Automatic Job Ads Classification, Based on Unstructured Text Analysis

Stevan J. Ostrogonac[1], Borko S. Rastović[1], Branislav Popović[2]

[1]Infostud 3 Ltd, Vladimira Nazora 7, 24000, Subotica, Serbia, e-mail: stevan.ostrogonac@infostud.com, borko@infostud.com

[2]Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, 21000, Novi Sad, Serbia, e-mail: bpopovic@uns.ac.rs

Abstract: Machine learning models have been tested on countless classification problems in the past. However, there is little information available on how well they perform when the task is learning abstract concepts, that are difficult to understand, even for humans. The object of this research was to find the best model for capturing the concepts of white-collar and blue-collar jobs, based on unstructured job ads, in Serbian. These concepts have become very difficult to define in the modern job market, since there are now many factors besides the required level of formal education, that determines the category of a job.

Keywords: document classification; natural language processing; neural networks; support vector machines; Serbian

1 Introduction

In recent years, different forms and levels of artificial intelligence (AI) have become common tools in all areas of everyday life. In industry, data analysis and machine learning (ML) are used for automation of several important business elements. One of them is personalization of user experience through various recommender systems [1] or general improvement of user experience by using natural language processing (NLP) to make search by keywords more flexible and efficient [2]. Furthermore, data analysis reports can be automatically generated. They provide relevant information based on which project managers can create business strategies and make important decisions. Search engine optimization is another important aspect of business that can benefit from exploiting machine learning algorithms [3]. However, automation of some parts or even entire production processes within enterprises is the most common application of AI.

The research that will be presented within this paper is a part of the efforts aimed at automating the process for the preparation of job ads, that need to be performed

prior to publishing the contents, on job boards. More precisely, the research is focused on ads classification to white-collar and blue-collar job positions. Based on this classification, each ad is redirected to a suitable portal within a group of job boards which constitute the website of the company Infostud Ltd.

The problem of textual document classification is one of the main contemporary tasks of natural language processing. Text classification is the key component of spam detection systems [4], automatic user complaints classification for the purpose of forwarding them to relevant persons [5], market analysis [6] and the acquisition of textual data for subsequent usage, which usually includes some machine learning algorithms as well.

In most cases, the document classes are well-defined and intuitive, making it easy for humans to deduce to which class a document belongs to, simply by employing common sense and briefly analyzing textual content. In this research, however, the document classes are not defined by some obvious features. In the past, blue-collar and white-collar jobs were distinguished by the level of formal education needed to perform the duties that those positions imply. In the modern job market, the situation is much more complicated, since there are new positions being created much faster than it is possible to form adequate formal education curriculum. Furthermore, the technology allows people with some basic skills to perform complicated tasks. On the other hand, for some jobs that do not inherently require high levels of education, the opposite applies – the technology dictates the level of skills people need to acquire in order to be able to do their work.

Due to the abovementioned state of the job market, job ads administrators classify the ads based on their empirical knowledge about the difficulty level of skills that are explicitly or implicitly provided in the text. Analyzing the text in order to make this decision takes up significant amounts of time. The administrators also need to consult with one another frequently in order to make sure that the ads will be classified in a unified manner. Business strategies can also influence the classification in some cases. Preparing the data and choosing the right machine learning model, as well as tuning the corresponding hyperparameters for automation of this task or similar tasks, is a special category of problems that have not been extensively explored in the past.

The rest of the paper is organized as follows. In Section 2, the dataset that was used in order to train the models is described. In Section 3, the details on data preparation are provided, including the description of tools and text processing methods which had to be developed and some specifics of the Serbian language which had to be addressed. In Section 4, a simplified description of the system for job ads classification based on ML models is provided. Section 5 contains details related to the experiments. Several types of ML models – Naïve Bayes, Logistic Regression, Multi-Layer Perceptron and Support Vector Machines are examined. Finally, in Section 6, some conclusions concerning the potential of ML for learning abstract concepts are drawn and future research topics are discussed.

2 Data Acquisition and Analysis

Over the past decade, a collection of around 230 thousand job ads have been posted on Infostud job portal poslovi.infostud.com. Roughly 80% of the ads are in Serbian. During most of the above-mentioned period, the ads were manually annotated for market analysis purposes. Among the structured data that were collected, minimal and maximal level of formal education required by employers have been noted, as well as standardized names of job positions, job categories based on fields of work and other. Around 15% of the ads were contained within images, therefore they were not used for training in this research (currently, our text extraction module is not accurate enough).

After selecting the ads suitable for training ML models, the dataset consisted of 130000 ads labelled as white-collar or blue-collar. The labels were inferred from the structured data where possible, and the rest of the ads were annotated manually.

Each ad that was used for the training belongs to one of 55 different areas of work. The distribution of job categories that comprised more than 1% of the entire training corpus is shown in Figure 1. Within the dataset, 56000 different job position names (in further text referred to as positions) were present, each of them being annotated as one of 417 standardized position names. Some of the standardized position names were good indicators for white-collar or blue-collar classification. For example, it was a business decision to treat all information technologies (IT) positions as white-collar, due to the wide variety of technical skills that each of those positions imply. On the other hand, some standardized position names include positions of both blue-collar and white-collar nature, such as pharmacist, which is the standardized position name of both pharmacy technicians and bachelor degree pharmacists.

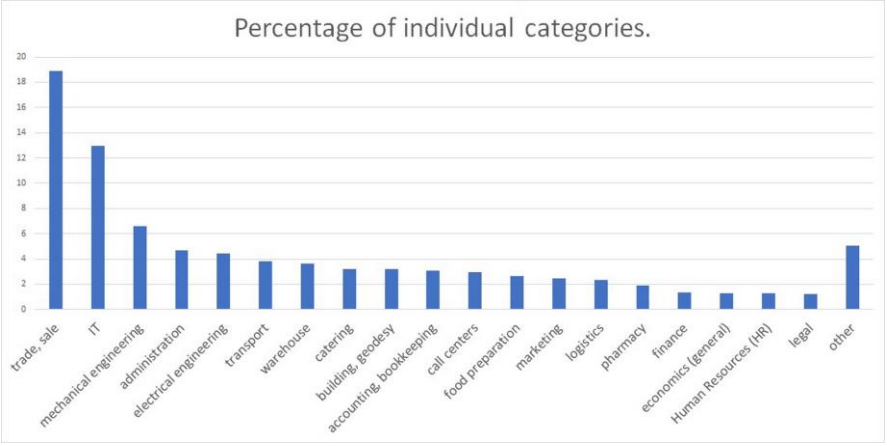

Figure 1
Ad distribution across different fields of work, for the fields that comprise more than 1% of all ads

The most problematic are positions for which it is not clear if they refer to a white-collar or a blue-collar job. Some of these positions are related to standardized position names such as bookkeeper, beautician etc. The ads for these positions would need to be annotated one by one by inspecting textual contents of the ads, which would be extremely time-consuming. Therefore, these ads were not used for the training and the idea was to determine if models can be trained to classify them based on textual data provided from other ads related to similar fields.

Prior to further data preparation, an experiment was conducted in order to determine the importance of word order within these domain-specific texts, which is helpful in choosing the right text representation as input for ML algorithms. Infostud (IS) corpus was compared to a textual corpus that was used for earlier research related to language modeling on the Faculty of Technical Sciences (FTS), University of Novi Sad. Table 1 shows relevant comparative data for the aforementioned corpora. Even though the FTS corpus contains half the number of sentences that comprise the IS corpus, the number of tokens is comparable. This is due to the specific construction of sentences that are typical for job ads that need to reflect short and concise messages. This writing style is similar to journalistic functional style. Extensive studies on the impact of functional styles on the main features of textual corpora have been conducted for Serbian [7-9]. Furthermore, it is an important fact that the vocabulary of FTS corpus is twice as large as the vocabulary of IS corpus. However, the perplexity values are the most indicative. Perplexity on a test set is calculated as the average perplexity on a sentence level, which is calculated as in (1), where m is the length of a sentence, and N is the order of a language model.

$$ppl = \sqrt[m]{\frac{1}{\prod_{i=1}^{m} P\left(w_i \mid w_{i-N+1}, \ldots, w_{i-1}\right)}} \tag{1}$$

Perplexities have been calculated on test portions of each of the corpora (10% of data in both cases) by using previously trained trigram language models. The models were trained using the SRILM language model toolkit [10]. These values show how difficult it is to predict the following word, when previous words are known. The smaller the value, the better the language model. A good language model is an indicator of a high-quality textual corpus. It is important here to emphasize that all the ads form the IS corpus have been reviewed by professional ad administrators, therefore high-quality content is to be expected. From Table 1 it is obvious that IS corpus contains textual content that is much easier to predict. In addition to perplexities, discrimination coefficients have been calculated as well. These values represent perplexities calculated on texts with randomized word orders divided with perplexities calculated on authentic texts [9]. It is evident that the language model trained on the IS corpus can distinguish between natural and random word order very well.

Table 1
Comparative data for textual corpora collected within Infostud and the Faculty of Technical Sciences

	Sentences	Tokens	Vocabulary	Perplexity
FTS corpus	985,498	20,320,616	352,304	294
IS corpus	1,885,625	23,164,550	172,444	16

The above-described experiment indicated that sentence constructions and word orders for job ads are relatively predictable, meaning they do not carry a lot of information. This conclusion implied that the presence of specific words is more important for storing semantic information than their position within textual content. This implied that, within the domain of job ads, bag-of-words (BOW) representation of text [11] might be appropriate for document classification tasks. One alternative to BOW was topic representation [12], which can be obtained by applying Latent Dirichlet Allocation (LDA) [13] to the textual corpus. However, while topic distributions are very useful for classifying job ads based on their fields of work, they do not contain relevant information for classifying the ads to white-collar and blue-collar. Another alternative to BOW was word embedding [14]. Unfortunately, that approach requires a large textual corpus for obtaining vector representations of words, especially for languages with complex morphology, such as Serbian. Therefore, this approach can be taken in the future, after the appropriate training corpus is obtained.

3 Data Preparation

For any machine learning algorithm, it is necessary to prepare data in a way that results in a numerical representation. For the textual content, pre-processing is the first step, performed in order to obtain a sequence of words from the original text, by removing punctuation and other chunks of content useless for a given purpose.

Pre-processing for Serbian is not trivial, since there are no customized tools for performing this task. The most commonly used tools for text pre-processing such as *nltk* library of codes written in Python programming language support many languages, however not Serbian. For this purpose, a Python library called *nlpheart* has been developed within this research [15]. This library, among other features, provides the following text pre-processing steps that are relevant for this research:

- Normalization of symbols (e.g. different forms of quotes need to be replaced with their unique representation)
- Language detection and translation (currently, the language translation module does not produce high-quality results, and therefore it is used only in the prediction phase; in the training phase, only language detection is used to remove content written in languages other than Serbian)

- Separation of words from punctuation (here, various special cases have to be addressed, e.g. web and e-mail addresses, filenames which contain extensions and telephone numbers need to stay in their original forms etc.)
- Conversion of text written in Cyrillic into Latin
- Sentence splitting (not needed for the machine learning algorithm used in this research, but it was needed for the experiment conducted in order to choose text representation model)
- Removal of non-word chunks of text
- Personal data anonymization

After pre-processing, conversion of text into a numerical representation can be performed. For this task, a vocabulary needs to be defined. The vocabulary should include as many of the words that hold useful information for a specific task, but it should not be too large, in order to avoid 'the curse of dimensionality' [16]. The domain-specific vocabulary for Serbian has been extracted from the training corpus. All the words that appeared less than 9 times in the corpus (an empirically determined threshold) have been removed, leaving the resulting vocabulary of 40,758 words. Furthermore, a set of 415 stop-words has been extracted and it was later used in the experiments in order to determine the impact of presence and absence of stop-words on the accuracy of models.

The conversion of each ad (document) to its numerical representation was done by representing the document as a vector which was the size of the vocabulary. Each value within the vector represented the number of times a word with the corresponding index in the vocabulary appeared in the document (BOW model). The vectors corresponding to the documents were, naturally, sparse. Even though there are more advanced numerical representations of text, this is still a common way of data preparation for ML in enterprise systems. The described conversion of text to its numerical representation was performed by using *CountVectorizer* class from a Python library called *scikit-learn*, which is one of the commonly used tools for machine learning, especially for natural language processing [17]. One additional step was performed, in order to improve the quality of data representations (although, not for all the experiments, as will be explained in the following section). This step aimed at giving the words that appear in a small number of ads higher significance and giving lower significance to the words that appear in many different ads, since it is probable that those words are not as important for ad classification purposes. This is performed by employing term frequency – inverse document frequency (tf-idf) algorithm [18], which is implemented within the *scikit-learn* library's *TfidfTransformer* module. The default settings were used as implemented in the library ($l2$ norm etc.)

After all the described steps, the data is finally ready for training the models that will be used for ads classification. It should be noted that the class labels need to be converted to numerical forms as well. In this case, 0 and 1 were enough to map the corresponding blue-collar and white-collar classes.

4 Ads Classification System

In this section, a brief description of the system for ads classification that is based on this research and that is currently being used in Infostud is provided in order to better illustrate the contemporary technologies needed for deploying machine learning models, and to point out some of the practical problems that need to be addressed in the process.

The deployment of the classifier was performed in three steps. The first step presumed creating a web application for serving predictions. For this task, the so-called Flask micro framework written in Python was used [18]. Flask is currently one of the most commonly used tools for creating a web service based on machine learning models due to its simplicity. The second step of the deployment include creating a Docker image in order to containerize the environment needed for the service to function on another machine [19]. The third step was running the image to start a Docker container on local Kubernetes cluster.

The above-mentioned technologies allow efficient delivery of software and automate scaling. Furthermore, Flask application allows the models to be loaded into memory once when the application is started, which is important for the system responsiveness. Loading the models for the ads classification system takes about 850 ms, while the time needed to process the input information and return a prediction is between 10 and 30 ms. The entire process, starting with the ad administration tool sending a request to the containerized application via an exposed port and ending with the tool receiving the response that contains the prediction, takes approximately 150 ms, which is acceptable in the sense that the ad administrators do not notice a delay. The schematic diagram of the system is given in Figure 2. The diagram shows that the classification is performed by combining the decisions of two separate models – one that was trained on the textual contents of the ads, and another that was trained on job position names (ad titles). The models are combined in such a manner that, when they give different outputs, final decisions are made by applying some heuristics. This process will be described in detail in the next section.

Each event of an ad administrator changing the decision of the ad classification system triggers the subsystem for monitoring the classifier's accuracy. The plan is to use the collected data to retrain the models periodically in order to maintain the accuracy, since new jobs are being created and the rules for blue-collar and white-collar class deduction change over time. With this step the entire project of building this enterprise system based on machine learning is completed.

It is important to note that this system can be used for classifying ads in languages besides Serbian, however another service such as Google Translate would be needed. Training separate classifiers for each of the languages in which significant percentage of ads is present would eliminate the need for a translation service, however it would require adequate training datasets.

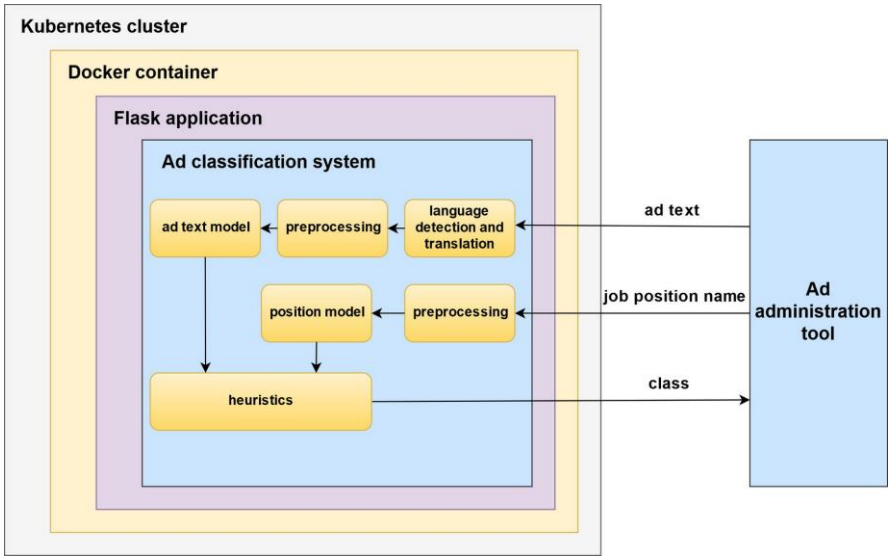

Figure 2
System for job ads classification to blue-collar and white-collar

5 Experiments and Results

The aim of the experiments that will be presented within this section was to test different models and combinations of their respective hyperparameters, as well as, determining how some data preparation steps influence the quality of the resulting classifiers. In other words, in addition to solving the task at hand, the experiments were aimed at obtaining best-practice rules or guidelines for future projects.

The data preparation steps that were tested include the following:

- Usage (or omission) of stop-words in the vocabulary that served for conversion of text to numerical representation
- Usage of the predefined vocabulary vs. the vocabulary inferred from training data
- Usage of higher-order *n*-grams for numerical representation of text (in addition to using the isolated words (unigrams), sequences of words of length 2 (bigrams) and 3 (trigrams) were included in the numerical representation)
- Usage (or omission) of tf-idf scaling of word (*n*-gram) counts

Four types of models were considered in the experiments – Multinomial Naïve Bayes (MNB), Logistic Regression (LR), Multi-layer Perceptron (MLPC) and Support Vector Machine (SVM). The implementations of all the models that have been used are those from *scikit-learn* library [17]. Deep neural networks (DNNs) were initially considered, however the limitations of available hardware and the size of the available dataset indicated that such complex models would not be beneficial.

The test set included the ads that were published on the job portal within two weeks. This set contained 2123 ads, out of which 1450 (68.3%) were blue-collar and 673 (31.7%) were white-collar. The blue-collar and white-collar distribution of the ads form this test set does not reflect the distribution on the entire database, which is close to uniform.

The first set of experiments included training of multiple MNB models. Considering the fact that Naïve Bayes methods are based on the assumption of conditional independence of every pair of features (which would, in this case, mean that the words within a single ad appear in text independently of one another), it could not be expected that these models achieve the highest classification accuracy. However, they were the most convenient for the initial experiments because of their training speed. Therefore, they were used to obtain information related to the effects of data preparation steps on the accuracy of the resulting classifiers. The results of these test are given in Table 2.

It can be concluded that introducing higher order *n*-grams to feature vectors slightly decreases the quality of classifiers, due to a very large number of dimensions of the feature space. Furthermore, applying tf-idf scaling to word counts ads to the model's accuracy significantly, as well as predefining the vocabulary. Finally, excluding stop-words from the vocabulary helps to obtain better models, but this does not apply to the scenarios in which tf-idf is applied and the vocabulary is predefined. The conclusions drawn from this set of experiments have helped to narrow the grid search space for the following experiments. Some of the data preparation steps influence were, however, explored for other models and the conclusions for the experiments with MNB models were confirmed.

The second set of experiments were aimed at finding the best LR model for classification. Different values of the parameter *alpha* (used to compute the learning rate in *scikit-learn* implementation of LR models – *SGDClassifier* with the loss function set to 'log') were tested. The best accuracy was obtained for the value of *alpha* 0.001. Furthermore, the maximal number of training iterations was varied and it was determined that 5 iterations were enough to obtain optimal results. The best LR model reached the accuracy of 78.7%. This is significantly lower than the best accuracy obtained with a MNB model – 82.89%.

Table 2
Accuracy of MNB models trained with different data preparation configurations

***n*-gram order**	**tf-idf applied**	**stop-words excluded**	**vocabulary predefined**	**accuracy [%]**
1-gram	no	no	no	78.38
			yes	81.62
		yes	no	79.64
			yes	82.23
	yes	no	no	81.57
			yes	82.89
		yes	no	81.62
			yes	82.85
2-gram	no	no	no	73.05
			yes	81.62
		yes	no	74.60
			yes	82.23
	yes	no	no	78.89
			yes	82.89
		yes	no	79.31
			yes	82.84
3-gram	no	no	no	72.30
			yes	81.62
		yes	no	74.60
			yes	82.23
	yes	no	no	78.18
			yes	82.89
		yes	no	78.70
			yes	82.84

The third set of experiments were focused on MLPC models. The neural network structure which was explored consisted of four layers – the input layer (the size of the vocabulary – 40758), two hidden layers and the output layer, which consisted of a single neuron that produces the values 0 or 1, that correspond to blue-collar and white-collar classes, respectively. The number of neurons in the hidden layer varied in order to determine the best structure for this task. Other than that, maximal number of epochs and learning rate have also been tested. The results showed that relatively small variations in accuracy can be accomplished by optimizing the hyperparameters, which can be observed from Table 3. MLPC models generally perform significantly better than the previously examined MNB models.

The last experiment included tests with SVM models. When using the predetermined best-practice configuration related to data preparation, the model's

accuracy was 87.91%, which is comparable to the accuracy of the best MLPC model (87.61%). However, the size of the MLPC model was around 160 MB, while the size of the SVM model was around 40 MB, which made the latter better suited for the deployment. Since the test data set contained significantly more blue-collar than white-collar ads, it is important to obtain more detailed information related to the deployed model's accuracy. The weighted precision of the model is 89.86%, and the weighted recall is 87.89%. The most important information can be obtained from the confusion matrix, which looks as in Table 4.

Table 3
Accuracy of MLPC models

maximal number of epochs	size of the first hidden layer	size of the second hidden layer	learning rate	accuracy
10	20	5	0.001	86.05
			0.01	87.28
			0.5	85.67
40	50	10	0.001	86.62
			0.01	87.09
			0.5	87.46
	100		0.001	87.36
			0.01	87.27
			0.5	86.94
	500	20	0.001	84.49
			0.01	87.61
			0.5	86.05

Table 4
Confusion matrix (from the final model, chosen for production)

	Classified as blue-collar	Classified as white-collar
Blue-collar ad	1230	220
White-collar ad	37	636

As it was mentioned in previous sections, in addition to the textual content of the ads, position names (ad titles) were also available for the classification task. Around 56 thousand of annotated short phrases were not adequate for training e.g. a neural network, so MNB model was chosen. The data preparation in this case included all the steps that were used in the previous experiments. If the ad text was written in Serbian, the position name was used as is. The best accuracy in the case when ads were classified based only on the position names was 85.3%. Of course, this raised the question about whether we need to model textual content of the ads at all. Therefore, some error analysis needed to be conducted in order to determine the redundancy of these models.

The analysis showed that the model of ad texts misclassified 257 ads, while the model of position names misclassified 264 ads. However, only 145 ads were misclassified by both models. This meant that there was possibility of gaining better accuracy by combining these models and using some heuristics obtained from error analysis to choose which model's decision to consider as final in which situations. After defining a set of those heuristics, the accuracy of the system that combined the two models was 90.9%. The heuristics included the presence or non-existence of certain indicative words or phrases in ads (e.g. managed, director, scientist, professor, pharmaceutical technician, help desk, etc.). The remaining errors were determined to be in fact related to the positions for which there were no annotated data in the training set. These positions mostly belonged to the fields of bookkeeping, sales and work related to beauty or fitness. However, after the inspection of the misclassified ads, it was determined that those were the ads that the annotators were not unanimous about when deciding about the class. The majority of ads in these fields were easy to classify for the annotators upon reading of the entire text, and the model of text handled those ads very well, even though it was only given the examples that belonged to the related, but not similar fields. The final conclusion of this research was that the created system can replace human annotator for the given task.

Conclusion and Further Research

The research that was presented herein was aimed at exploiting machine learning techniques, to automate the process of job ads classification for blue-collar and white-collar ads. The experiments showed that SVM models, as well as, neural networks have the potential to learn from unstructured text and successfully classify textual documents, based on the concepts that humans cannot easily define or explain by creating a set of rules. The research resulted in knowledge about best practices related to document classification for the explored domain. Furthermore, the work resulted in language resources and text processing tools for the Serbian language, for which, there are currently no alternatives.

Further research on this topic will be directed at automating other steps of ad administration, such as, deducing other elements of the structured data, that are later used by search engines, advanced spell-checking tools and discrimination detection algorithms. Further improvement of the models of textual content will include lemmatization of texts and, in later stages, the introduction of semantic classes, for which a large textual corpus in Serbian will have to be collected. After the implementation of all of the most common text processing tools, the authors plan to release *nlpheart* as an NLP library, for the Serbian language, under GNU General Public License.

Acknowledgement

This research was supported by the Innovation Fund of the Republic of Serbia, under the Matching Grants Program for 2020. The authors would also like to

express their gratitude towards Infostud 3 Ltd. for providing data and necessary conditions for the presented experiments to be conducted.

References

[1] Kitazawa T.: Sketching Dynamic User-Item Interactions for Online Item Recommendation. In Proceedings of the 2017 Conference on Human Information Interaction and Retrieval. Oslo, Norway. doi: 10.1145/3020165.3022152 (2017)

[2] Pandiarajan S., Yazhmozhi V. M., Praveen kumar P.: Semantic Search Engine Using Natural Language Processing. Advanced Computer and Communication Engineering Technology, Lecture Notes in Electrical Engineering 315, Springer, Cham. doi: 10.1007/978-3-319-07674-4_53 (2015)

[3] Yuniarthe Y.: Application of Artificial Intelligence (AI) in Search Engine Optimization (SEO). Paper presented at the 2017 International Conference on Soft Computing, Intelligent System and Information Technology (ICSIIT), Denpasar, Indonesia, September 26-29, doi: 10.1109/ICSIIT.2017.15 (2017)

[4] Karim A., Sami A., Shanmugam B., Kannoorpatti K., Aalazab M.: A Comprehensive Survey for IntelligentSpam Email Detection. IEEE Access 7, November, doi: 10.1109/ACCESS.2019.2954791 (2019)

[5] Rathore M., Gupta D., Bhandari D.: Complaint Classification Using Word2Vec Model. International Journal of Engineering & Technology, Vol. 7, No. 4, pp. 402-404, doi: 10.14419/ijet.v7i4.5.20192 (2018)

[6] Kuo C., Nagasawa Sh.: Applying Machine Learning to Market Analysis: Knowing Your Luxury Consumer. Journal of Management Analytics. doi: 10.1080/23270012.2019.1692254 (2019)

[7] Ostrogonac S.: Automatic Detection and Correction of Semantic Errors in Texts in Serbian. Primenjena lingvistika (Applied Linguistics), No. 17, pp. 265-278, ISSN 1451-7124, UDK: 81'33 (2016)

[8] Ostrogonac S., Pakoci E., Sečujski M., Mišković D.: Morphology Based vs Unsupervised Word Clustering for Training Language Models for Serbian. Acta Polytechnica Hungarica, Journal of Applied Sciences, Joint Special Issue on TP Model Transformation and Cognitive Infocommunications, Vol. 16, No. 2, pp. 183-197, ISSN: 1785-8860 (2019)

[9] Ostrogonac S., Sečujski M., Mišković D.: Impact of Training Corpus Size on the Quality of Different Types of Language Models for Serbian. Paper presented at the 20th Telecommunications Forum TELFOR-2012, Belgrade, Serbia, November 20-22, ISBN: 978-1-4799-1419-7 (2012)

[10] Stolcke, A.: SRILM – An Extensible Language Modeling Toolkit. In Proceedings of ICSLP 2, pp. 901-904, Denver, USA (2002)

[11] Zhang Y., Rong J., Zhi-Hua Zh.: Understanding Bag-of-Words Model: A Statistical Framework. International Journal of Machine Learning and Cybernetics, 1(1-4), pp. 43-52, doi: 10.1007/s13042-010-0001-0 (2010)

[12] Hingmire S., Chougule S., Palshikar G., Chakraborti S.: Document Classification by Topic Labeling. Paper presented at the International Conference on Information Retrieval (SIGIR 2013), Dublin, Ireland (2013)

[13] Blei D., Andrew Ng., Jordan M.: Latent Dirichlet Allocation. Journal of Machine Learning Research, 3(4-5), pp. 993-1022, doi: 10.1162/jmlr.2003.3.4-5.993 (2003)

[14] Mikolov T.: Statistical Language Models Based on Neural Networks. PhD Thesis. Brno University of Technology (2012)

[15] Stevan O., Borko R., Elizaveta L.: A Python Package for Text Processing for Serbian: nlpheart, Scientific Technical Review, Vol. 70, No. 3, pp. 41-45, doi: 10.5937/str2003041O (2020)

[16] Sarkar D.: Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data, 2nd edition, ISBN: 9781484243534 (2019)

[17] Pedregosa et al.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, Vol. 12, pp. 2825-2830 (2011)

[18] Armash Aslam F., Hawa Nabeel M., Lokhande P.: Efficient Way of Web Development Using Python and Flask. International Journal of Advanced Research in Computer Science, Vol. 6, No. 2. doi: 10.26483/ijarcs.v6i2.2434 (2015)

[19] Babak Bashari R., Harrison J., Ahmadi M.: An Introduction to Docker and Analysis of its Performance. JCSNS International Journal of Computer Science and Network Security, Vol. 17, No. 3 (2017)