Mining Labor Market Requirements Using Distributional Semantic Models and Deep Learning

Dmitriy Botov[1], Julius Klenin[1], Andrey Melnikov[2], Yuri Dmitrin[1], Ivan Nikolaev[1], and Mikhail Vinel[1]

[1] Chelyabinsk State University, 129, Bratiev Kashirinykh street, 454001 Chelyabinsk, Russia
dmbotov@gmail.com
[2] Ugra Research Institute of Information Technologies, 151 Mira str., Khanty-Mansiysk, Russia

Abstract. This article describes a new method for analyzing labor market requirements by matching job listings from online recruitment platforms with professional standards to weigh the importance of particular professional functions and requirements and enrich the general concepts of professional standards using real labor market requirements. Our approach aims to combat the gap between professional standards and reality of fast changing requirements in developing branches of economy. First, we determine professions for each job description, using the multilabel classifier based on convolutional neural networks. Secondly, we solve the task of concept matching between job descriptions and standards for the respective professions by applying distributional semantic models. In this task, the average word2vec model achieved the best performance among other vector space models. Finally, we experiment with expanding general vocabulary of professional standards with the most frequent unigrams and bigrams occurring in matching job descriptions. Performance evaluation is carried out on a representative corpus of job listings and professional standards in the field of IT.

Keywords: natural language processing, distributional semantic model, deep learning, convolutional neural networks, multilabel classification, semantic similarity, information extraction, labor market requirements, professional standards

1 Introduction

Nowadays, during the transition to a digital economy, leading industries, such as IT develop more and more rapidly, with technology life cycle being reduced to 1-2 years. Demands of IT companies are constantly changing, while the shortage of qualified personnel is only growing. The concept of lifelong learning, which emerged in the 20th century, becomes increasingly more relevant in the new information age [1]-after all, what was learned only yesterday, today has already lost its relevance. At the same time, requirements of educational and professional standards are too general and do not capture a complete picture of what knowledge and skills are the most relevant, focusing instead on determining which competencies will ensure the successful development of a particular region and the country as a whole. All of these factors force the developers of curricula, online courses, and programs for advanced training and refresher courses to regularly update educational content and ensure the relevance of the learning outcomes. Job listings contain labor market demands, which can be efficiently extracted and analyzed using various data mining approaches, like the ones in [2–6]. However, Russian language and the federal Russian professional standards themselves have certain specifics, which are yet to be investigated in existing research. Thus, we know of no research into the efficiency of various distributive language models and machine learning algorithms in semantic analysis of such texts, which would take into account these specifics. The goal of this paper is the development of such approach; able to mine the actual labor market requirements by comparing them with professional standards, using IT industry as an example. In order to reach this goal, we define the following tasks:

- use the job description to determine if it matches one or several professions according to the classification described in the professional standards for IT industry;
- match specific requirements and duties from job descriptions with the functions and requirements of the standard;
- determine the most significant functions and requirements of the standard;
- enrich the general vocabulary of the professional standard with key concepts from the job descriptions.

To do so, we apply both classic machine learning models and distributive semantic based deep learning algorithms to this task. Evaluation of these models and algorithms is carried out on a representative corpus of job listings and professional standards.

2 Related Work

In [2], authors present their job search engine, which, among other filters, allows user to filter listings by certain skills. Here, skills are acquired by extracting information from social networks, then processed by removing unimportant words and lemmatizing the rest with regard for common word pairs. In order to produce ranked lists of job advertisements, the system weights job description by averaging the weights of each skill in every job description, which in turn are calculated by using TF-IDF and the probability of occurrence for any given skill, based on the title of a job description. While the system does show promising practical results, no scientific evaluation is present to support the claims to quality. Authors of [3] perform an analysis of currently most demanded jobs in several regions, based on the online job listings and a O*NET database of occupational requirements. Their method is based on evaluating similarities between LSA vectors of O*NET descriptions and job listings, in order to map the latter onto the former. Once again, a paper shows the application for an approach, but does not test the quality of the approach itself. Another analytical application is described in [4]. Here, authors apply LSA to the descriptions of a number of online job listings, with the goal of acquiring so-called ideal types of employees: especially effective or successful combinations of skills, which are in high demand. The paper describes various groups of professions, extracted for different numbers of LSA classes, as well as providing the analysis of the ideal skills for each class. A task similar to ours is shown in [5], where authors present their NER-based system, capable of matching skills from job listings onto the skill ontology, generated using Wikipedia. First step is taxonomy generation, using seed phrases from skill descriptions to retrieve matching categories from Wikipedia articles. The second step is skill taggingmatching skill description to one of the surface forms in taxonomy. The paper presents a proper evaluation of the approach with taxonomy having a quality of 83 F1 points, and tagging having a quality of 75.5 F1 points. Authors of [6] presented the evaluation of various approaches for soft skill extraction. The task was specified as a binary classification of text fragments as either containing a description of a soft skill or not. From a variety of classification methods, trained and tested on top of the word2vec vectors for text documents, the best results were achieved by a LSTM neural network, trained on the unmodified texts.

3 Method

3.1 Conceptual Model

Conceptual model (Fig. 1) of the domain can be represented as a directed graph $G = (V, E)$, where V is a set of vertices describing the basic domain concepts, and E is a set of edges defining asymmetric semantic relations between vertices. The set $V = F, R$ is divided into two subsets: Flabor functions/actions and Rrequirements to knowledge and skills, education, or work experience. The set of relationships (edges of the graph) $E = Include, Require, Match$, includes three subsets describing the possible types of relations between concepts:

- $Include \subset F \times F$ is a part-whole relation between generalized labor functions, position, and specific labor actions;
- $Require \subset F \times R$ is a relation between the required knowledge, skill, experience, or education on one side and labor functions or position on the other;
- $Match \subset S \times J$ is a semantic correspondence relation between the common functions and requirements of professional standards: $S = F_s \cup R_s$ and similar specific actions and requirements of job description: $J = F_j \cup R_j$.

3.2 Algorithm

Include and Require relations can be determined by analyzing the structure of professional standards and job descriptions from online recruitment systems and

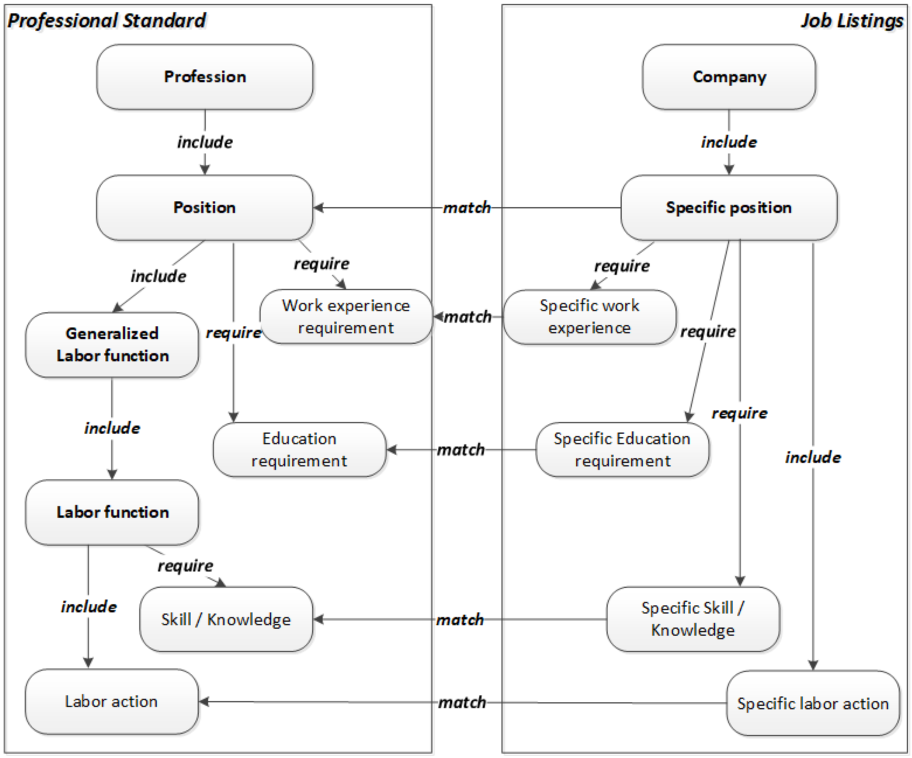

Fig. 1. A conceptual model, which defines the relationships between concepts from professional standards (S) and job descriptions (J)

composing a number of simple rules. To be able to mine the actual requirements of the labor market, we propose to discover Match relations by matching job descriptions published by employers on the Internet with the general requirements of professional standards for relevant professions. We can then determine the significance of certain functions and requirements listed in the standards, based on the frequency of references to them occurring in job listings. Finally, we propose enrichment of the general concepts from professional standards with the most frequent keywords from job descriptions. Thus, the structure and content of professional standards, in a way, acts as a framework, with the particular requirements of the labor market being added on to it, complementing the picture of professional field, placing emphasis on what is currently the highest priority from the point of view of industrys employers. A detailed algorithm is presented in figure 2.

Preprocessing We have tested our models using different combinations of preprocessing steps, as preprocessing does destroy or modify original information. Thus we have the following list of preprocessing steps, some of which we have skipped, if that led to improvement for a given model:

- multiline documents are joined into a single line;
- any symbol that is not alphanumeric, whitespace, or selected special character is removed;

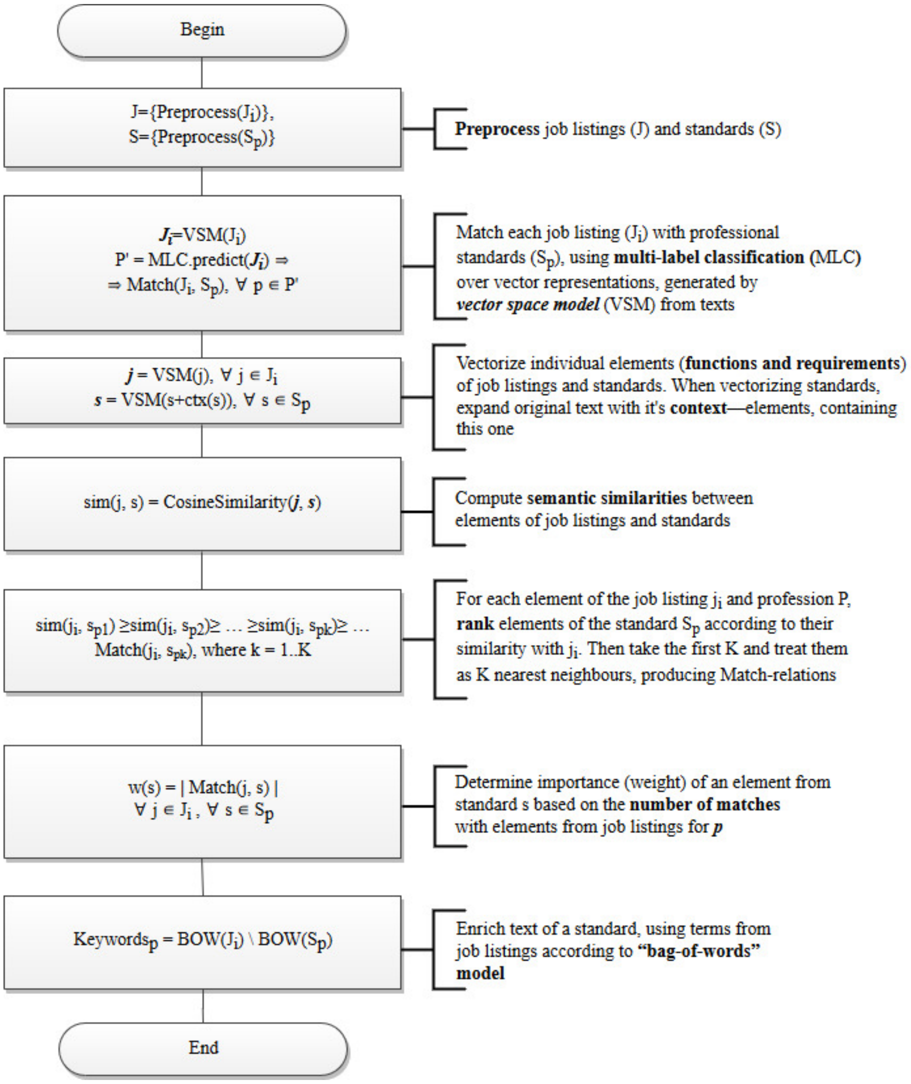
Begin
J={Preprocess(J_i)},
S={Preprocess(S_p)}
Preprocess job listings (J) and standards (S)
J_i=VSM(J_i)
P' = MLC.predict(***J_i***) ⇒
⇒ Match(J_i, S_p), ∀ p ∈ P'
Match each job listing (J_i) with professional standards (S_p), using **multi-label classification** (MLC) over vector representations, generated by ***vector space model*** (VSM) from texts
j = VSM(j), ∀ j ∈ J_i
s = VSM(s+ctx(s)), ∀ s ∈ S_p
Vectorize individual elements (**functions and requirements**) of job listings and standards. When vectorizing standards, expand original text with it's **context**—elements, containing this one
sim(j, s) = CosineSimilarity(***j***, ***s***)
Compute **semantic similarities** between elements of job listings and standards
sim(j_i, s_{p1}) ≥sim(j_i, s_{p2})≥ … ≥sim(j_i, s_{pk})≥ …
Match(j_i, s_{pk}), where k = 1..K
For each element of the job listing j_i and profession P, **rank** elements of the standard S_p according to their similarity with j_i. Then take the first K and treat them as K nearest neighbours, producing Match-relations
w(s) = | Match(j, s) |
∀ j ∈ J_i , ∀ s ∈ S_p
Determine importance (weight) of an element from standard s based on the **number of matches** with elements from job listings for ***p***
Keywords$_p$ = BOW(J_i) \ BOW(S_p)
Enrich text of a standard, using terms from job listings according to **"bag-of-words" model**
End

Fig. 2. Algorithm for mining the requirements of the labor market by matching job listings and professional standards

- sequence is tokenized and each token is lemmatized (where possible);
- lemmatized tokens are appended with their POS-tag;
- stopwords are removed (conjunctions, prepositions, pronouns).

Additionally, for the case of character N-grams, we have replaced word tokenization with separation into character N-grams.

Vector space models Machine learning models in our experiments utilize various vector representations of texts (vector models): one-hot encoding, which represents words and documents with a binary, zero-filled vector, with a one being assigned to the position, corresponding to a specific word in the vocabulary; character N-grams, which are a continuous set of groupings of N characters; TF-IDF (Term Frequency Inverse Document Frequency), a classical frequency-based weighting scheme used often in various NLP tasks; word2vec [7], based on a shallow neural network, trained to connect words with their context; paragraph2vec [8] (sometimes referred to as doc2vec), which considers the document itself as well as the surrounding words, to be a context for any given word.

Multi-label Classification In order to compare a job listing with professional standards, it is necessary to determine which profession it belongs to. Employers formulate positions titles with high degree of variability, so it is impossible to achieve high quality with a simple set of rules or a gazetteer. There are quite a few examples where title is related to one profession, while the description points to a completely different one, usually due to the employer misunderstanding the requirements of professional standards or mixing up similarly named professions (e.g. employers often call the position of a system administrator a system programmer and vice versa). Furthemore, since a single job description can include a description that simultaneously corresponds to several professions (e.g. a system analyst and a software testing specialist, or a programmer and a software architect), we can treat this task as a multilabel classification problem. We use a both a classic algorithms for classification and a more sophisticated neural network classifier. In our research we have tested the following classifiers, as implemented in scikit-learn [9]: LogisticRegression, LinearSVC, GradientBoostingClassifier, RandomForestClassifier, MLPClassifier (neural network model) and others. As for multi-label strategies, we have used the following implementations, provided in scikit-multilearn library [10]: OneVsRestClassifier, BinaryRelevance, ClassifierChain, LabelPowerset, MatrixLabelSpaceClusterer. Text classification can be performed using deep neural networks, such as convolutional neural networks [11] and long short-term memory networks. Such architectures show traditionally high quality in such tasks as classifying intents in conversational systems or performing sentiment analysis including Russian-language datasets [12]. In this paper we experiment with a CNN-based classifier.

Matching concepts In order to determine the most significant concepts in professional standards, we have to match each concept from a job description to their closest concepts from a professional standard. It is worth noting that at this stage we basically perform short text analysis, unlike during the previous steps. To improve matching accuracy, we propose expansion of the concepts text using the text of the labor function and the generalized labor function that contains it in conceptual graph. Text vectorization is performed using one of the basic distributional semantic models: word2vec trained on a large body of job listings and professional standards. This model shows the best results in competitions on semantic similarity and word sense extraction in Russian [13, 14]. To measure semantic similarity itself of vector representations itself, we use a cosine similarity measure.

4 Evaluation Methods and Results

4.1 Text Collections

For various parts of our experiments, we have assemble 4 datasets, which are detailed in table 1. Datasets and experimental results are placed in the repository: https://github.com/master8/vacanciesanalysis. First, to train word and sentence embedding models, we have assembled a large corpus of job listings (Job653K) for positions in the IT industry. To do so we collected some of the listings from the online recruitment platforms: headhunter and superjob. To ensure that collected documents are similarly up-to-date, we have only retrieved postings from the past 5 years. Secondly, we have also collected a collection of 40 professional standards (Std40Label) from category 06 - Communication, information and communication technologies. Before training, we have labelled elements of these standards as either generalized labor functions, concrete labor functions, labor activities, knowledge / skills requirements, work experience requirements, or education requirements. In order to train and evaluate multilabel classification, a separate corpus of job listings was prepared, covering 20 professions (Job20Label), labelled manually by experts. The details of Job20Label are presented in table 2. Details of label counts for different classes are presented in

Table 1. Dataset specifications

Dataset	Document count	Label count (number of professions)	Concept count	Token count	Unique token count (vocabulary size)
Job653K	653K	-	3.6M	130M	200K
Std40Label	40	40	26.6K	574K	3.7K
Job20Label	4652	20	24K	195K	9K
Job120Concepts	98	4	120	754	425

image 3. The class labelled with ”14” is the programmer classthe most overlapping class in the dataset. Finally, from the test dataset of job listings, we have

Table 2. Job20Label dataset detalization

Element type	Element count
Knowledge	8985
Activities	5058
Skills	3803
Posts	3270
Education and training requirements	1916
Requirements for practical experience	1345

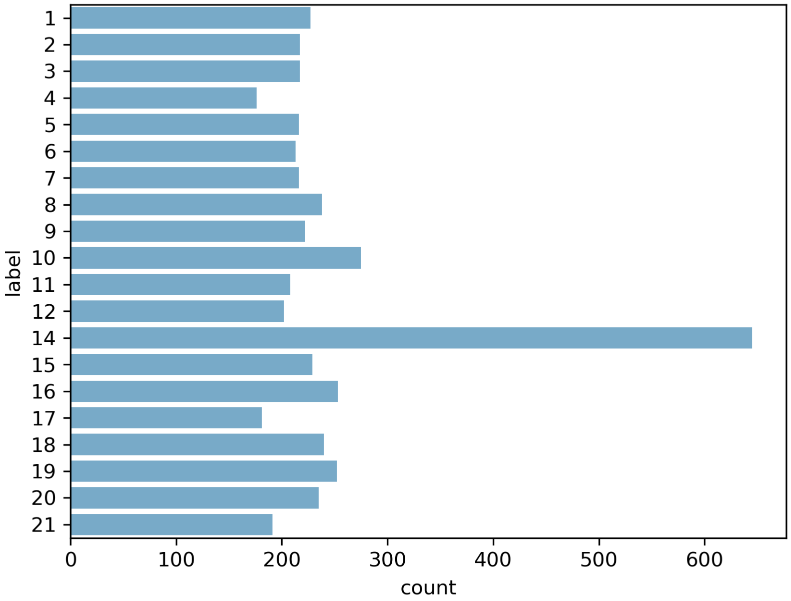

Fig. 3. Label counts for different professions in Job20Label

produced a separate collection of 120 elements (labor actions, knowledge/skills requirements) for 4 professions to serve as an assessment for pairwise matching methods for concepts from job descriptions and standards. (Job120Concepts) Experts assessed the relevance between each element of a job description and 5 most semantically similar (using cosine similarity) elements of professional standards.

Neural Language Models Setup We used word2vec and paragraph2vec (doc2vec) implementation from the gensim library as our neural language models. Table 3 presents the training parameters for them.

Table 3. Training parameters for neural language models

Model	Architecture	Dimensionality	Min. occurence	Epoches
Paragraph2vec	PV-DM	200	3	5
Paragraph2vec	PV-DBOW	200	3	5
Word2vec	skip-gram	300	3	5
Word2vec	CBOW	300	3	5

Multilabel Classification Evaluation and Results Our solution to the job description classification task, was to train a variety of multilabel classifiers using the models and approaches mentioned in section 3. Specifically, the full texts of job descriptions were preprocessed and used in training, including the position title. In our evaluation we compute both micro- and macro-averaged versions of F1 score. The main difference between them is in the way they treat results of individual label classification, with micro approach using results of all labels together, calculating a total precision and recall across all of them, while macro approach calculates precision and recall for each label individually, averaging them afterwards. This results in macro approach being more descriptive of overall quality and micro being more fare when datasets contain imbalanced classes. To reduce the dependency of our results on chance, we perform 5-fold cross-validation, every time splitting dataset randomly 7 to 3, train and test subset respectively. Impact of overfitting should be minimal, since class counts in Job20Label are balanced. Table 4 presents the best classifiers for each of the vector space models. In case of linear classifiers, it can be noted that different multi-label strategies, cause different vector representations to have the best results. Among the basic classification models, Logistic Regression consistently achieved the best results. For instance, in the case of OneVsRestClassifier strategy, the best result was achieved using averaged word2vec, as was the case with LabelPowerset and TF-IDF. Paragraph2vec (Doc2vec) proved to be significantly worse than both averaged word2vec and TF-IDF. However, the best result for either version of F1, was achieved by a more sophisticated CNN classifier, using one-hot word embeddings, on data, preprocessed without lemmatization. It is worth mentioning that the word2vec pretrained on large Internet corpora (wiki, Russian National Corpus (RNC), Araneum) performed significantly worse, in contrast to word2vec models we have trained on our Job653K dataset. CNN architecture, which had the best performance in this experiment, used Keras framework implementation. Details of the its hyperparameters, are presented in Table 5. To analyze effects of the training sample size on the performance of the classifier, we plotted training curves for the best classifiers (Fig. 4). It should

Table 4. Results of multi-label profession classification task

Multi-labeling Classifier/Strategy	Base Classifier	Vector representation	F1-micro	F1-macro
One vs Rest Classifier	Logistic Regression	TF-IDF	0.8384	0.8349
		Avg. Word2Vec (CBOW)	0.8522	0.8515
		Doc2Vec (DBOW)	0.6091	0.6039
LabelPowerset	Logistic Regression	TF-IDF	**0.8767**	**0.8782**
		Avg. Word2Vec (CBOW)	0.8662	0.8682
		Doc2Vec (DBOW)	0.6182	0.6176
	LinearSVC	TF-IDF	0.8529	0.8572
		Avg. Word2Vec (CBOW)	0.8251	0.8294
	MLP Classifier	TF-IDF	0.8326	0.8308
		Avg. Word2Vec (CBOW)	0.8364	0.8390
Convolutional Neural Networks (CNN)		Char N-grams with punct.	0.8730	0.8764
		Char N-grams w/o punct.	0.8592	0.8643
		Word Emb with lemm	0.8734	0.8796
		Word Emb w/o lemm	**0.8900**	**0.8876**

Table 5. Hyperparameters of CNN classifier

Embedding	Vocabulary size depending on the corpus Output size of embedding-layer 300 Spatial dropout 0.2
CNN	Vocabulary size depending on the corpus Output size of embedding-layer 300 Spatial dropout 0.2
Dense	Number of convolution filters 1024 Kernel size 3 Activation function RELU GlobalMaxPooling
Loss function: binary cross-entropy Optimizer: adam with default values The average number of learning epochs: 20	

be noted that CNN already starts leading in F1-macro just at 50 examples per class (out of 20 classes). Analyzing classifier errors, it is worth noting that linear classifiers using word2vec and CNN are more likely to make mistakes on longer job descriptions, where most text is not a description of a position, but rather a description of the company. The quality of classification could be improved by developing rules to remove this kind of information from text.

Concept Matching Evaluation and Results We treat concept matching as a task of finding semantic similarity of paraphrases with paraphrases being individual concepts from the Job120Concepts corpus and 4 professional standards. These standards correspond to professions defined for original job description via multilabel classification. While we match job concepts directly to standard concepts, we do make note of the entire labor function description for each matched standard concept. Relevance of each matched pair of concepts, as well as relevance to the standard concepts containing labor function was determined and labeled by experts. The resulting labelled sequence of selected standard concepts was evaluated using traditional metric used for ranking tasks: mean average precision (MAP) for 1, 3 and 5 concepts, ordered by their predicted semantic similarity. Matching job concepts to concepts of all standards from Std40Label with no filtering based on profession, acts as a baseline for this experiment. This baseline was selected in order to assess the impact preliminary multi-label classification of job descriptions has on the quality of concept matching. TF-IDFs The poor quality of ranking can be explained by the significant difference of vocabularies between professional standards and job descriptions. Table 7 illustrates pairs of concepts matched by averaged word2vec, despite being different in terminology. TF-IDF failed to match these concepts.

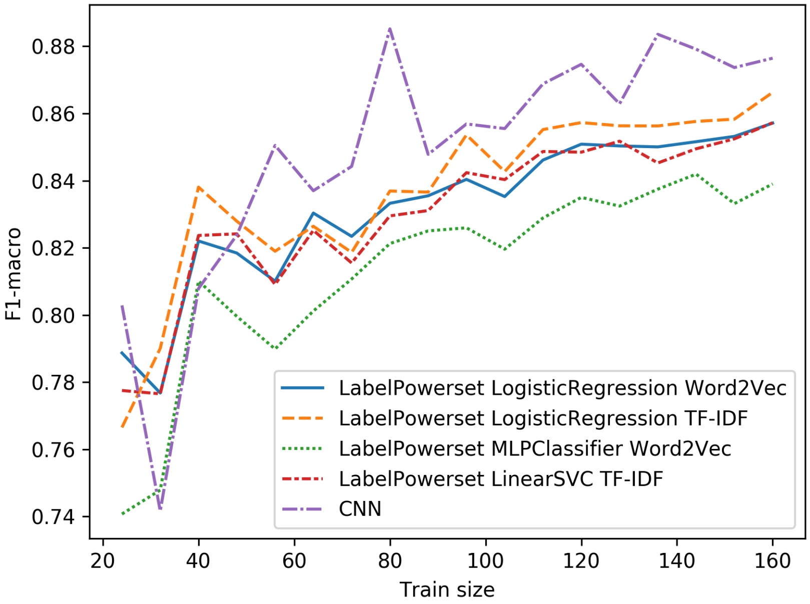

Fig. 4. Learning curves for the best multi-label classifiers (F1-macro / train size per class)

Table 6. Results of concept matching task

Matching method	Distribution semantic model	Matching labor function			Matching labor action/req.		
		MAP@1	MAP@3	MAP@5	MAP@1	MAP@3	MAP@5
Matching with all professional standards (w/o multilabel classification)	TF-IDF	0.6734	0.6944	0.6687	0.4167	0.5278	0.4750
	Avg. word2vec (CBOW)	0.9167	0.8472	0.8333	0.8315	0.7639	0.7583
Matching with one professional standards(using multilabel classification)	TF-IDF	0.8333	0.8611	0.8365	0.6250	0.6667	0.6583
	Avg. word2vec (CBOW)	**0.9967**	**0.9722**	**0.9500**	**0.8333**	**0.8889**	**0.8167**

Table 7. Pairs of job and standard concepts, matched despite using different terminology

Job description	Profession standards	
Job concepts: skill, knowledge, action, experience	Labor function concepts	Labor action, requirements concepts
Build a continuous integration, continuous delivery process	Development of software module integration procedures	Apply methods and tools for assembling modules and software components, developing procedures for deploying software ...
Work in the project team (developers, analysts, key users)	Requirements Development and Software Design	Distribution of tasks between programmers in accordance with the technical specifications
Communication with the customer and the team of Backend programmers	Requirements Development and Software Design	Communicate with stakeholders
Providing professional consulting services for customers	Software requirements analysis	Evaluate and justify recommended solutions.
Refinement of the existing functionality in accordance with user requests	Functional check and software code refactoring	Making changes to the program code to eliminate the detected defects

Keyword extraction Results We enrich professional standards using the most frequent terms from job descriptions. To do so, we have constructed a vocabulary from the texts of job descriptions marked with the label of the relevant profession, with terminology from professional standards excluded from this vocabulary. We then used a term frequency (TF) analysis and the analysis of TF-IDF weights of vocabularies for each class to generate word clouds. Word clouds visualization is implemented using WordCloud [15]. The results of the experiments are presented in the figure 5.

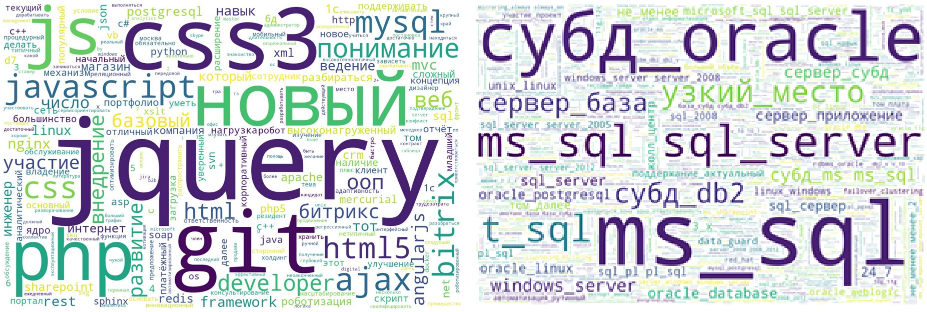

Fig. 5. Learning curves for the best multi-label classifiers (F1-macro / train size per class)

5 Conclusion and Future Work

In this paper, we achieve our goal of creating a method for mining the actual requirements of the labor market, based on a matching concepts between job listings and professional standards which allows us to deal with the gap, existing between some professional standards and ever-developing reality. In our first step, the multi-label classification of jobs by profession, the best results were achieved by a model based on a convolutional neural network trained on one-hot word embeddings of unlemmatized documents. Slightly worse was the performance of a classic logistic regression classifier, trained on TF-IDF vectors, while using LabelPowerset multi-label strategy. In the task of matching individual concepts between job descriptions and professional standards, average word2vec was in a significant lead, when compared to others. We also demonstrated a simple approach to enriching the vocabulary of a professional standard with the most frequent terms from the corresponding vacancies with the option of result visualization. In the future, we plan to continue improving our approach in following ways:

– Explore other distributional semantic models, such as fasttext, as well as topic modeling with additive regularization (ARTM);
– Experimentally evaluate other vector space models for individual concepts, while taking into account the context (structural links in the conceptual model);
– Develop a methodology for determining learning outcomes to be used in creation and updating of the educational programs, relevant to the requirements of the labor market;
– Implement a interactive software interface for labor market analysis and results visualization

6 Acknowledgments

Research has been supported by the RFBR grant No. 18-47-860013 r_a Intelligent system for the formation of educational programs based on neural network models of natural language to meet the requirements of the digital economy. We are grateful to the students and lecturers of Chelyabinsk State University for help in preparing and marking data, as well as in conducting experiments. We are grateful to the head and IT-specialists of the Intersvyaz company (is74.ru) who provided the necessary computational platform for the experiments.

References

1. Gorshkov, M.K., and G.A. Kliucharev: Nepreryvnoe obrazovanie v kontekste modernizatsii. [Continuing education in the context of modernization]. Moscow: IS RAN, FGNU TsSI. 232 p. 2011.

2. Muthyala, R., Wood, S., Jin, Y., Qin, Y., Gao, H., Rai, A.: Data-Driven Job Search Engine Using Skills and Company Attribute Filters. In: 2017 IEEE International Conference on Data Mining Workshops (ICDMW). (2017).
3. Karakatsanis, I., Alkhader, W., Maccrory, F., Alibasic, A., Omar, M.A., Aung, Z., Woon, W.L.: Data mining approach to monitoring the requirements of the job market: A case study. Information Systems. 65, 16 (2017).
4. Mller, O., Schmiedel, T., Gorbacheva, E., Brocke, J.V.: Towards a typology of business process management professionals: identifying patterns of competences through latent semantic analysis. Enterprise Information Systems. 10, 5080 (2014).
5. Zhao, M., F. Javed, F. Jacob, and M. McNair. 2015, January. SKILL: A System for Skill Identification and Normalization. In: Proceedings of the Twenty-Seventh Conference on Innovative Applications of Artificial Intelligence. 40124018, (2015).
6. Sayfullina, L., Malmi, E., Kannala, J.: Learning Representations for Soft Skill Matching. In: Lecture Notes in Computer Science Analysis of Images, Social Networks and Texts. 141152, (2018).
7. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. (2013).
8. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31st International Conference on Machine Learning. Beijing, China. JMLR: WCP 2014. v. 32: 11881196. (2014).
9. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J.: Scikit-learn: Machine learning in Python. Journal of machine learning research. 12(Oct):28252830. (2011).
10. Szymaski, P., Kajdanowicz, T.: A scikit-based Python environment for performing multi-label classification. arXiv preprint arXiv:1702.01460. (2017).
11. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar. 17461751. (2014).
12. Arkhipenko, K., Kozlov, I., Trofimovich, J., Skorniakov, K., Gomzin, A., Turdakov, D.: Comparison of neural network architectures for sentiment analysis of russian tweets. In: Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference Dialogue 2016. Moscow: RGGU. 50-58. (2016).
13. Panchenko, A., Loukachevitch, N., Ustalov, D., Paperno, D., Meyer, C., Konstantinova., N.: RUSSE: the first workshop on Russian semantic similarity. In Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference Dialogue 2015. Moscow: RGGU. v. 2: 89-105. (2015).
14. Panchenko, A., Lopukhina, A., Ustalov, D., Lopukhin, K., Arefyev, N., Leontyev, A., Loukachevitch, N.: RUSSE'2018: a shared task on word sense induction for the Russian language. In: Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference Dialogue 2015. Moscow: RGGU. 547-564. (2018).
15. WordCloud for Python Documentation. Available at: https://amueller.github.io/word$_{c}loud/(accessedNovember29, 2018)$.