Keyphrase Extraction in Russian and English Scientific Articles Using Sentence Embeddings

Quang Huy Nguyen
St. Petersburg Electrotechnical University
Saint Petersburg, Russia
nguyenquanghuy1997@gmail.com

Mark Zaslavskiy
St. Petersburg Electrotechnical University, JetBrains Research
Saint Petersburg, Russia
mark.zaslavskiy@gmail.com

***Abstract*—Keyphrases provide an overview of the articles, making it a powerful tool for categorizing scientific articles. This paper introduces and describes our supervised machine learning model for automatic keywords extraction. The model calculates features from traditional statistical metrics and new state-of-the-art sentence embeddings to predict a confidence score annotating conformity of keyphrase candidate. The model is tested on corpora of Russian as well as English scientific articles. When compared to the chosen baseline methods of the experiment, our model achieved a comparable F1 score when applied to the Russian corpora; and outperformed them when applied to the English corpora. Using F1-score as the evaluation metric, we also experimented with the model's parameters, such as the embedder and the set of features used as input. We found the pre-trained embedder that provides the best possible outcome for our task and confirmed that our model works best with the full set of features - non of the input to the model is redundant. For future works, we set our goal on deploying the model on existing system. Moreover, we suggest training a delicated embedding module to improve the model performance when working with articles written in Russian.**

I. Introduction

In general, a list of keyphrases provides a crucial role in providing a concrete overview of any scientific literature while also allowing for categorization and queries. For a keyword to be qualified as "good", it must represent the essence of the research and provides the readers with its main idea. Keyphrases are usually chosen manually by authors only when the authors deem it necessary or required by the publishers. As a result, there are numerous articles without keywords, making it harder for other researchers to access and build upon. On the other hand, when authors want to create a list of keywords based on their article, there is no way to assess the relevance of the keyphrase. Therefore, there exists the need for a scoring rubric for keyword evaluation. Most systems providing automated verification of common errors in scientific articles [1] usually cannot check the conformity of keyphrases assigned by authors. Thus, the result of this paper can serve as a valuable resource to improve existing systems. This paper presents a supervised learning model that calculates a score that estimates the quality of a keyword. The score can then be used for other tasks such as assessing keyword candidates or automatic extraction of keywords from research corpora. In our work, we set our interest of research in corpora of scientific articles written in English and Russian. As supervised methods can only work on corpora similar to the corpus they were trained on, we create a method for training the model, which can be used to train both English and Russian versions. The next section features the summarization of related research literature. Section 3 and 4, respectively, present the description of the model and the evaluation and investigation of its performance when applied to research literature written in English and Russian. Finally, in addition to presenting the result and conclusion of the paper, section 5 also features the authors' suggestion on possible future research and practical application of this paper.

II. Related work

This section features some of the traditional methods and more recent methods that are commonly applied when extracting keyphrases. Furthermore, a subsection is dedicated to discussing sentence embedding, since our model relies on this specific technique.

A. Sentence embeddings

Sentence embedding is the task of representing sentences and their semantic information in the form of vectors that can be compared to detect similarity in meaning. Due to the fact that words constitute a sentence, most sentence embedding methods are based on word embedding methods. From word2vec [2], which provides word embeddings, sent2vec [3] and doc2vec [4] utilize average word vectors in ways that produce meaningful sentence vectors. Compared to word2vec, the new state-of-the-art BERT (Bidirectional Encoder Representations from Transformers) not only sets new records in accuracy but also enables the possibility of capturing the context of the word, which means that with the same word in a different context the model would produce different embeddings. However, the sentence embeddings that BERT produces are not suitable to be use with common vector similarity measures, such as cosine similarity. To overcome this shortcoming, Sentence-BERT (SBERT) [5] is a modification of the pre-trained BERT that allows deriving semantically meaningful sentence embeddings, which then can be compared using cosine-similarity. Furthermore, SBERT is proved to be more computationally efficient in terms of speed. Although there is a limit on the length due to memory constraint, SBERT can be applied to both shorter phrases as well as paragraphs containing multiple sentences. Thus, the better and highly meaningful sentence embeddings provided by SBERT serve as the core features of our model. Additionally, the method for making monolingual sentence embeddings multilingual [6] from the same authors further enhances the capability of the SBERT embedder by adding multilingual-support to it,

which subsequently allows sentences with similar meanings in different languages to be mapped closer to each other within the vector space.

B. Keyword extraction

Keyword extraction methods are categorized into two main types: unsupervised methods and supervised methods. Unsupervised methods are further categorized into statistical-based methods and graph-based methods. As one of the simplest statistical approaches to keyword extraction, TF-IDF [8] consists of term frequency (TF) - the number of times the keyword shows up in the article and inverse document frequency (IDF) - the inverse of the number of times the keyword appears in the whole corpus of articles. The more recent EmbedRank [9] is an unsupervised corpus-independent method for keyword extraction using doc2vec [4] sentence embeddings. The keyphrase candidates and the documents are represented in high-dimensional vector space, with the distances between the vectors are then calculated and ranked. Whereas graph-based methods, which involve representing the documents themselves as graphs, are considered to be state-of-the-art regarding unsupervised keyphrase extraction. The basic idea of graph-based methods is to decide the importance of a vertex, which represents a word or phrase within a graph, which represents the document. From the success of PageRank [10] for ranking web pages, various graph-based methods for extracting keyphrases are proposed. For example:

- TextRank [11] represents tokens extracted from the text as nodes while word co-occurrences are described with edges.
- PositionRank [12] instead incorporated the positions of words and their frequency into the graph.
- TopicRank [13] clusters candidate keyphrases into topics before applying the graph-based ranking algorithm. The keyphrases are then selected from the top-ranked topics.
- MultipartiteRank [14] further leveraged TopicRank by storing keyphrase candidates and their topical relations in a single multipartite graph.

In contrast to the aforementioned unsupervised methods, supervised methods required an annotated dataset to train the classifier for labelling keyphrases and non-keyphrases. The KEA (Keyphrase Extraction Algorithm) [15] uses term frequency (TF), inverse document frequency (IDF), and first occurrence, which is the relative position of the first appearance of the phrase in the article as input features for training a Naive Bayes model. Nguyen and Kan [16] extended KEA with morphological features and section distribution vectors (the frequency of keyphrase candidates in different sections of the article) of keyphrases. Existing methods either do not consider the link between the article content and the keyword meaning, or compare the embedding vectors of the keyword and the article, but leave out the relation between the keyphrase and different part of the article (in EmbedRank [9] case, the article is embedded as a single vector). To overcome this, in our model, aiming for the keywords to hold the most relevance to the article, we decided to compute the semantic textual similarity of the candidate keyphrases to sections of the article by using the state-of-the-art sentence embeddings provided by SBERT. Furthermore, since we are only interested in corpora of scientific articles in this work, we decided to choose supervised method for our model.

III. Proposed model

In this paper, we propose a supervised model for extracting keyphrases from scientific articles. This section details our model for extracting keyphrases from documents. The method used to construct the model consists of three main processes:

1) Candidate phrase identification.
2) Feature extraction.
3) Keyphrase extraction using a neural network.

The dataflow in processes are visualized in Fig. 1.

A. Candidate phrase identification

Several methods of choosing candidates are considered, for example, selection based on stopwords and maximum word length [15] [11], selecting all simplex noun phrases [16], selection based on part-of-speech [14]. In our model, candidate extraction from text is performed based on part-of-speech (POS) sequences, as this method allows extracting most of the keyphrases, while avoiding unnecessary candidates with redundant POS sequences. In order to do this, first, the article must go through a Natural Language Processing (NLP) pipelines for POS tagging. In our experiment, we used Stanza [17] for this task, due to its wide range of language support. After that, the candidates are extracted based on a statistical analysis of the common POS sequences that forms keyphrases in the corpora. As the experiments are performed on both English and Russian corpora, different candidates are extracted depending on the language. The pattern sequences for each language are shown in the Table I. The extracted sequences are highlighted.

B. Features extraction

Features are defined as elements of input vectors that are fed to the neural network. After the keyphrases candidates have been extracted from the article, six features are extracted from the candidates and the article. Those features are:

- Keyphrase length: Due to the fact that a keyphrase might contain more than one word, keyphrase length is defined as the number of words in a keyphrase. As a common practice, before being fed to the neural network, features are often rescaled - normalized to the same value range [7]. For keyphrase length, we apply a simple linear rescaling to normalize the value by dividing it by the length of the longest keyphrases in the corpus.
- Term frequency and document frequency: These are metrics commonly used in information retrieval [8].
  - Term frequency is defined by the formula:

$$TF = \frac{tc(P, D)}{|D|}$$

where $tc(P, D)$ - the number of times P appears in D, $|D|$ - the number of words in D

TABLE I. MOST COMMON PART-OF-SPEECH SEQUENCES OF KEYPHRASES AND THEIR RESPECTIVE RATES OF APPEARANCE

English		Russian	
Pattern	Percentage	Pattern	Percentage
NOUN NOUN	24.21%	PROPN PROPN	18.31%
ADJ NOUN	20.43%	ADJ NOUN	16.89%
NOUN	12.90%	NOUN	12.83%
ADJ NOUN NOUN	8.23%	PROPN	11.29%
NOUN NOUN NOUN	5.83%	NOUN NOUN	7.95%
ADJ ADJ NOUN	2.42%	PROPN PROPN PROPN	5.84%
VERB NOUN	2.31%	NOUN ADJ NOUN	2.11%
PROPN	1.79%	ADJ NOUN NOUN	1.72%
VERB NOUN NOUN	1.30%	ADJ ADJ NOUN	1.69%
ADJ NOUN NOUN NOUN	1.14%	NOUN NOUN NOUN	1.51%
NOUN PUNCT NOUN NOUN	1.13%	PROPN X PROPN PROPN	1.22%
Others	<1%	Others	<1%

- Document frequency is defined by the formula:

$$DF = \frac{dc(P, C)}{|C|}$$

where $dc(P, C)$ - the number of documents in corpus C in which P appears, $|C|$ - the size of the examined corpus C

Term frequency indicates the relevance of a phrase in a document, whereas document frequency indicates the importance of a phrase in the corpus. Document frequency is calculated and saved during the training process of the model.

- First appearance: This feature is calculated as the position of the first appearance of the phrase relative to the length of the document, resulting in a number between 0 and 1.
- Similarity curve: The similarity curves display the connection between the varying relationships between the candidate keyphrase and sections of the article. For example, a keyphrase could represent the essence of the introduction or abstract of an article but could seem irrelevant to the section discussing future application or experiment results. Academic publications tend to follow a consistent sequential structure: starting with Abstract, followed by an Introduction, Related Work, Methods, Evaluation, Conclusions and finally References. Based on this assumption, [16] utilized a logical section detection module to extract sections of the article based on headers. However, our model simply divides the whole text into paragraphs of length convenient for our sentence embedder. The candidate keyphrase, together with the paragraphs, then go through a 3-step algorithm to produce the similarity curve:
  1) Calculate embeddings for each paragraph of the article and the candidate keyphrases using sentence embeddings.
  2) Calculate the cosine similarity of the phrase to each of the paragraph embeddings.
  3) Due to the variation in the number of paragraphs between articles, we normalize this feature by creating a 25-point interpolation curve from the graph, built from the similarity scores in the previous step.

Ultimately, we obtain an array of 29 float-number features, all normalized between 0 and 1 for each candidate keyphrase.

C. Keyphrase extraction using neural network

Before the neural network could be used for keyphrase extraction, a training process must be performed. The training required an annotated dataset of documents with annotated keyphrases. Each document in the dataset consisted of the full

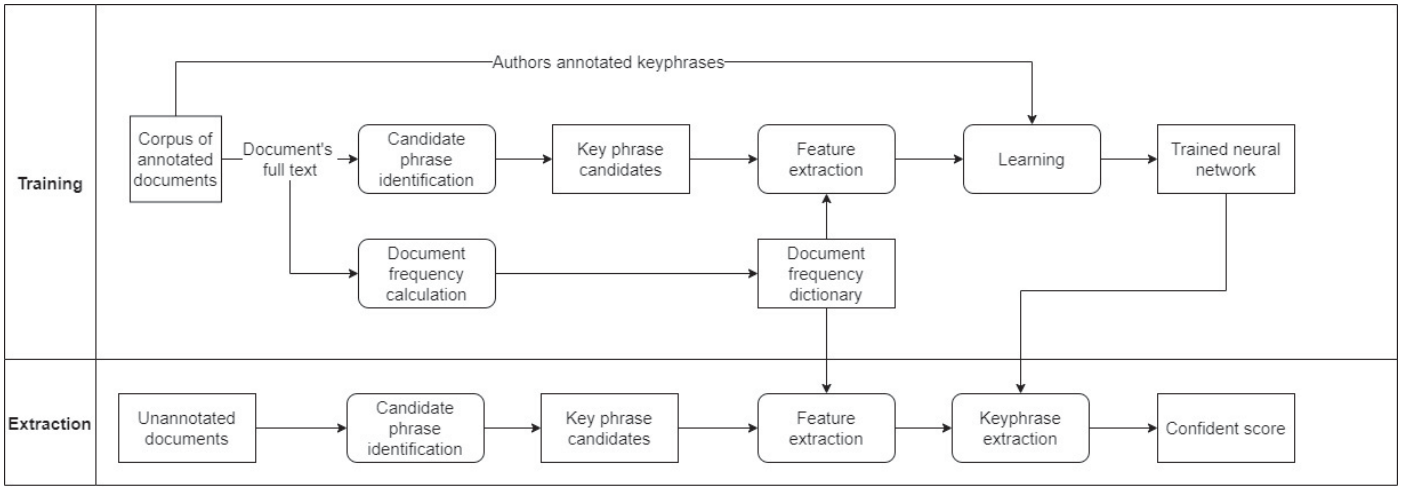

Fig. 1. Dataflow in training and extraction processes

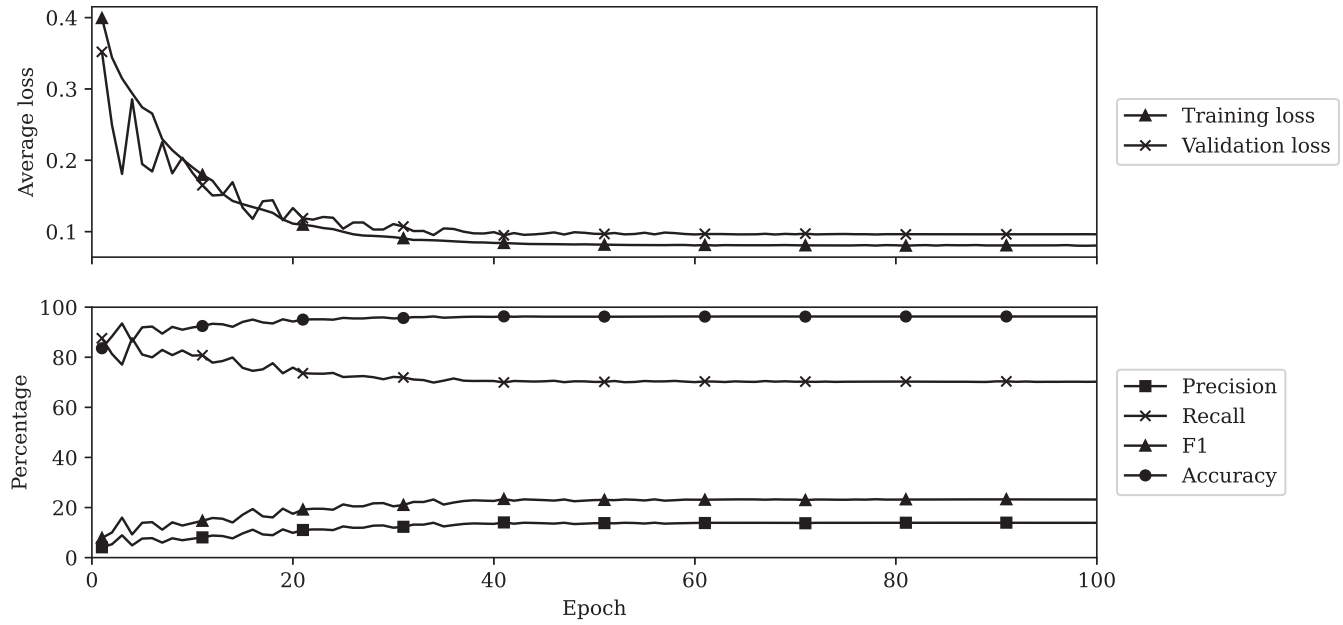

Fig. 2. a. (Top) Average loss over training progress. b. (Bottom) Evaluation metrics over training progress

article's text and a set of keyphrases annotated by the authors or human annotators. After the candidate phrase identification process, we gained a set of candidates. This set of candidates was separated into two sample sets: a positive sample set containing annotated keyphrases, and a negative sample set containing the rest of the candidate keyphrases that we extracted in the previous step. The feature extraction process was applied to each sample in both of these sets to form the input. In the case of input which originated from the positive sample set, we assigned a value of "1.0" as expected output; for input which originated from the negative sample set - a value of "0.0". Both types of samples were redistributed into a training set and a validation set in a 3:1 ratio. Due to the fact that the number of non-keyphrase candidates vastly outnumbered the number of true keyphrases, the research team had to face a class imbalance in the data, which is known to negatively affect the model accuracy [18]. Overcoming this problem required a data manipulation method to balance the dataset [18], which, in our case, was applying simple random-oversampling on the training dataset. A Multilayer Perceptron Neural Network was built and trained with the help of Pytorch [19] on the training set to classify the phrases into one of two categories: positive (keyphrase) or negative (non-keyphrase). The neural network used in our model consists of:

- The input layer that consists of 29 neurons corresponding to the array of 29 float-numbered features received during feature extraction.
- The hidden layers, located between the input layer and output layer, is where all the calculation occurs. The number of layers and number of neurons per layer (which is the same for all hidden layers) plays a key role regarding the behaviour of the network. Based on this information, only through various trials and errors did the research team successfully choose the right value for these two parameters. During our experiment, the number of layers was tested from 1 to 4, and the number of neurons per layer was tested within a range of 300 to 700 with a step of 50 neurons. Based on the performance observed during the training process, the configuration of 3 layers, each with 450 neurons, was considered to be the most optimal. All hidden layers uses LeakyReLU [20] activation function as it has many advantages over the old ReLU [21] activation function including resolving dead neuron issues. Overfitting, which happens when the neural network is too familiar with the training set and does not generalize well when working with new data, is also a problem that we must take into account when building a neural network. Adding a dropout layer is proven to be an effective regularization technique to reduce overfitting when training a network [22].
- The output layer contains a single neuron that serves as the output of the network. A sigmoid activation function is applied to the output layer to map the output to a range between 0 and 1.

We used Adam [23] optimization algorithm to train the model as it is one of the best optimization algorithms for deep learning with fast growing popularity [24]. Over the training progress, the learning rate - the rate at which the network updates its parameters, initially has a value of 0.001 then slowly decayed over time as the loss reduces and the network gets closer to its optimum state. The model was trained on the dataset in mini-batch with a size of 1024 samples and trained for a total of 100 epochs. For performance evaluation, accuracy (total number of cases in which the model predicts correctly/total number of predictions) would not be able to provide us with much information due to the fact that this was an imbalanced classification problem [18]. Therefore, we needed other metrics to evaluate the performance of our model. We chose the 3 metrics:

- Precision: the ratio of correctly predicted positive ob-

TABLE II. THE DATASETS USED IN OUR PAPER

Dataset	Number of documents	Language	Average tokens	Average number of keywords	Keyword missing in documents
Inspec	2000	English	140.3	13.5	45.43%
NUS	211	English	7893.4	10.8	25.68%
CL_mixed	900	Russian	2197.7	7.6	42.58%
CL_long	900	Russian	3117.4	7.9	39.59%
CL_short	900	Russian	1393.7	7.2	38.12%

servations to the total predicted positive observations.

$$Precision = \frac{TP}{TP + FP}$$

where TP - number of true positive cases, FP - number of false positive cases

- Recall is the ratio of correctly predicted positive observations to all observations in the actual class

$$Recall = \frac{TP}{TP + FN}$$

where TP - number of true positive cases, FN - number of false negative cases

- F1 Score is the weighted average of Precision and Recall.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

The total losses in the training process can be seen in Fig. 2.a. The change in metrics that were used to evaluate the performance of the model (Precision, Recall, F1, Accuracy) can be seen in Fig. 2.b.

IV. EXPERIMENTS AND RESULTS

There are commonly two approaches when evaluating a keyphrase extraction model. The first approach involves a human annotator, who reads the article and the result extracted by the model and assesses them manually. This approach requires a high amount of manual effort, and the result can be affected by subjective opinions. The second approach makes use of the metrics like Precision, Recall, and F1-score and compares the extracted list of keyphrases with the list of keyphrases annotated by authors. One can argue that the list of keyphrases annotated by authors can also be subjective. However, with a large number of articles comes a large number of authors. Thus, we can simply treat the poorly annotated keyphrases as noise in the dataset.

A. Dataset

There are many datasets in English available for evaluating keyphrase extraction models. We decided to use the Inspec dataset [25] and the NUS dataset [16]. The Inspec dataset consists of 2000 short documents from scientific journal abstracts with two sets of keyphrases assigned. To compare with other methods, we trained our model on the training set (1500 documents) and evaluated our model on the test dataset (500 documents). NUS [16] consists of 211 full scientific papers in English. Each paper has several sets of keyphrases assigned by authors and annotators. Again, we trained the model on 150 documents and evaluated on 61 remaining documents. Unfortunately, we could not find a publicly available scientific document dataset in Russian. Thus, we constructed our dataset by running a scraper on CyberLeninka. CyberLeninka is a Russian scientific electronic library. A vast collection of scientific articles are available for free. We decided to fetch articles in the field of Computer and Information Science as this is the closest to the field of research of our faculty - Computer Science and Technology, and if the module is going to be added to the existing system for checking student writings [1], this fits the needs of our faculty. In total, 2098 articles with authors assigned list of keyphrases were fetched. Most of them are from VAK (Higher Attestation Commission - VAK [26] is a Russian national government agency that oversees awarding of advanced academic degrees) level journals. Due to memory and resource constraints, we then divided this dataset into three smaller datasets by article length. Each of these smaller datasets consists of 900 articles. The CL-short contains short articles of length fewer than 10000 characters. CL-long contains longer articles of length greater than 10000. And CL-mixed contains 900 random documents - short and long. Similar to the English dataset, these datasets are further split into a training set (600 documents) and a validation set (300 documents). The detailed information on the datasets can be seen in Table II All datasets are going through the same cleanup process, which:

- Remove special symbols.
- Remove the list of annotated keyphrases in the text.

For future researches, the Russian dataset (CLDataset) is made publicly available at [27]

B. Experiment with different sentence embedders

Due to sentence embeddings serving as the foundation for the core features of our model, we want to see how changing the embedding module would have affected the performance of our model. We trained and evaluated the performance of the model with different embedders on CL-mixed datasets.SBERT [5] authors provided 4 multilingual models for semantic textual similarity task [28]. We tested each of these embedder models as the embedding module in our model. The results are populated in Table III

TABLE III. RESULTS ON CL_MIXED VALIDATION SET WITH DIFFERENT EMBEDDING MODELS

Model	Precision	Recall	F1
distilbert-multilingual-nli-stsb-qr	18.58%	79.92%	30.15%
xlm-r-distilroberta-base-paraphrase-v1	31.43%	70.68%	43.52%
distiluse-base-multilingual-cased	35.00%	69.85%	46.64%
xlm-r-bert-base-nli-stsb-mean-tokens	38.71%	64.91%	48.49%

From the result, the xlm-r-bert-base-nli-stsb-mean-tokens embedding model gave the best result in terms of F1-score. Thus, this module was selected as our embedding module in succeeding experiments.

C. Experiment with different sets of features

For the purpose of assessing the importance of the features that were fed into our neural network, we masked some of the features in the input and examined the evaluation metrics when training the network. We tested with 4 set-ups:

- Setup 1: Leaving phrase length (PL), term frequency (TF) and first appearance (FA)
- Setup 2: Leaving PL, TF, FA and document frequency (DF)
- Setup 3: Leaving PL, TF, FA and similarity curve (SC)
- Setup 4: Leaving PL, TF, FA DF and SC

TABLE IV. RESULTS ON CL_MIXED VALIDATION SET WITH DIFFERENT SETS OF FEATURES

Feature set	Precision	Recall	F1
PL+TF+FA	15.11%	80.29%	25.44%
PL+TF+FA+SC	33.67%	66.41%	44.68%
PL+TF+FA+DF	14.78%	81.63%	25.03%
PL+TF+FA+DF+SC	36.10%	65.46%	46.54%

Table IV contained the evaluation result, from which our observation has shown that setup 4 (PL, TF, FA, DF, SC) allowed for better results compared to other setups. This result also solidified the notion that none of our features were unnecessary.

D. Benchmark against other algorithms

In this part, we tested our model against five keyphrase extraction algorithms (implementations provided by pke [29]): TFIDF [8], KEA [15], TextRank [11], Multipartite [14], PositionRank [12]. Due to the fact that the benchmark results of the algorithms when applied to CL_short, CL_long and CL_mixed were observed to be virtually identical, Table V only featured the result for CL_mixed. For TextRank we set the window size to 2 and top-percent (the number of top-ranked words taken into account when building phrases) to 33 percent. For other algorithms, we left the parameters as default. We followed the common practice to stem the golden-scale keyphrases and extracted keyphrases with SnowballStemmer [30]. This was done in order to normalize the result, treating words in different tenses (for both English and Russian), cases, genders (for Russian). All algorithms were fed with the validation set of each dataset, and were required to extract top N candidate keyphrases. The result of the experiment for N=3, N=5 and N=7 is populated in Table V

As shown in Table V, our model performs better than competing algorithms in terms of F1-score for the English models. However, the performance archived on the Russian dataset is at a similar level to KEA and simple TFIDF (only better by a mere percent). The most reasonable explanation for this result would be the differences in the performance of the embedding module when working on different languages. All SBERT models used were originally trained in English, then expanded to support multilingual. As mentioned in the original paper [5], adding more languages to a model could degrade the performance as the capacity of the model would remain the same.

V. CONCLUSION AND FUTURE WORKS

In this paper, we have introduced, described, and evaluated our model for keywords extraction in English and Russian scientific articles. When compared to the chosen baseline methods, models and algorithm, our model was able to achieve a comparable F1 score when applied to the Russian corpora; and outperformed them when applied to the English corpora. The results show that the model could achieve a Recall score of 0.65, which means it could find at least half of the keywords that the author would use to describe his article. Our model has two key aspects: having the multilingual version of SBERT as the foundation of the model, and utilizing Stanza pipelines for preprocessing tasks. These two key aspects allow the models to be adapted for scientific corpora of other languages besides English and Russians. Moreover, as the model provides a confidence score between 0 and 1, it could be used as a reference for authors to assess their choice of keywords. Furthermore, during the evaluation of our research, we also compiled and introduced a preprocessed dataset of more than 2000 scientific publications with an annotated list of key phrases which can be useful for other future work on keyphrase extraction, or in general fields including text mining, information retrieval, and natural language processing. Our current work focuses on the deployment of this model on existing systems, for example, the system for checking students' assignment in scientific writing [1]. In the future, it would be of great interest to train our own sentence embedding module, re-evaluate our model and try to get better results for scientific articles written in Russian.

REFERENCES

[1] E.I. Blees and M.M. Zaslavskiy, "Criteria for text conformity to scientific style", *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, vol.19, Apr.2019, pp.299-305.

[2] T. Mikolov, K. Chen, G.S. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space", *Proceedings of Workshop at ICLR*, Jan.2013.

[3] M. Pagliardini, P. Gupta and M. Jaggi, "Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features", *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol.1, Jan.2018, pp.528-540.

[4] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents", *31st International Conference on Machine Learning, ICML 2014*, vol.4, May.2014.

[5] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks", *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Nov.2019.

[6] N. Reimers and I. Gurevych, "Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation", *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Nov.2020.

[7] C. Bishop, *Neural Networks For Pattern Recognition*. Oxford University Press, Jan.2005.

[8] G. Salton and M.J. McGill, *Introduction to modern information retrieval*. McGraw-Hill, Inc., 1986.

[9] K. Bennani-Smires, C. Musat, M. Jaggi, A. Hossmann and M. Baeriswyl, "EmbedRank: Unsupervised Keyphrase Extraction using Sentence Embeddings", *Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018)*, Oct.2018, pp.221-229.

[10] L. Page, S. Brin, R. Motwani and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web", Stanford InfoLab, 1999.

[11] R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Text", *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Jul.2004, pp.404-411.

TABLE V. COMPARISON WITH OTHER METHODS ON INSPEC, NUS AND CL_MIXED DATASETS

N	Model/Algorithm	Inspec			NUS			CL_mixed		
		Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**
3	TF-IDF	20.80%	4.66%	7.61%	16.94%	7.03%	9.94%	14.11%	5.60%	8.02%
	KEA	20.00%	4.48%	7.32%	16.94%	7.03%	9.94%	14.89%	5.91%	8.46%
	TextRank	9.62%	2.13%	3.49%	10.93%	4.54%	6.41%	5.67%	2.25%	3.22%
	MultipartiteRank	32.73%	7.33%	11.98%	15.30%	6.35%	8.97%	8.78%	3.48%	4.99%
	PositionRank	35.33%	7.91%	12.93%	4.37%	1.81%	2.56%	4.33%	1.72%	2.46%
	Our model	**35.80%**	**8.00%**	**13.08%**	**23.50%**	**9.75%**	**13.78%**	**15.13%**	**6.00%**	**8.59%**
5	TF-IDF	19.28%	7.20%	10.48%	15.41%	10.66%	12.60%	12.00%	7.94%	9.55%
	KEA	17.68%	6.60%	9.61%	16.72%	11.56%	13.67%	**13.20%**	**8.73%**	**10.51%**
	TextRank	11.31%	4.06%	5.98%	7.54%	5.22%	6.17%	4.67%	3.09%	3.72%
	MultipartiteRank	27.87%	10.39%	15.14%	13.44%	9.30%	10.99%	7.33%	4.85%	5.84%
	PositionRank	31.21%	11.63%	16.95%	4.92%	3.40%	4.02%	5.00%	3.31%	3.98%
	Our model	**32.89%**	**12.24%**	**17.84%**	**21.45%**	**14.74%**	**17.47%**	12.68%	8.38%	10.09%
7	TF-IDF	17.60%	9.20%	12.08%	14.75%	14.29%	14.52%	10.33%	9.57%	9.94%
	KEA	16.46%	8.60%	11.30%	15.46%	14.97%	15.21%	11.24%	10.41%	10.81%
	TextRank	10.89%	5.20%	7.03%	7.49%	7.26%	7.37%	4.10%	3.79%	3.94%
	MultipartiteRank	25.47%	13.26%	17.44%	12.88%	12.47%	12.67%	6.86%	6.35%	6.59%
	PositionRank	28.48%	14.80%	19.47%	4.68%	4.54%	4.61%	4.52%	4.19%	4.35%
	Our model	**29.91%**	**15.57%**	**20.48%**	**18.68%**	**17.91%**	**18.29%**	**11.68%**	**10.80%**	**11.23%**

[12] C. Florescu and C. Caragea, "PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents", *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, Aug.2017.

[13] A. Bougouin, F. Boudin and B. Daille, "TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction", *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, Oct.2013, pp.543-551.

[14] F. Boudin, "Unsupervised Keyphrase Extraction with Multipartite Graphs", *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol.2, Jan.2018, pp.667-672.

[15] I.H. Witten, G.W. Paynter, E. Frank, C. Gutwin and C.G. Nevill-Manning, "KEA: Practical Automatic Keyphrase Extraction", *Proceedings of the Fourth ACM Conference on Digital Libraries*, Aug.1999, pp.254-255.

[16] T.D. Nguyen and M. Kan, "Keyphrase Extraction in Scientific Publications", *Proceedings of the Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers, 10th International Conference on Asian Digital Libraries (ICADL 2007)*, Dec.2007, pp.317-326.

[17] P. Qi, Y. Zhang, Y. Zhang, J. Bolton and C.D. Manning, "Stanza: A Python Natural Language Processing Toolkit for Many Human Languages". *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Jan.2020.

[18] X. Guo, Y. Yin, C. Dong, G. Yang and G. Zhou, "On the Class Imbalance Problem", *Fourth International Conference on Natural Computation, ICNC '08*, vol.4, Oct.2008.

[19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library", *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., 2019, pp.8024-8035.

[20] A.L. Maas, A.Y. Hannun and A.Y. Ng, "Rectifier nonlinearities improve neural network acoustic models", *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, Jun.2013.

[21] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines", *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, Jun.2010.

[22] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors", *ArXiv*, vol.abs/1207.0580, Jul.2012.

[23] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", *International Conference on Learning Representations*, Dec.2014.

[24] S. Ruder, "An overview of gradient descent optimization algorithms", *ArXiv*, vol.abs/1609.04747, Jul.2012.

[25] A. Hulth, "Improved Automatic Keyword Extraction Given More Linguistic Knowledge", *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, (EMNLP 03)*, Jun.2003, pp.216-233.

[26] Official website of the Higher Attestation Commision, Web: https://vak.minobrnauki.gov.ru/main.

[27] Q.H Nguyen, CLDataset, Web: https://github.com/levi218/CLDataset.

[28] N. Reimers and I. Gurevych, SBERT Pretrained Models for Semantic Textual Semilarity, Web: https://www.sbert.net/docs/pretrained_models.html.

[29] F. Boudin, "pke: an open source python-based keyphrase extraction toolkit", *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, Dec.2016, pp.69-73.

[30] M.F. Porter, "Snowball: A language for stemming algorithms", Jan.2001.