	-	X	X	X	
Comma	Commas	-	X	X	X	
Punct	Punctuation marks (including commas)	-	X	X	X	
Sent	Sentences	-	X	X		
WdEnt	Word Entropy	-	X	X	X	

The morphology category (see Table 3) contains indices computed using the part of speech tagger from spaCy. Statistics are computed for each part of speech (e.g., nouns, verbs), while more detailed statistics are considered for sub-types of pronouns provided by linguists as predefined lists.

Table 3. **ReaderBench Morphology indices**

Abbreviation	**Description**	**Dependencies**	**Granularity**			
			D	**P**	**S**	**W**
PosMain	Words with a specific POS	-	X	X	X	
UnqPosMain	Unique words with a specific POS	-	X	X	X	
Pron	Specific pronoun types	Pronoun lists	X	X	X	

From the syntax point of view (see Table 4), ReaderBench provides indices derived from the dependency parsing tree. An index is computed for each dependency type available in the spaCy parser, such as "nsubj" or "cc". The depth of the parsing tree is also an important feature in quantifying textual complexity: if the depth is high, then the text may become harder to understand.

Table 4. **ReaderBench Syntax indices**

Abbreviation	Description	Dependencies	Granularity			
			D	P	S	W
Dep	Dependencies of specific type	-	X	X	X	
ParseTreeDpth	Depth of the parsing tree	-			X	

Table 5 presents the indices that take into consideration text cohesion derived from the CNA graph. Cohesion is an important component when assigning text difficulty, as a lack of cohesion or cohesion gaps can make a text harder to follow (Dascalu 2014). As expected, a semantic model is required, either word2vec or the newly introduced BERT-base models. Note that the indices AdjSentCoh, AdjParCoh, IntraParCoh and InterParCoh were newly added to ReaderBench for this research.

Table 5. **ReaderBench Cohesion indices**

Abbreviation	Description	Dependencies	Granularity			
			D	P	S	W
AdjSentCoh	Cohesion between two adjacent sentences	Semantic Model	X	X		
AdjParCoh	Cohesion between two adjacent paragraphs	Semantic Model	X			
IntraParCoh	Cohesion between sentences contained within a given paragraph	Semantic Model	X	X		
InterParCoh	Cohesion between paragraphs	Semantic Model	X			
StartEndCoh	Cohesion between first and last text element	Semantic Model	X	X		
StartMiddleCoh	Cohesion between start and all middle text elements	Semantic Model	X	X		
MiddleEndCoh	Cohesion between all middle and last elements	Semantic Model	X	X		
TransCoh	Cohesion between the last sentence of the current paragraph and the first sentence from the upcoming paragraph	Semantic Model	X			

ReaderBench also provides statistics at individual words level (see Table 6). Name entity features are computed based on the Named Entity Recognizer from spaCy, while specific tags depend on the corpus on which the NER model was trained. For example, the Russian model is trained on a Wikipedia corpus and offers only 3 tags: location (“LOC”), organization (“ORG”), person (“PER”), while other models such as the English one offer 18 categories. This may affect the global statistics when comparing the complexity of texts from two languages, as observed in follow-up experiments. The syllables are computed using the “Pyphen” library for each language (Kozea 2016).

For other languages besides Russian, ReaderBench also includes additional textual complexity indices. For example, none of the Wordnet indices (e.g., sense counts, depths in hypernym trees) are currently available as the Russian WordNet (Loukachevitch et al. 2016) is in a different format when compared to the models integrated in Natural Language Toolkit (NLTK). Additionally, specific word lists like Age of Acquisition, Age of Exposure, and discourse connectors are not yet available for Russian; as such, their corresponding indices are not computed.

Table 6. **ReaderBench Word indices**

Abbreviation	Description	Dependencies	Granularity			
			D	P	S	W
WdLen	Number of characters in a word	-				X
WdDiffLemma	Distance in characters between word (inflected form) and its corresponding lemma	-				X
Repetition	Number of occurrences of the same lemma	-	X	X	X	
NmdEnt	Number of specific types of named entity	Named Entity Recognizer	X	X	X	
Syllab	Number of syllables in a word	Rules or Dictionary				X

6. Neural Network Architectures combining Textual Complexity Indices and Language Models

Our first approach for predicting text difficulty involved using ReaderBench to extract the complexity indices available for the Russian language that were further introduced into a neural network depicted in Figure 1.a. The architecture started with an input layer which received the complexity indices for each text as a list. An optional layer with 128 units and Rectified Linear Unit (“RELU”) activation function can be added to increase the complexity of the function computed by the neural network. Next, a dense layer with 32 units and with “RELU” as the activation function is used as a hidden layer. Finally, the output layer is a dense layer with only one output and the activation function ‘sigmoid’, which provides the class of the text.

Second, BERT and its derived models hold state-of-the-art results in multiple text classification tasks. Thus, we decided to test an architecture that uses only RuBERT, a BERT-base model trained for the Russian language. We obtained a semantic representation for each text by computing the mean of the last hidden state from the RuBERT output. Then, the embedding was feed into a neural network with an architecture similar to the previous one (see Figure 1.b).

Third, we tested a combination of the two inputs, as the RuBERT embeddings were concatenated with the ReaderBench indices and fed as input into the neural network. The architecture of the neural network can be observed in Figure 1.c

7. Statistical Analyses

A statistical approach was adopted to determine which features were significant in differentiating between textual complexity classes. The Shapiro normality test (Shapiro & Wilk 1965), as well as the skewness and kurtosis tests (Hopkins & Weeks 1990), were used to filter ReaderBench indices in terms of normality. Since most indices were not normally distributed, the Kruskal-Wallis analysis of variance (Kruskal & Wallis 1952) was employed to determine the statistical importance of the indices.

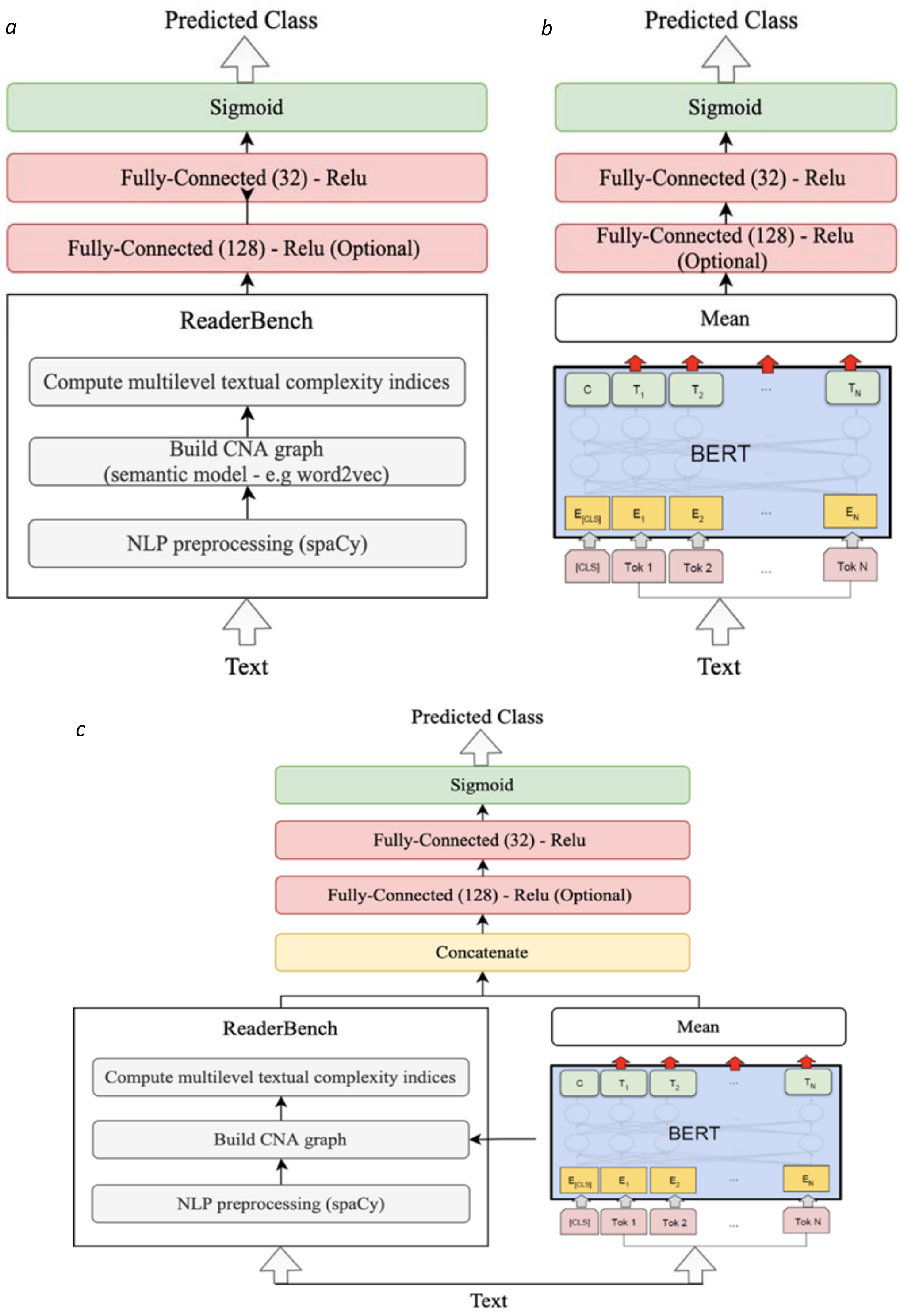
a
Predicted Class
Sigmoid
Fully-Connected (32) - Relu
Fully-Connected (128) - Relu (Optional)
ReaderBench
Compute multilevel textual complexity indices
Build CNA graph
(semantic model - e.g word2vec)
NLP preprocessing (spaCy)
Text
b
Predicted Class
Sigmoid
Fully-Connected (32) - Relu
Fully-Connected (128) - Relu
(Optional)
Mean
C
T1
T2
...
TN
BERT
E[CLS]
E1
E2
EN
[CLS]
Tok 1
Tok 2
Tok N
Text
c
Predicted Class
Sigmoid
Fully-Connected (32) - Relu
Fully-Connected (128) - Relu (Optional)
Concatenate
ReaderBench
Compute multilevel textual complexity indices
Build CNA graph
NLP preprocessing (spaCy)
Mean
BERT
[CLS]
Tok 1
Tok 2
Tok N
Text

Figure 1. Neural network architectures:
a) Neural Network with ReaderBench indices as input; b) Neural Network with RuBERT embeddings as input; C) Neural network with both ReaderBench indices and RuBERT embeddings as input

8. Experimental Setup

The process of training neural networks requires the setup of hyperparameters. Thus, the Adam optimizer was considered with a learning rate of 1e-3. The loss function was binary cross-entropy, given that only two classes were predicted. Finally, each model was trained for 64 epochs with a batch size of 16.

The neural network architectures were used to classify the Russian texts on the two language levels: A (Basic user) and B (Independent user). The paragraphs were extracted from each text and labeled as the category the source text belonged to. We decided to perform cross-validation to evaluate the models due to the limited number of examples. There are multiple ways in which cross-validation can be performed, the most common ones being the 5-fold or 10-fold cross-validations. However, employing those methods involves limiting even more the input of the neural network, which in turn requires a substantial amount of data to be trained. Thus, given the limited number of entries, we elected to use a "leave-one-out" approach, where the entire corpus except a single entry is used for training a model at one iteration, followed by evaluation on the remaining entry. The process is repeated for each entry until the corpus is exhausted and performance is computed as the mean of all evaluation scores. Our corpus was composed of paragraphs and leaving one out would have meant that the other paragraphs from the same text would have been used in the training process which, again, could have generated biased. Thus, we decided to employ the "leave one text out" cross-validation. In this approach, an entire text (i.e., all the paragraphs belonging to the selected text) was left out, while the models were trained on all the other paragraphs. The final accuracy was reported as the mean of the results for each text.

9. Results

Table 7 depicts the results for the three neural architectures. The complexity indices from ReaderBench, as well as the RuBERT embeddings, were used as input to two different architectures: the first with only one hidden layer of 32 units, and the second with two hidden layers of 128 and 32 units. The scenario where the two input sources were combined is also presented.

Table 7. **Neural networks results**

Model Input Features	Hidden Layers	Leave one text out cross-validation (%)
Complexity Indices	1 hidden layer – 32 units	90.58
RuBERT	1 hidden layer – 32 units	87.49
Complexity Indices	2 hidden layers – 128, 32 units	87.05
RuBERT	2 hidden layers – 128, 32 units	88.69
Complexity Indices + RuBERT	1 hidden layer – 32 units	88.23
Complexity Indices + RuBERT	2 hidden layers – 128, 32 units	**92.36**

Table 8 presents a summary of the results obtained by applying Kruskal-Wallis test. The indices are divided by categories and subcategories, and each slot introduces specific indices that are either statistically significant in differentiating the texts from the classes A and B or not. The notation is condensed and indices with the same characteristics are grouped using the "|" character. For example, the first entry considers the category "Surface" and subcategory word ("Wd"); the notation "M|Max(Wd / Doc|Par|Sent)" can be expanded to all the possibilities where "|" appears: M(Wd / Doc), M(Wd / Par), M(Wd / Sent), Max(Wd / Doc), Max(Wd / Par), Max(Wd / Sent). Additionally, in the "Dep" subcategory there is a list of dependency types that fitted the same pattern, and they are represented in a mathematical manner as a set. An important observation is that all features at document granularity were disregarded in this analysis, given the structure our data – i.e., all documents in the dataset contain only 1 paragraph; as such, the indices for the two granularities were the same. Similarly, the maximum values at paragraph level were ignored since the maximum and the mean values of only one entry are the same. An extended table with the descriptive statistics and corresponding χ2 and *p* values for all statistically significant textual complexity indices is provided in Appendix 1.

Table 8. **Summary of the predictive power of textual complexity indices.**

Indices Category	**Indices Subcategory**	**Significant Indices (*p* < .05)**	**Not Significant Indices (*p* > .05)**
Surface	Wd	M|Max(Wd / Sent), M(Wd / Par), SD(Wd / Sent)	SD(Wd / Par)
	UnqWd	M|Max(UnqWd / Sent), M(UnqWd / Par), SD(UnqWd / Sent)	SD(UnqWd / Par)
	Comma	M|Max(Commas / Sent), M(Commas / Par), SD(Commas / Sent)	SD(Commas / Par)
	Punct	M(Punct / Par), SD(Punct / Sent)	M|Max(Punct / Sent), SD(Punct / Par)
	Sent	M(Sent / Par)	SD(Sent / Par)
	WdEntr	M|Max|SD(WdEntr / Sent), M|SD(WdEntr / Par)	-
	NgramEntr	M|Max|SD(NgramEntr_2 / Word)	-
Morphology	POS	M|Max(POS_noun|_adj|_adv / Sent), M(POS_noun|_adj|_adv / Par), SD(POS_noun|_adj|_adv / Sent)	SD(POS_noun|_adj|adv / Par)
		SD(POS_pron / Sent)	M|Max(POS_noun / Sent), M(POS_noun / Par), SD(POS_noun / Par)
		M(POS_verb / Par), SD(POS_verb / Sent)	M|Max(POS_verb / Sent), SD(POS_verb / Par)
	UnqPOS	M|Max(UnqPOS_noun|_adj|_adv / Sent), M(UnqPOS_noun|_adj|_adv / Par), SD(UnqPOS_noun|_adj / Sent)	SD(UnqPOS_noun|_adj|_adv / Par)
		SD(UnqPOS_pron / Sent)	M|Max(UnqPOS_noun / Sent), M|SD(UnqPOS_noun / Par)
		M(UnqPOS_verb / Par), SD(UnqPOS_verb / Sent)	M|Max(UnqPOS_verb / Sent), SD(UnqPOS_verb / Par)

Indices Category	Indices Subcategory	Significant Indices (*p* < .05)	Not Significant Indices (*p* > .05)
	Pron	M|Max(Pron_indef / Sent), M(Pron_indef / Par), SD(Pron_indef / Sent)	SD(Pron_indef / Par)
		-	M|Max|SD(Pron_fst|Pron_thrd / Sent), M|SD(Pron_fst|Pron_thrd / Par)
		SD(Pron_snd / Sent)	M|Max(Pron_snd / Sent), M(Pron_snd / Par), SD(Pron_snd / Par)
Syntax	Dep	M|Max(Dep_**X** / Sent), M(Dep_**X** / Par), SD(Dep_**X** / Sent)	SD(Dep_**X** / Par)
		X ∈ {nmod, amod, case, acl, obl, det, xcomp, nummod, conj, appos, mark, cc, objt}	
		M(Dep_nsubj / Par), SD(Dep_nsubj / Sent)	M|Max(Dep_nsubj / Sent), SD(Dep_nsubj / Par)
		* All other types of dependencies were not significant	
	ParseDepth	M|Max(ParseDepth / Sent), M(ParseDepth / Par), SD(ParseDepth / Sent)	SD(ParseDepth / Par)
Cohesion	AdjCoh	M|Max(AdjCoh / Par)	SD(AdjCoh / Par)
	IntraCoh	M|Max(IntraCoh / Par)	SD(IntraCoh / Par)
	* StartEndCoh, StartMidCoh, MidEndCoh, TransCoh – Not Relevant for this analysis		
Word	Chars	M|Max|SD(Chars / Sent|Word), M|SD(Chars / Par)	-
	LemmaDiff	Max|SD(LemmaDiff / Word)	Max|M|SD(LemmaDiff / Sent), M(LemmaDif / Word), M|SD(LemmaDiff / Par)
	Repetitions	M|Max|SD(Repetitions / Sent), M|SD(Repetitions / Par)	-
	NmdEnt	M|Max(NmdEnt_loc|_org / Sent|Word), SD(NmdEnt_loc|_org / Sent|Word), M(NmdEnt_loc|_org / Par),	SD(NmdEnt_loc|_org / Par), *All for NmdEnt_per
	Syllab	M|Max(Syllab / Sent|Word), M(Syllab / Par), SD(Syllab / Sent|Word)	SD(Syllab / Par)

* mean (abbreviated "M"), standard deviation (abbreviated "SD"), and maximum (abbreviated "Max") are the aggregation functions applied at various granularities.

Two methods were employed to determine the efficiency of textual indices from ReaderBench in differentiating texts from two language levels (i.e., A versus B): neural networks and statistical analyses. In the first approach, the ReaderBench features performed better than RuBERT embeddings (see Table 7). Nonetheless, the neural networks that used only the RuBERT embeddings as input performed well (i.e., accuracy of 88.69%), even though the BERT embeddings are recognized for their capabilities to model the meaning of a text. Note that this result does not imply that ReaderBench indices are better than BERT on text classification tasks in general, but rather argue that ReaderBench textual complexity indices can be successfully employed to assess text difficulty.

Both inputs, ReaderBench textual complexity indices and RuBERT embeddings, were used in different versions of initial neural network. The results from Table 7 indicate that adding an extra hidden layer for the neural network with only textual complexity indices decreased performance, thus arguing that the function that maps the inputs to the predicted class should be a simple one. In contrast, the BERT embeddings benefitted from the additional layer, therefore arguing that the mapping between the encodings and the complexity of a text is more complex than in the previous case. In the third configuration the two input sources were combined and tested on the same task; this architecture achieved the highest score (92.36%) with two hidden layers, benefiting from both handcrafted features and BERT contextualized embeddings. The intuition behind the performance increase is that the two approaches complement each other.

The statistical analysis using Kruskal-Wallis statistical test showed that the majority of indices were significant in differentiating between the two classes. In general, the indices aggregated with the standard deviation function were not so statistically significant, while the mean and the maximum related indices proved to be more predictive. While considering Appendix 1, the "nmod" dependency category was the most influential one, ranking first in the Kruskal-Wallis $\chi 2(1)$ score with the index *Max(Dep_nmod / Sent) (*$\chi 2 = 84.48$, $p < .001$), as well as having 6 appearances in top 10 most influential features. The nominal modifier appeared more frequently in more complex texts (B) than in the less complex texts (A). In the same syntactic category, the "amod" dependency also exhibited similar patterns.

In terms of morphology, the number of nouns was higher in B texts than in A text, both as rough count and unique count. The mean value of nouns at sentence level was ranked 2nd in terms of effect size (M(POS_noun / Sent); $\chi 2 = 84.31$, $p < .001$), while other 3 related indices made it to top 10 most predictive features. The number of adjectives was also statistically significant, with the most predictive index in this subcategory (i.e., M(POS_adj / Par); $\chi 2 = 69.28$, $p < .001$) ranking in top 5% of all the indices.

From the Word category, character indices performed best in terms of separating the two types of texts (e.g., M(Chars/Word); $\chi 2 = 76.03$, $p < .001$), all the three variations being close to each other in the ranking. This finding supports the intuition that easier texts generally have shorter words in their composition. Strongly related to this subcategory is the syllables subcategory that also had an important impact (e.g., M(Syllab / Word); $\chi 2 = 73.08$, $p < .001$).

From the remaining two categories, Surface and Cohesion, the highest impact was obtained by the features regarding the number of unique words (e.g., M(UnqWd / Par); $\chi 2 = 32.74$, $p < .001$) and, respectively, the middle end cohesion feature (e.g., M(MidEndCoh / Par); $\chi 2 = 25.89$, $p < .001$). As it can be seen from Table 8, these features were still statistically significant in differentiating the two categories of texts, but they are in the middle of overall rankings in terms of predictive power (i.e., ranks between places 70 and 100).

10. Discussion

Our findings indicate that the ReaderBench textual complexity indices, which span across multiple levels of analysis, provide valuable insights into the differences between two language levels for foreign Russian learners (A-Basic User and B-Independent User). From a machine learning perspective, the results are interesting, as a simple neural network using the features extracted with ReaderBench outperformed the Russian version of BERT, namely RuBERT, in the task of text classification. Nonetheless, this result likely occurred given that the complexity indices were specifically fitted for this task. In addition, we observed that the combination of features from both methods improved the overall classification scores. As such, the methods complement one another and the texts from the two categories differ from each other in terms of both textual complexity features and underlying themes (represented by meaning).

A follow-up analysis was centered on the textual complexity features; as such, the Kruskal-Wallis test was used to identify the most predictive indices, individually and per category. From the syntactic point of view, we can observe that the two most impactful features were “nmod” and “amod”. The nominal modifier (i.e., “nmod”) consists of a noun or a noun phrase that is expressed in Russian using genitive, while showing the possessiveness of another noun; “amod” is similar, with the difference that the syntactical formation is an adjectival phrase. Thus, both “amod” and “nmod” modify the meaning of a noun. In other words, adding more information to the nouns seems to make texts more difficult to comprehend.

From the surface category, the most significant feature is the number of unique words. Although this feature is not that impactful, it suggests that B texts tend to be longer than A texts. Nonetheless, it is more interesting to emphasize the underlying reason: from the morphological category, the number of nouns and of adjectives influence most the differences between the two types of texts; thus, additional concepts (i.e., nouns) are introduced, with corresponding descriptions (i.e., adjectives). In contrast, the number of verbs indicative of actions has a lower $\chi 2$ value in comparison to the previously mentioned parts of speech. Thus, texts that are ranked as being more difficult include more descriptive passages rather than action centered.

When considering semantics, text cohesion does not differ that much in comparison to the other categories, although is statistically significant at in-between sentences from the input paragraph. Yet again, this was an expected result, given that text cohesion is a measure of how well ideas relate to one another and flow throughout the text; nevertheless, texts are well written by experts and should be cohesive.

Our statistical analysis pinpointed that the difficulty of Russian texts comes from the usage of more descriptive passages that include phrases rich in nouns and adjectives. Other characteristics, such as the number of (unique) words, are logical implications of the previous idea. Given that the considered corpus was developed by language experts and can be considered of reference for the Russian educational system, our findings can further support the design of new materials for L2 education. In addition, ReaderBench can be used in other experiments or domains where textual complexity is an important factor, as it can be used to quantify the differences between B and C language level texts, between manuals from two different grade levels, or to estimate the difficulty of science, politics, or law texts.

11. Conclusions and future work

This paper introduced the adaptation of the open-source ReaderBench framework to support multilevel analyses of Russian language in terms of identifying text characteristics reflective of its difficulty. Numerous improvements were made, starting from code refactoring, the addition of new indices (e.g., adjacent cohesion for sentences and for paragraphs, inter-paragraph cohesion) and of the maximum aggregation function, the integration of BERT language model as input for building the CNA graph, as well as the usage of the MUSE version of word2vec that provides multilingual word embeddings.

The ReaderBench textual complexity indices together with BERT contextualized embeddings were used as inputs to predict the language level of texts from two classes: A (Basic User) and B (Independent User). Both approaches, namely neural network architectures and the statistical analyses using the Kruskal-Wallis test, confirmed that the complexity indices from ReaderBench are reliable predictors for text difficulty. The best performance of the neural network using both handcrafted features and BERT embeddings achieved a 92.36% leave one text out cross-validation, thus arguing for the model's capability to distinguish between text of various difficulties.

ReaderBench can be used to assess the complexity of Russian texts in different domains, including law, science, or politics. In addition, our framework can be employed by designers and developers of educational materials to evaluate and rank learning materials.

In terms of future work, we want to further extend the list of Russian textual complexity indices available in ReaderBench, including discourse markers and the Russian WordNet which currently is not aligned with the Open Multilingual Wordnet format. In addition, we envision performing additional studies regarding the complexity of the Russian texts and focusing on textbooks used in the Russian educational system, as well as multilingual analyses highlighting language specificities.

Acknowledgements

This research was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS – UEFISCDI, project number TE 70 *PN-III-P1-1.1-TE-2019-2209,* ATES – "Automated Text Evaluation and Simplification".

REFERENCES

Abadi, Martin. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)* Savannah, GA, USA: {USENIX} Association. 265–283.

Akhtiamov, Raouf B. 2019. Dictionary of abstract and concrete words of the Russian language: A methodology for creation and application. *Journal of Research in Applied Linguistics*. Saint Petersburg, Russia: Springer. 218–230.

Bansal, S. 2014. Textstat. Retrieved September 1st, 2021. URL: https://github.com/shivam5992/textstat (accessed 26.05.2022).

Blei, David M., Andrew Y. Ng & Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3(4–5). 993–1022.

BNC Consortium. 2007. British national corpus. *Oxford Text Archive Core Collection*.

Boguslavsky, Igor, Leonid Iomdin & Victor Sizov. 2004. Multilinguality in ETAP-3: Reuse of lexical resources. In *Proceedings of the Workshop on Multilingual Linguistic Resources*. Geneva, Switzerland: COLING. 1–8.

Brysbaert, Marc, Boris New & Emmanuel Keuleers. 2012. Adding part-of-speech information to the SUBTLEX-US word frequencies. *Behavior Research Methods* 44(4). 991–997.

Brysbaert, Marc, Amy Beth Warriner & Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods* 46(3). 904–911.

Choi, Joon Suh & Scott A. Crossley. 2020. ARTE: Automatic Readability Tool for English. NLP Tools for the Social Sciences. linguisticanalysistools.org. Retrieved September 1st, 2021. URL: https://www.linguisticanalysistools.org/arte.html (accessed 26.05.2022).

Churunina, Anna A., Ehl'zara Gizzatullina-Gafiyatova, Artem Zaikin & Marina I. Solnyshkina. 2020. Lexical Features of Text Complexity: The case of Russian academic texts. In *SHS Web of Conferences*. Nizhny Novgorod, Russia: EDP Sciences.

Coltheart, Max. 1981. The MRC psycholinguistic database. *The Quarterly Journal of Experimental Psychology Section A* 33(4). 497–505.

Conneau, Alexis, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer & Hervé Jégou. 2018. Word translation without parallel data. In *6th International Conference on Learning Representations*. Vancouver, BC, Canada: OpenReview.net.

Crossley, Scott A., Franklin Bradfield & Analynn Bustamante. 2019. Using human judgments to examine the validity of automated grammar, syntax, and mechanical errors in writing. *Journal of Writing Research* 11(2). 251–270.

Crossley, Scott A., Kristopher Kyle, Jodi Davenport & Danielle S. McNamara. 2016. Automatic assessment of constructed response data in a Chemistry Tutor. In *International Conference on Educational Data Ining.* Raleigh, North Carolina, USA: International Educational Data Mining Society. 336–340.

Dale, Edgar & Jeanne S. Chall. 1948. A formula for predicting readability: Instructions. *Educational Research Bulletin* 27(1). 37–54.

Dascalu, Mihai. 2014. *Analyzing Discourse and Text Complexity for Learning and Collaborating, Studies in Computational Intelligence* (534). Switzerland: Springer.

Dascalu, Mihai, Philippe Dessus, Stefan Trausan-Matu & Maryse Bianco. 2013. ReaderBench, an environment for analyzing text complexity and reading strategies. In H. Chad Lane, Kalina Yacef, Jack Mostow & Philip Pavlik (eds.), *16th Int. Conf. on Artificial Intelligence in Education (AIED 2013),* 379–388. Memphis, TN, USA: Springer.

Dascalu, Mihai, Danielle S. McNamara, Stefan Trausan-Matu & Laura K. Allen. 2018. Cohesion Network Analysis of CSCL Participation. *Behavior Research Methods* 50(2). 604–619. https://doi.org/10.3758/s13428-017-0888-4

Dascalu, Mihai, Lucia Larise Stavarache, Stefan Trausan-Matu & Philippe Dessus. 2014. Reflecting comprehension through French textual complexity factors. In *26th Int. Conf. on Tools with Artificial Intelligence (ICTAI 2014).* 615–619. Limassol, Cyprus: IEEE.

Dascalu, Mihai, Wim Westera, Stefan Ruseti, Stefan Trausan-Matu & Hub J. Kurvers. 2017. ReaderBench learns Dutch: Building a comprehensive automated essay scoring system for Dutch. In Anne E. Baker, Xiangen Hu, Ma. Mercedes T. Rodrigo, Benedict du Boulay, Ryan Baker (eds.), *18th Int. Conf. on Artificial Intelligence in Education (AIED 2017),* 52–63. Wuhan, China: Springer.

Davies, Mark. 2010. The corpus of contemporary American English as the first reliable monitor corpus of English. *Literary and Linguistic Computing* 25(4). 447–464.

Delvin, Jacob, Ming-Wei Chang, Kenton Lee & Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* Minneapolis, MN, USA: Association for Computational Linguistics. 4171–4186.

Flesch, Rudolf F. 1949. *Art of Readable Writing.*

Gabitov, Azat, Marina Solnyshkina, Liliya Shayakhmetova, Liliya Ilyasova & Saida Adobarova. 2017. Text complexity in Russian textbooks on social studies. *Revista Publicando* 4(13 (2)). 597–606.

Gifu, Daniela, Mihai Dascalu, Stefan Trausan-Matu & Laura K. Allen. 2016. Time evolution of writing styles in Romanian language. In *28th Int. Conf. on Tools with Artificial Intelligence (ICTAI 2016).* San Jose, CA: IEEE. 1048–1054.

Graesser, Arthur C., Danielle S. McNamara, Max M. Louwerse & Zhiqiang Cai. 2004. Coh-Metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, & Computers* 36(2). 193–202.

Guryanov, Igor, Iskander Yarmakeev, Aleksandr Kiselnikov & Iena Harkova. 2017. Text complexity: Periods of study in Russian linguistics. *Revista Publicando* 4(13 (2)). 616–625.

Gutu-Robu, Gabriel, Maria-Dorinela Sirbu, Ionut S Cristian Paraschiv, Mihai Dascălu, Philippe Dessus & Stefan Trausan-Matu. 2018. Liftoff – ReaderBench introduces new online functionalities. *Romanian Journal of Human – Computer Interaction* 11(1). 76–91.

Honnibal, Montani & I. Montani. 2017. Spacy 2: Natural language understanding with bloom embeddings. *Convolutional Neural Networks and Incremental Parsing* 7(1).

Hopkins, Kenneth D. & Douglas L. Weeks. 1990. Tests for normality and measures of skewness and kurtosis: Their place in research reporting. *Educational and Psychological Measurement* 50(4). 717–729.

Kincaid, J. Peter, Robert P. Fishburne Jr., Richard L. Rogers & Brad S. Chissom. 1975. *Derivation of New Readability Formulas: (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel.* Naval Air Station Memphis: Chief of Naval Technical Training.

Kozea. 2016. Pyphen. Retrieved September 1st, 2021. URL: https://pyphen.org/ (accessed 20.05.2022).

Kruskal, William H. & Allen W. Wallis. 1952. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association* 47(260). 583–621.

Kuperman, Victor, Hans Stadthagen-Gonzalez & Marc Brysbaert. 2012. Age-of-acquisition ratings for 30,000 English words. *Behavior Research Methods* 44(4). 978–990.

Kuratov, Yuri & Mikhail Arkhipov. 2019. Adaptation of deep bidirectional multilingual transformers for Russian language. *arXiv preprint arXiv:1905.07213.*

Kyle, Kristopher. 2016. *Measuring Syntactic Development in L2 Writing: Fine Grained Indices of Syntactic Complexity and Usage-based Indices of Syntactic Sophistication.*

Kyle, Kristopher, Scott A. Crossley & Cynthia Berger. 2018. The tool for the automatic analysis of lexical sophistication (TAALES): Version 2.0. *Behavior Research Methods* 50(3). 1030–1046.

Kyle, Kristopher, Scott A. Crossley & Scott Jarvis. 2021. Assessing the validity of lexical diversity indices using direct judgements. *Language Assessment Quarterly* 18(2). 154–170.

Kyle, Kristopher, Scott A. Crossley & Youjin J. Kim. 2015. Native language identification and writing proficiency. *International Journal of Learner Corpus Research* 1(2). 187–209.

Landauer, Thomas K., Peter W. Foltz & Darrell Laham. 1998. An introduction to Latent Semantic Analysis. *Discourse Processes* 25(2/3). 259–284.

LanguageTool. 2021. Language Tool. Retrieved September 1st, 2021. URL: https://languagetool.org/ (accessed 20.05.2022).

Loukachevitch, Natalia V., G. Lashevich, Anastasia A. Gerasimova, Vyacheslav V. Ivanov. Boris V. Dobrov. 2016. Creating Russian wordnet by conversion. In *Computational Linguistics and Intellectual Technologies: Annual conference Dialogue 2016*. Moscow, Russia. 405–415.

Mc Laughlin, G. H. 1969. SMOG grading-a new readability formula. *Journal of Reading* 12(8). 639–646.

Mccarthy, Kathryn, Danielle Siobhan, Marina I. Solnyshkina, Fanuza Kh. Tarasove & Roman V. Kupriyanov. 2019. The Russian language test: Towards assessing text comprehension. *Vestnik Volgogradskogo Gosudarstvennogo Universiteta. Seriya 2: Yazykoznanie* 18(4). 231–247.

Mikolov, Tomas, Kai Chen, Greg Corrado & Jeffrey Dean. 2013. Efficient estimation of word representation in Vector Space. In *Workshop at ICLR*. Scottsdale, AZ.

Myint. 2014. language-check. Retrieved September 1st, 2021. URL: https://github.com/myint/language-check (accessed 23.05.2022).

Pearson, Karl. 1895. VII. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London* 58. 240–242.

Pedregosa, Fabian, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot & Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research* 12. 2825–2830.

Quispesaravia, Andre, Walter Perez, Marco Sobrevilla Cabezudo & Fernando Alva-Manchego. 2016. Coh-Metrix-Esp: A complexity analysis tool for documents written in Spanish. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. 4694–4698.

Rehurek, Radim & Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks.* Valletta, Malta: ELRA. 45–50.

Roscoe, Rod, Laura K. Allen, Jennifer L. Weston & Scott A. Crossley. 2014. The Writing Pal intelligent tutoring system: Usability testing and development. *Computers and Composition* 34. 39–59.

Sadoski, Mark, Ernest T. Goetz & Maximo Rodriguez. 2000. Engaging texts: Effects of concreteness on comprehensibility, interest, and recall in four text types. *Journal of Educational Psychology* 92(1). 85.

Sakhovskiy, Andrey, Valery D. Solovyev & Marina Solnyshkina. 2020. Topic modeling for assessment of text complexity in Russian textbooks. In *2020 Ivannikov Ispras Open Conference (ISPRAS).* Moscow, Russia: IEEE. 102–108.

Schmid, Helmut, Marco Baroni, Erika Zanchetta & Achim Stein. 2007. Il sistema ‘tree-tagger arricchito’–The enriched TreeTagger system. *IA Contributi Scientifici* 4(2). 22–23.

Senter, R.J. & E.A. Smith. 1967. Automated readability index: CINCINNATI UNIV OH.

Shannon, Claude E. 1948. A mathematical theory of communication. *The Bell System Technical Journal* 27(3). 379–423.

Shapiro, S.S. & M.B. Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika* 52(3/4). 591–611.

Sharoff, Serge, Elena Umanskaya & James Wilson. 2014. *A Frequency Dictionary of Russian: Core Vocabulary for Learners*. Routledge.

Solnyshkina, Marina I., Valery Solovyev, Vladimir Ivanov & Andrey Danilov. 2018. Studying text complexity in Russian academic corpus with Multi-Level Annotation. *CEUR WORKSHOP PROCEEDINGS. Proceedings of Computational Models in Language and Speech Workshop, co-located with the 15th TEL International Conference on Computational and Cognitive Linguistics, TEL 2018.*

Solovyev, Valery, Marina Solnyshkina, Mariia Andreeva, Andrey Danilov & Radif Zamaletdinov. 2020. Text complexity and abstractness: Tools for the Russian language. In *International Conference "Internet and Modern Society" (IMS-2020).* St. Petersburg, Russia: CEUR Proceedings. 75–87.

Solovyev, Valery, Marina I. Solnyshkina & Vladimir Ivanov. 2018. Complexity of Russian academic texts as the function of syntactic parameters. In *19th International Conference on Computational Linguistics and Intelligent Text Processing, CICLing*. Hanoi, Vietnam: Springer Lecture Notes in Computer Science.

Spearman, Carl. 1987. The proof and measurement of association between two things. *The American Journal of Psychology* 100(3/4). 441–471.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser & Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems.* Long Beach, CA, USA: Curran Associates, Inc. 5998–6008.

Vorontsov, Konstantin & Anna Potapenko. 2015. Additive regularization of topic models. *Machine Learning* 101(1) 303–323.

***Appendix 1.* Statistically significant ReaderBench indices**

Index	A M (SD)	B M (SD)	χ2(1)	*p*
Max(Dep_nmod / Sent)	0.49 (0.84)	1.24 (1.33)	84.48	<.001
M(POS_noun / Sent)	1.50 (1.25)	2.58 (1.84)	84.31	<.001
M(Dep_nmod / Sent)	0.22 (0.41)	0.64 (0.79)	83.55	<.001
Max(POS_noun / Sent)	2.27 (2.12)	3.82 (2.63)	82.50	<.001
M(Dep_nmod / Par)	1.10 (3.57)	2.14 (2.67)	81.24	<.001
M(UnqPOS_noun / Sent)	1.50 (1.24)	2.51 (1.76)	79.97	<.001
Max(UnqPOS_noun / Sent)	2.26 (2.09)	3.73 (2.54)	78.43	<.001
Max(NgramEntr_2 / Word)	2.05 (0.34)	2.20 (0.45)	76.74	<.001
M(Chars / Word)	3.97 (0.98)	4.43 (1.12)	76.03	<.001
Max(Chars / Word)	9.21 (2.46)	10.76 (3.39)	74.83	<.001
M(POS_noun / Par)	6.96 (18.39)	8.81 (8.13)	73.83	<.001
M(Dep_amod / Par)	1.91 (6.70)	2.41 (2.72)	73.77	<.001
M(Syllab / Word)	1.73 (0.32)	1.89 (0.46)	73.08	<.001
Max(Syllab / Word)	3.66 (1.04)	4.27 (1.39)	72.10	<.001
M(UnqPOS_noun / Par)	5.80 (13.29)	7.96 (7.10)	69.45	<.001
M(POS_adj / Par)	2.65 (8.85)	3.28 (3.39)	69.28	<.001
M(UnqPOS_adj / Par)	2.42 (7.52)	3.18 (3.25)	69.05	<.001
Max(ParseDepth / Sent)	4.06 (1.61)	5.09 (2.06)	66.62	<.001

Index	A M (SD)	B M (SD)	χ2(1)	*p*
Max(Dep_amod / Sent)	0.70 (1.12)	1.29 (1.23)	66.28	<.001
M(Dep_amod / Sent)	0.35 (0.57)	0.73 (0.82)	65.66	<.001
SD(Dep_nmod / Sent)	0.18 (0.36)	0.47 (0.61)	63.21	<.001
Max(POS_adj / Sent)	0.97 (1.30)	1.68 (1.46)	62.27	<.001
Max(UnqPOS_adj / Sent)	0.97 (1.29)	1.66 (1.43)	62.25	<.001
SD(Syllab / Word)	0.88 (0.28)	1.01 (0.41)	62.20	<.001
M(NgramEntr_2 / Word)	0.88 (0.27)	0.98 (0.27)	62.15	<.001
M(POS_adj / Sent)	0.51 (0.67)	0.97 (0.98)	60.85	<.001
M(UnqPOS_adj / Sent)	0.51 (0.66)	0.96 (0.96)	60.81	<.001
SD(Chars / Word)	2.71 (0.71)	3.03 (0.94)	58.24	<.001
M(ParseDepth / Sent)	3.43 (1.00)	4.07 (1.47)	53.99	<.001
SD(Dep_amod / Sent)	0.23 (0.41)	0.46 (0.53)	47.14	<.001
M(Dep_case / Par)	3.03 (7.92)	3.45 (3.74)	39.03	<.001
SD(POS_noun / Sent)	0.61 (0.83)	1.07 (1.15)	38.47	<.001
SD(POS_adj / Sent)	0.34 (0.52)	0.58 (0.62)	38.30	<.001
SD(UnqPOS_adj / Sent)	0.34 (0.52)	0.58 (0.61)	37.76	<.001
SD(UnqPOS_noun / Sent)	0.61 (0.82)	1.04 (1.11)	36.88	<.001
SD(ParseDepth / Sent)	0.54 (0.69)	0.85 (0.81)	34.05	<.001
M(UnqWd / Par)	27.37 (48.36)	31.79 (24.87)	32.74	<.001
SD(NgramEntr_2 / Word)	0.78 (0.18)	0.82 (0.21)	31.88	<.001
Max(UnqWd / Sent)	11.72 (7.00)	14.93 (8.52)	31.81	<.001
M(WdEntr / Par)	2.58 (0.93)	2.84 (1.06)	31.78	<.001
M(Wd / Par)	39.37 (93.19)	41.26 (36.14)	31.65	<.001
Max(WdEntr / Sent)	2.23 (0.67)	2.4 (0.82)	31.30	<.001
Max(Wd / Sent)	12.76 (8.31)	16.56 (10.35)	29.49	<.001
Max(Dep_case / Sent)	1.17 (1.36)	1.69 (1.50)	29.39	<.001
SD(Dep_acl / Sent)	0.03 (0.11)	0.09 (0.20)	29.16	<.001
M(Pron_indef / Par)	1.34 (3.69)	1.65 (2.08)	28.83	<.001
M(Dep_case / Sent)	0.66 (0.78)	0.94 (0.85)	28.77	<.001
SD(Pron_indef / Sent)	0.24 (0.38)	0.4 (0.48)	28.44	<.001
M(Dep_obl / Par)	2.66 (7.21)	2.72 (3.06)	27.91	<.001
M(Dep_det / Par)	0.88 (2.34)	1.22 (1.72)	27.26	<.001
Max(Dep_det / Sent)	0.45 (0.74)	0.77 (1.01)	27.20	<.001
M(Dep_xcomp / Sent)	0.11 (0.26)	0.21 (0.35)	26.88	<.001
SD(Dep_case / Sent)	0.39 (0.54)	0.64 (0.70)	26.82	<.001
M(Dep_xcomp / Par)	0.60 (1.89)	0.76 (1.21)	26.80	<.001
Max(Dep_xcomp / Sent)	0.29 (0.55)	0.51 (0.74)	26.43	<.001
SD(Wd / Sent)	2.47 (3.25)	3.86 (3.92)	26.32	<.001
Max(Pron_indef / Sent)	0.61 (0.85)	0.92 (0.99)	26.19	<.001
M(MidEndCoh / Par)	0.23 (0.32)	0.35 (0.35)	25.89	<.001
Max(LemmaDiff / Word)	1.31 (0.87)	1.63 (0.99)	25.80	<.001
M(Dep_acl / Sent)	0.02 (0.12)	0.06 (0.14)	24.68	<.001
SD(Dep_det / Sent)	0.19 (0.36)	0.33 (0.45)	24.57	<.001
SD(UnqWd / Sent)	2.17 (2.77)	3.31 (3.28)	24.31	<.001
M(Dep_acl / Par)	0.17 (1.02)	0.26 (0.61)	24.23	<.001
Max(Dep_acl / Sent)	0.08 (0.30)	0.20 (0.43)	24.18	<.001
M(Dep_nummod / Sent)	0.04 (0.22)	0.10 (0.30)	23.29	<.001

Index	**A** **M (SD)**	**B** **M (SD)**	**χ2(1)**	***p***
M(Dep_det / Sent)	0.19 (0.33)	0.33 (0.62)	22.94	<.001
Max(Dep_nummod / Sent)	0.12 (0.41)	0.29 (0.64)	22.90	<.001
M(Dep_nummod / Par)	0.16 (0.62)	0.35 (0.86)	22.10	<.001
Max(Dep_obl / Sent)	1.02 (1.25)	1.38 (1.27)	21.88	<.001
M(UnqWd / Sent)	9.02 (4.39)	10.94 (6.11)	21.57	<.001
M(Pron_indef / Sent)	0.29 (0.43)	0.43 (0.56)	21.20	<.001
SD(Dep_xcomp / Sent)	0.12 (0.24)	0.23 (0.35)	21.19	<.001
SD(Dep_obl / Sent)	0.36 (0.51)	0.54 (0.60)	20.91	<.001
SD(Dep_nummod / Sent)	0.05 (0.18)	0.13 (0.30)	20.37	<.001
M(Sent / Par)	3.58 (7.30)	3.32 (2.59)	20.15	<.001
M(Wd / Sent)	9.57 (4.95)	11.84 (7.39)	19.55	<.001
SD(Repetitions / Sent)	0.34 (0.68)	0.53 (0.82)	18.27	<.001
M(Dep_conj / Par)	2.18 (5.95)	2.08 (2.48)	18.14	<.001
M(Dep_obl / Sent)	0.54 (0.68)	0.73 (0.71)	17.79	<.001
M(Commas / Par)	3.01 (7.55)	3.12 (3.43)	16.97	<.001
M(Dep_appos / Sent)	0.09 (0.27)	0.18 (0.41)	16.95	<.001
M(WdEntr / Sent)	1.99 (0.57)	2.09 (0.72)	16.82	<.001
SD(POS_adv / Sent)	0.37 (0.52)	0.51 (0.57)	16.19	<.001
M(StartMidCoh / Par)	0.23 (0.32)	0.32 (0.34)	16.09	<.001
SD(UnqPOS_adv / Sent)	0.36 (0.52)	0.50 (0.55)	15.72	<.001
M(Dep_obj / Par)	1.74 (4.35)	1.89 (2.43)	15.49	<.001
SD(Dep_cc / Sent)	0.27 (0.42)	0.39 (0.46)	15.46	<.001
Max(Dep_appos / Sent)	0.21 (0.51)	0.36 (0.66)	15.25	<.001
Max(Dep_conj / Sent)	0.94 (1.33)	1.23 (1.34)	15.03	<.001
SD(Dep_advmod / Sent)	0.40 (0.56)	0.57 (0.66)	14.70	<.001
M(Dep_appos / Par)	0.33 (1.07)	0.42 (0.86)	14.34	<.001
M(UnqPOS_adv / Par)	1.94 (3.90)	2.21 (2.50)	13.75	<.001
Max(UnqPOS_adv / Par)	1.94 (3.90)	2.21 (2.50)	13.75	<.001
SD(Pron_int / Sent)	0.15 (0.28)	0.24 (0.34)	13.58	<.001
M(Pron_int / Par)	0.57 (1.42)	0.80 (1.22)	13.47	<.001
M(POS_adv / Par)	2.19 (4.85)	2.33 (2.72)	13.44	<.001
M(Punct / Par)	8.34 (18.53)	7.74 (6.6)	13.39	<.001
M(Dep_mark / Par)	0.61 (1.54)	0.84 (1.41)	13.22	<.001
Max(Dep_obj / Sent)	0.75 (0.91)	0.99 (0.99)	12.82	<.001
SD(Dep_conj / Sent)	0.35 (0.56)	0.48 (0.61)	12.76	<.001
M(Dep_nsubj / Par)	4.18 (10.36)	3.77 (3.7)	12.74	<.001
SD(Commas / Sent)	0.45 (0.61)	0.60 (0.66)	12.74	<.001
Max(POS_adv / Sent)	1.01 (1.20)	1.29 (1.27)	12.66	<.001
SD(Dep_mark / Sent)	0.15 (0.29)	0.23 (0.34)	12.62	<.001
Max(Dep_mark / Sent)	0.35 (0.59)	0.53 (0.73)	12.61	<.001
SD(WdEntr / Sent)	0.23 (0.29)	0.30 (0.29)	12.49	<.001
Max(Commas / Sent)	1.32 (1.41)	1.68 (1.53)	12.41	<.001
Max(UnqPOS_adv / Sent)	1.00 (1.18)	1.27 (1.22)	12.32	<.001
Max(Pron_int / Sent)	0.36 (0.57)	0.55 (0.73)	12.32	<.001
SD(Dep_obj / Sent)	0.29 (0.41)	0.40 (0.45)	12.25	<.001
M(Dep_cc / Par)	1.73 (4.53)	1.71 (2.16)	12.14	<.001
M(Repetitions / Par)	1.85 (5.64)	1.96 (3.06)	11.87	<.001

Index	A M (SD)	B M (SD)	χ2(1)	*p*
M(SentAdjCoh / Par)	0.35 (0.33)	0.43 (0.33)	11.81	<.001
M(Dep_mark / Sent)	0.15 (0.31)	0.22 (0.39)	11.66	<.001
M(Dep_advmod / Par)	2.65 (5.95)	2.72 (3.19)	11.36	<.001
M(StartEndCoh / Par)	0.35 (0.34)	0.42 (0.35)	11.32	<.001
M(POS_verb / Par)	5.65 (13.16)	5.16 (5.15)	11.19	<.001
M(UnqPOS_verb / Par)	5.19 (11.26)	4.92 (4.83)	10.90	<.001
SD(NmdEnt_loc / Sent)	0.09 (0.30)	0.15 (0.35)	10.82	.001
SD(POS_verb / Sent)	0.54 (0.67)	0.70 (0.72)	10.74	.001
SD(Punct / Sent)	0.66 (0.84)	0.87 (0.97)	10.32	.001
Max(Repetitions / Sent)	0.98 (1.78)	1.37 (2.13)	10.11	.001
SD(UnqPOS_verb / Sent)	0.54 (0.67)	0.68 (0.71)	9.91	.002
M(Pron_int / Sent)	0.15 (0.28)	0.22 (0.39)	9.69	.002
Max(Dep_advmod / Sent)	1.18 (1.33)	1.48 (1.44)	9.67	.002
Max(Dep_cc / Sent)	0.73 (0.94)	0.93 (0.99)	9.59	.002
M(Dep_fixed / Sent)	0.03 (0.13)	0.08 (0.24)	9.51	.002
SD(LemmaDiff / Word)	0.39 (0.22)	0.42 (0.23)	9.35	.002
M(NmdEnt_org / Sent)	0.01 (0.12)	0.06 (0.29)	9.05	.003
Max(NmdEnt_org / Sent)	0.06 (0.51)	0.12 (0.5)	8.91	.003
M(NmdEnt_org / Par)	0.10 (0.98)	0.14 (0.66)	8.87	.003
SD(Dep_expl / Sent)	0.00 (0.05)	0.02 (0.1)	8.62	.003
M(NmdEnt_loc / Sent)	0.08 (0.29)	0.13 (0.34)	8.60	.003
M(Dep_conj / Sent)	0.47 (0.64)	0.62 (0.86)	8.50	.004
M(NmdEnt_loc / Par)	0.40 (2.53)	0.51 (1.26)	8.42	.004
SD(Dep_fixed / Sent)	0.05 (0.18)	0.12 (0.32)	8.24	.004
M(Dep_obj / Sent)	0.39 (0.49)	0.50 (0.56)	8.17	.004
Max(Dep_expl / Sent)	0.01 (0.10)	0.04 (0.2)	8.16	.004
M(Dep_expl / Sent)	0.00 (0.06)	0.01 (0.08)	8.14	.004
M(Dep_expl / Par)	0.01 (0.13)	0.05 (0.22)	8.13	.004
Max(Dep_fixed / Sent)	0.15 (0.47)	0.25 (0.58)	7.93	.005
SD(Dep_appos / Sent)	0.07 (0.21)	0.13 (0.30)	7.76	.005
Max(NmdEnt_loc / Sent)	0.23 (0.72)	0.33 (0.73)	7.76	.005
M(Dep_fixed / Par)	0.19 (0.63)	0.27 (0.65)	7.67	.006
SD(Dep_iobj / Sent)	0.11 (0.25)	0.15 (0.26)	7.34	.007
SD(POS_pron / Sent)	0.44 (0.58)	0.56 (0.67)	7.02	.008
SD(Dep_advcl / Sent)	0.09 (0.21)	0.13 (0.24)	6.67	.010
SD(Dep_nsubj / Sent)	0.36 (0.47)	0.45 (0.52)	6.62	.010
M(Commas / Sent)	0.74 (0.80)	0.93 (1.01)	6.50	.011
SD(UnqPOS_pron / Sent)	0.42 (0.56)	0.52 (0.60)	6.11	.013
M(Repetitions / Sent)	0.43 (0.73)	0.64 (1.52)	5.74	.017
M(POS_adv / Sent)	0.55 (0.74)	0.65 (0.74)	5.72	.017
SD(Pron_snd / Sent)	0.06 (0.20)	0.11 (0.27)	5.51	.019
M(UnqPOS_adv / Sent)	0.55 (0.73)	0.65 (0.73)	5.49	.019
SD(NmdEnt_org / Sent)	0.01 (0.12)	0.05 (0.22)	5.39	.020
SD(Dep_csubj / Sent)	0.04 (0.15)	0.07 (0.18)	5.07	.024
SD(Dep_ccomp / Sent)	0.07 (0.19)	0.11 (0.23)	5.01	.025
Max(Dep_csubj / Sent)	0.10 (0.32)	0.16 (0.39)	4.85	.028
M(Dep_csubj / Sent)	0.04 (0.14)	0.05 (0.18)	4.80	.029

Index	A M (SD)	B M (SD)	χ2(1)	*p*
M(Dep_ccomp / Par)	0.27 (0.84)	0.36 (0.80)	4.55	.033
Max(Dep_ccomp / Sent)	0.19 (0.43)	0.26 (0.50)	4.35	.037
M(Dep_csubj / Par)	0.16 (0.59)	0.18 (0.46)	4.34	.037
Max(Dep_iobj / Sent)	0.28 (0.54)	0.35 (0.56)	4.28	.039
M(Dep_ccomp / Sent)	0.08 (0.23)	0.10 (0.23)	4.24	.039
M(Dep_iobj / Par)	0.50 (1.43)	0.47 (0.91)	3.95	.047
M(Dep_cc / Sent)	0.38 (0.50)	0.45 (0.55)	3.89	.049

Article history:
Received: 20 October 2021
Accepted: 06 February 2022

Bionotes:

Dragos CORLATESCU is a Teaching Assistant and a PhD student at the University Politehnica of Bucharest researching the field of Natural Language Processing from various perspectives. He has explored the areas of text analysis and classification, assessment of online communities, and chatbot development.
Contact information:
University Politehnica of Bucharest
Splaiul Independentei 313, Bucharest, 060042, Romania
e-mail: dragos.corlatescu@upb.ro
ORCID: 0000-0002-7994-9950

Stefan RUSETI is a Lecturer at the University Politehnica of Bucharest (UPB) with a PhD in Natural Language Processing. He has over 25 publications in the field, 6 of them in top ranked conferences, and an extensive experience in projects using NLP techniques and AI (deep neural networks) architectures.
Contact information:
University Politehnica of Bucharest
Splaiul Independentei 313, Bucharest, 060042, Romania
e-mail: stefan.ruseti@upb.ro
ORCID: 0000-0002-0380-6814

Mihai DASCALU is Full Professor at the University Politehnica of Bucharest (UPB) with a strong background in Computer Science applied in Education. He has extensive experience in national and international research projects with more than 200 published papers.
Contact information:
University Politehnica of Bucharest
Splaiul Independentei 313, Bucharest, 060042, Romania
e-mail: mihai.dascalu@upb.ro
ORCID: 0000-0002-4815-9227

Сведения об авторах:
Драгош КОРЛАТЕСКУ – ассистент и аспирант Политехнического университета Бухареста, работает в области обработки естественного языка, имеет опыт анализа и классификаци текстов, оценки онлайн-сообществ и разработки чат-ботов.
Контактная информация:
University Politehnica of Bucharest
Splaiul Independentei 313, Bucharest 060042, Romania
e-mail: dragos.corlatescu@upb.ro
ORCID: 0000-0002-7994-9950

Штефан РУСЕТИ – преподаватель Политехнического университета Бухареста (UPB) имеет степень доктора философии в области обработки естественного языка. Автор более 25 публикаций в этой области, 6 из них – на конференциях с высоким рейтингом, имеет опыт работы в проектах с использованием методов НЛП и архитектуры искусственного интеллекта (глубоких нейронных сетей).
Контактная информация:
University Politehnica of Bucharest
Splaiul Independentei 313, Bucharest 060042, Romania
e-mail: stefan.ruseti@upb.ro
ORCID: 0000-0002-0380-6814

Михай ДАСКАЛУ – профессор Политехнического университета Бухареста (UPB) с большим опытом работы в области компьютерных наук, применяемых в образовании. Участвовал во многих национальных и международных исследовательских проектах, автор более 200 опубликованных работ.
Контактная информация:
University Politehnica of Bucharest
Splaiul Independentei 313, Bucharest 060042, Romania
e-mail: mihai.dascalu@upb.ro
ORCID: 0000-0002-4815-9227