# Experimental Comparison of Unsupervised Approaches in the Task of Separating Specializations within Professions in Job Vacancies*

No Author Given

No Institute Given

**Abstract.** In this article we present an unsupervised approach to analyzing labor market requirements, allowing to solve the problem of discovering latent specializations within broadly defined professions. For instance, for the profession of "programmer", such specializations could be "CNC programmer", "mobile developer", "frontend developer", and so on. We have experimentally evaluated various statistical methods of vector representation of texts: TF-IDF, probabilistic topic modeling, neural language models based on distributional semantics (word2vec, fasttext), and deep contextualized word representation (ELMo and multilingual BERT). We have investigated both pre-trained models, and models trained on the texts of job vacancies in Russian. The experiments were conducted on dataset, provided by online recruitment platforms. We have tested several types of clustering methods: K-means, Affinity Propagation, BIRCH, Agglomerative clustering, and HDBSCAN. In case of predetermined number of clusters (k-means, agglomerative) the best result was achieved by ARTM. However, if the number of clusters was not specified ahead of time, word2vec, trained on our job vacancies dataset, has outperformed other models. The models, trained on our corpora perform much better then pre-trained models with large even multilingual vocabulary.

**Keywords:** natural language processing · vector space model · word embedding · topic models · clustering methods · neural language mode.

## 1 Introduction

The modern labor market involves a lot of employers to post various job vacancies. Traditionally, characterization and classification of employees occupation have been done manually. However this issue keeps being longstanding and core one in labor economics. Each employer seeks to find the most eligible applicant that would possess certain skills and proficiencies. However human factor influences the accuracy of job vacancy description or even its title. That is why modern online platforms for job posting are oversaturated with unstructured

* Supported by RFBR

vacancies or those which name do not correspond with its description and defining specialization presents difficulties. This issue is especially challenging for IT professions. For example, the job vacancy can be titled "PHP Programmer", however the description of demands to the candidate either in part or in whole correspond with comparable job vacancy with the name "Frontend Developer" or even "Layout Designer". And if a demand arises to analyze a current state of employers requirements there is no time-saving way to do this. The expert should overview each job vacancy and search patterns with the help of keywords (tags), job names and descriptions simultaneously. The combination of modern computational tools including machine learning could speed up data processing and make it more consistent.

Many foreign organizations, like the United States Census Bureau and the U.S. Bureau of Labor Statistics, are already investigating the potential of using text analysis and machine learning, to automatically classify workers specializations [1].

To solve this challenging problem we suggest to use a way based on unsupervised machine learning algorithms. The main objective of the study is to compare different models of vector representation of texts and clustering methods. The novelty of our research is specified by the use of the most modern contextual language embeddings based on bidirectional LSTM model and contemporary Topic modelling word representation ARTM.

## 2 Related Work

We have studied several papers in which authors describe their approaches to solving this problem. The most widespread approach is methods of simple classification. In [2] authors present their approach for automatically classifying million Webjob vacancies on a standard taxonomy of occupation based on Bag of Word and word2vec for Feature Extraction, SVM and neural networks as classifiers. The similar approach for Italian labor market is presented in [3] they have used word2vec with Continuous Bag of Words (CBOW) and Doc2Vec. As classifier they also use SVM to predict new emerging occupations. More detailed analysis of job vacancy texts based on separated data analysis for title and description, words and semantic meaning is described in [4]. There author extracts information like keywords and bigrams by using KNN for each type of profession. This allows to determine the most relevant skills.

Other authors use tricks to solve the problem of vacancy structure analysis, split up requirements by html-tags. In this way authors of [5] aiming to improve job search engine, pick out the most informative search snippets from the vacancy. For the same purpose, the authors of [6] extract and rank skills by relevance. They start with TF-IDF weight and then adjust the results using topical information based on the job title of the posting. Such approach may be useful for search engines, but it excludes the fact that sometimes employers can give incorrect titles to various job vacancies.

The idea of text analysis with the help of clustering is not novel. In [7] authors describe method K-means in the context of TF-IDF word representation, however there is no legible clustering quality assessment. Also, Affinity Propagation [8] is used as one of the popular methods of text clustering and achieves best results in comparison with other methods [9].

We have considered in detail modern word embeddings and found there is a large number of variations. For example the authors of [10] propose 5 conventional representation methods as baselines of their research (TF-IDF, PCA, SVD, LDA and NMF) the following methods represent utter interest for our work. This methods are compared with recent word embeddings namely word2vec, GloVe, FastText and BERT with embedding vector dimensions of 300, 300, 300 and 768, respectively. The best results were achieved by BERT and FastText plus L2 deep convolutional autoencoders. The result of this study inspired us to use the same methods in our paper. The authors of [11] compare several word representations including LSA, LDA, ParagraphVectors, word2vec and TF-IDF. According to their research, the most efficient word representation is ParagraphVectors with Distributed Bag of Words. However its effectiveness for different modifications depends strongly on dataset choice.

The right choice of word representation can make a big contribution to the result. In [12] authors visually show the difference between various settings of embeddings. In [13] this issue is presented wider, since comparison of topic modelling by LSA, word2vec and GloVe embeddings are also considered. One of the most useful topic modelling tools is described in [14], authors show how topic modelling uncovers a hidden thematic structure of the text. Topic modelling itself is like soft clustering. It splits up a document on several clusters - topics. And here [15] the author describes the ARTM - modern solution for word representation, which looks promising compared to other methods of topic modeling due to its good interpretation abilities and possibility to be used with a pool of regularizers, which can be applied together in any combination.

## 3 Dataset description

For experiment evaluation of clustering we use dataset of 430K job vacancies texts, provided by popular online recruitment platform headhunter. This dataset contains only russian language positions from IT industry.

After that we get 22K of random samples from original data frame and carry out all the experiments and choose about 10% of samples from every profession for marking (manual division by 38 types - 16 for programmers and 22 for other IT professions). The marking was conducted the way that the differences in cluster sizes in the sample on the one hand correspond to the distribution in the main dataset, but differ from each other by no more than an order of magnitude. For that we reduce the biggest data sets and supplement smallest ones. Tags distribution diagram is shown in Fig. 1, y-axis denotes the name of the specializations, x-axis is the cluster volume. The markup takes into account the name of the vacancy and the full text.

The rationale for choosing the number of specializations is based on manual separating by professions. Partially it depends on keywords, thus "C# without special focus on web" (9) and "C# .NET" (7) are different specializations in our opinion. On the other hand, it is based on labor function, thereby "system administrator" (102), "technical support" 108 and "technical support engineer" (121) may have similar title, but 102 is more connected with "in office" support, 108 "on phone" support and 121 is more likely to be described as repairman than others.

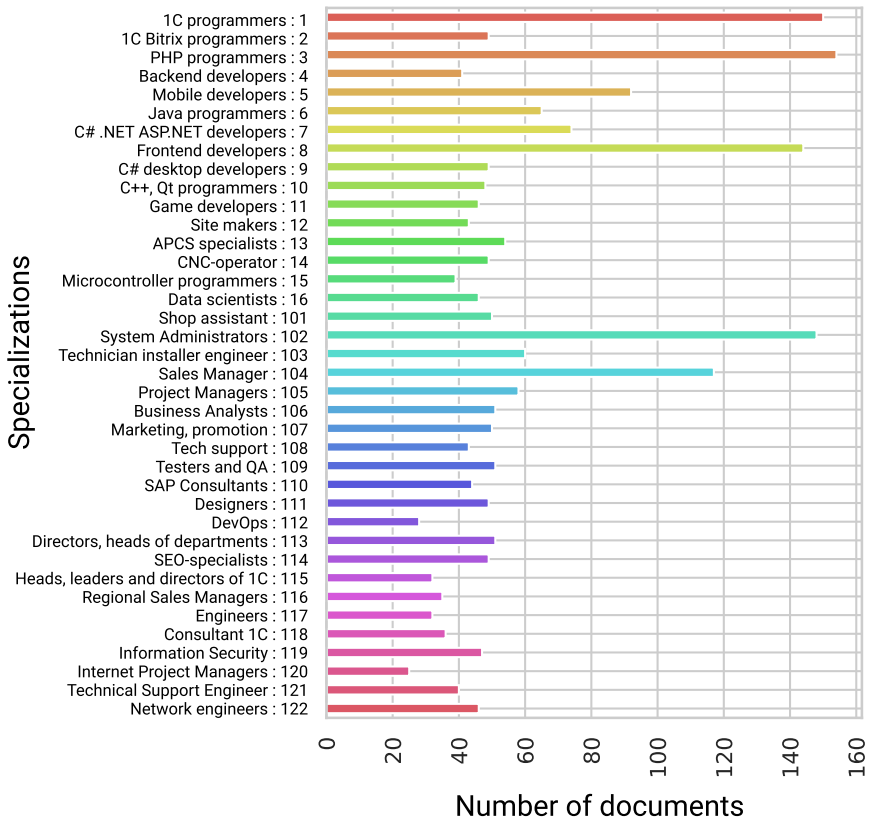


**Fig. 1.** Tags distribution by test dataset.

## 4 Experimental setup

Vector representation models, used it our study are described in section 4.1. Section 4.2 focuses on clustering methods and 4.3 describes evaluation methods and reasons for their choice. Experimenting process is simple and consists of:

- leaning, where we remove all non-specified characters.
- Lemmatization plus POS tags. This step is skipped for multilingual BERT, because this model has been pre-trained on non-lemmatized data
- Vectorization. Embeddings applying, topic modelling or TF-IDF calculation.
- Clustering
- Evaluation. Comparison with the human judgement of separation on the clusters.

Source code of the experiments and dataset laid out in open access:
`https://github.com/omega1996/vacancy_clustering`

### 4.1 Vector space models

Word representations are an integral part of many natural language processing tasks. The quality of vector space models strongly affects the performance of these tasks. However, the evaluation of semantic space is challenging and there are no quantitative criteria of valuability.

#### Word frequency features

*TF-IDF* is used to obtain the document-term matrix of $N \times P$ in which each row corresponds to a job vacancy and each column corresponds to a unique word/token. Truncated Singular Value Decomposition is commonly employed dimensionality reduction technique in which a matrix is reduced or approximated into a low rank decomposition. When the data matrix is obtained by TF-IDF representations, the technique is also known as Latent Semantic Analysis. In our case we use TF-IDF with Truncated SVD dimensionality 80, 300 and 500, built on 22K dataframe.

#### Embeddings

*Word2vec* we use continuous bag-of-words (CBOW) word2vec model for our approach which is based on feedforward neural net language model. We use 300-dimensional word2vec and word2vec weighted by TF-IDF for training on our 430 dataset. For "Avg. W2V_300 job" document vector is got by the average of all the word vectors in a sentence. For "Weighted Avg. W2V 300 job" - by multiplication of word vectors with their TF-IDF scores.

*FastText* is a Facebooks library for text classification and representation. Compared to word2vec, FastText treats each word as composed of character ngrams. The vector for a word is made of the sum of this character n grams. We conduct an experiment with pre-trained model "tayga_none_fasttextcbow_300_10_2019" [16] with dimensionality 300.

*BERT or Bidirectional Encoder Representations from Transformers* is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers. It has 768 dimensions and pre-trained on "wiki_multilingual_cased" corpus [17].

*ELMo or Embeddings from Language Models* is a deep contextualized word representation, developed in 2018 by AllenNLP. It goes beyond traditional embedding techniques and uses a deep, bi-directional LSTM model to create word representations. We use 1024-d models trained on newspapers and twitter as well as 200, 300, 500 and 1024-d ones trained on wikipedia pages [18].

**Topic modelling**

*ARTM or Additive Regularization of Topic Model* Topic modeling is a rapidly developing branch of statistical text analysis. ARTM is flexible non-Bayesian approach with powerful implementation - BigARTM library [14]. Here we use different types of word space - 500, 300, 200 and 80 themes. We teach this representation on 22K dataset with 60 and 30 number of iterations. Practice has shown that the greater number of iterations gives better model's interpretability.

*LSA* assumes that words that are close in meaning will occur in similar pieces of text (the distributional hypothesis) and we train it on 22K and distinguish 500 topics.

*LDA* is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. For the training, a dataset 22K is chosen, it also contains 500 topics.

### 4.2 Clustering methods

Globally, clustering methods can be divided into two types according to the method of determining the number of clusters: pre-set and non specified. In our research we have tested the following clustering methods:

- **K-means**: One of the most popular and simple clustering algorithms. K represents the number of categories identified, with each categorys average (mean) characteristics being appreciably different from that of other categories.
- **Agglomerative** is a method of cluster analysis which seeks to build a hierarchy of clusters.

– We have also examined **HDBSCAN** an extension of DBSCAN, but after preliminary experiments, it was decided not to conduct a study of the HDBSCAN algorithm as it shows incomparably low quality scores and defines many objects as noise.

For the previous three types of clustering methods, we use the same pre-set number of clusters based on our manual marking equal to 38.

– **Affinity Propagation** is based on the concept of "message passing" between data points. Each iteration data point calculate "responsibilities" and "availabilities" between neighbours to become a possible cluster center.
– **BIRCH (balanced iterative reducing and clustering using hierarchies)** An advantage of BIRCH is its ability to incrementally and dynamically cluster incoming, multi-dimensional metric data points in an attempt to produce the best quality clustering for a given set of resources (memory and time constraints).

Number of clusters is not specified for the later two types.

### 4.3 Evaluation

The task of assessing the quality of clustering is more complex compared to assessing the quality of classification. Estimates should not depend on the values of the labels themselves, but only on the sample fragmentation itself. For our study we choose simple and prevalent metrics, which are independent of label values and permutations and insensitive to cluster size differences.

The following external measures of effectiveness are used in the study: Adjusted Mutual Information (AMI), Adjusted Rand Index (ARI) and V-measure. We use several efficiency measures, since for this task there is no single generally accepted measure. It is worth noting that if we train embeddings or vector models, the training sample for embeddings and the test sample marked up by experts do not overlap with each other. The description of marking is presented in paragraph 3.

## 5 Results and Discussion

After conduction and evaluation of the experiments, we present the following results of our work.

Overviewing the data in the Table 1, three word representation types are leading: ARTM on 500 and 300 topics, 60 and 30 iterations respectively, word2vec with 300 dimensions and TF-IDF with SVD reduction to 300. Although the difference between the results of clustering algorithms is significant, the main leaders are always the same. Dimensionality reduction of neural word embeddings makes results worse.

In Fig. 2 the distribution between clusters overlays on specializations distribution. Different colours mean different clusters. Such a presentation allows to

**Table 1.** Quality metrics are based on our markup - ARI, AMI and V-measure for different vector space models. Larger numbers indicate better performance. Missing values mean very close to zero.

| Model | Clusters | ARI | AMI | V-measure |
|---|---|---|---|---|
| TF-IDF_300 job | KMeans | 0.4429 | 0.6641 | 0.7169 |
| | AffinityPropagation | 0.0961 | 0.2603 | 0.6352 |
| | Agglomerative | 0.3267 | 0.5633 | 0.6355 |
| | Birch | 0.0233 | 0.1683 | 0.3115 |
| ARTM_500_60 job | KMeans | **0.4422** | **0.7091** | **0.7384** |
| | AffinityPropagation | 0.0973 | 0.2819 | 0.6457 |
| | Agglomerative | 0.3931 | 0.6446 | 0.6884 |
| | Birch | 0.3856 | 0.6137 | 0.7034 |
| ARTM_200_30 job | KMeans | 0.4531 | 0.6875 | 0.7189 |
| | AffinityPropagation | 0.1003 | 0.2998 | 0.6539 |
| | Agglomerative | **0.4465** | **0.6709** | **0.7051** |
| | Birch | **0.4812** | **0.6682** | **0.7251** |
| LDA_500 job | KMeans | 0.1580 | 0.4540 | 0.5261 |
| | AffinityPropagation | 0.0466 | 0.1718 | 0.5976 |
| | Agglomerative | 0.1561 | 0.4373 | 0.5137 |
| | Birch | 0.1499 | 0.3521 | 0.5625 |
| LSI_500 job | KMeans | 0.1991 | 0.4421 | 0.5040 |
| | AffinityPropagation | 0.0607 | 0.1915 | 0.5799 |
| | Agglomerative | 0.2395 | 0.4978 | 0.5571 |
| | Birch | | | |
| Avg. W2V_300 job | KMeans | 0.3180 | 0.5858 | 0.6313 |
| | AffinityPropagation | **0.1397** | **0.3537** | **0.6846** |
| | Agglomerative | 0.3253 | 0.6022 | 0.6449 |
| | Birch | 0.3359 | 0.5828 | 0.6264 |
| Weighted Avg. W2V_300 job | KMeans | 0.2289 | 0.4963 | 0.5487 |
| | AffinityPropagation | 0.0794 | 0.2471 | 0.6107 |
| | Agglomerative | 0.2662 | 0.536 | 0.5815 |
| | Birch | | | |
| FastText_300 taiga | KMeans | 0.1640 | 0.3792 | 0.4400 |
| | AffinityPropagation | 0.0463 | 0.1525 | 0.5526 |
| | Agglomerative | 0.1803 | 0.3923 | 0.4506 |
| | Birch | 0.1062 | 0.2361 | 0.4843 |
| FastText_300 job | KMeans | 0.1162 | 0.3592 | 0.4303 |
| | AffinityPropagation | 0.071 | 0.2200 | 0.5748 |
| | Agglomerative | 0.1202 | 0.3799 | 0.4573 |
| | Birch | 0.0831 | 0.3079 | 0.4854 |
| ELMo_300 wiki | KMeans | 0.1818 | 0.4135 | 0.4752 |
| | AffinityPropagation | 0.0408 | 0.1581 | 0.5533 |
| | Agglomerative | 0.1731 | 0.4137 | 0.4715 |
| | Birch | | | |
| ELMo_1024 news | KMeans | 0.1427 | 0.3651 | 0.4289 |
| | AffinityPropagation | 0.0384 | 0.1386 | 0.5447 |
| | Agglomerative | 0.1418 | 0.3714 | 0.4312 |
| | Birch | | | |
| ELMo_1024 twitter | KMeans | 0.1248 | 0.3455 | 0.4074 |
| | AffinityPropagation | 0.0409 | 0.1461 | 0.5454 |
| | Agglomerative | 0.1441 | 0.3530 | 0.4154 |
| | Birch | | | |
| BERT_768 wmc | KMeans | 0.0527 | 0.1786 | 0.2576 |
| | AffinityPropagation | 0.0196 | 0.0888 | 0.4805 |
| | Agglomerative | 0.0514 | 0.1676 | 0.2472 |
| | Birch | | | |

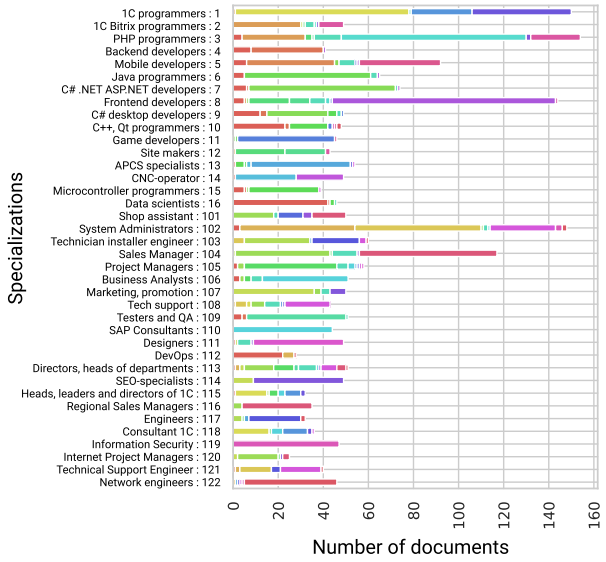


**Fig. 2.** K-means ARTM clustering results matching with specializations. The colors show the predicted cluster.

see visually the completeness and homogeneity of the cluster. V-measure considers both of these values.

High hopes were placed on multilingual BERT. Unfortunately, our expectations were not entirely met. The color spreading on marked labels is presented in Fig. 3. We explain this situation by difference in vocabularies between training dataset and job vacancies text.

After looking at a similar chart for TF-IDF or ARTM (Fig. 4), the difference between best and worst models becomes evident: clusters are arranged more evenly, some of them are determined principally by one color, in contradistinction to distribution shown in Fig. 3.

We noticed that TF-IDF clustered on BIRCH can lead to imbalanced distribution, but word2vec embeddings and topic modelling works great in all cases. Word2vec even breaks the lead on Affinity Propagation. We explain this fact by the structure of word2vec which bases on semantic similarity. Affinity Propagation takes as input a set of pairwise similarities between data points, this structure is suitable for word2vec. As shown in Fig. 5, Affinity Propagation stops at a different number of clusters on different representations of words in our dataset, but the difference is not so significant. On the other hand - BIRCH shows large spread of cluster volumes (Fig. 6), and number of clusters is not necessarily the nearest to determined by human.

K-means performance outstands other clustering methods and greatly combines with topic modelling and TF-IDF. In this case word2vec takes into consid-

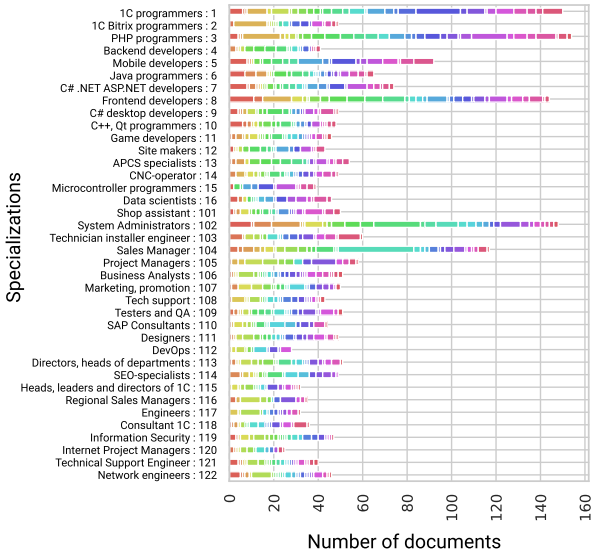


**Fig. 3.** K-means BERT clustering results matching with specialization. The colors show the predicted cluster.

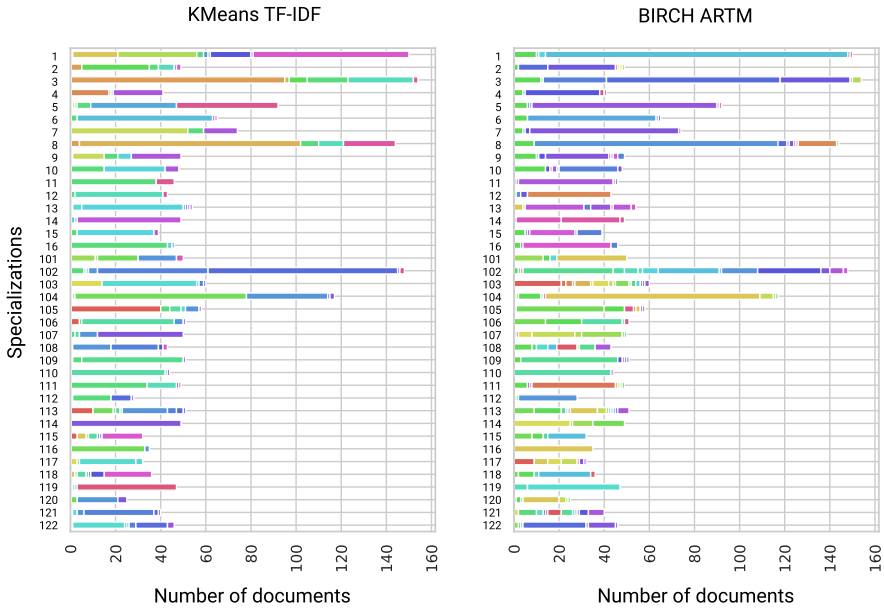


**Fig. 4.** Clustering results matching with specialization. On the left K-means with TF-IDF, on the right BIRCH with artm. The colors show the predicted cluster.

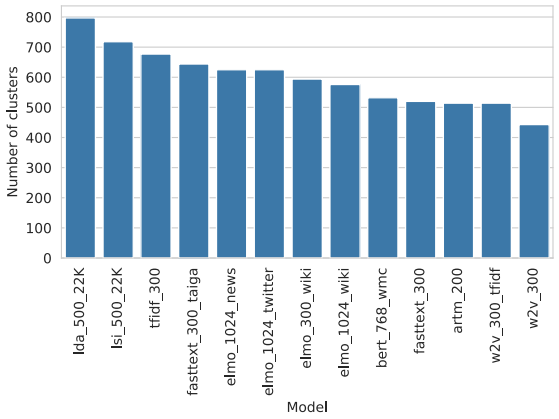


**Fig. 5.** Number of clusters divided on Affinity Propagation for different vector representations.

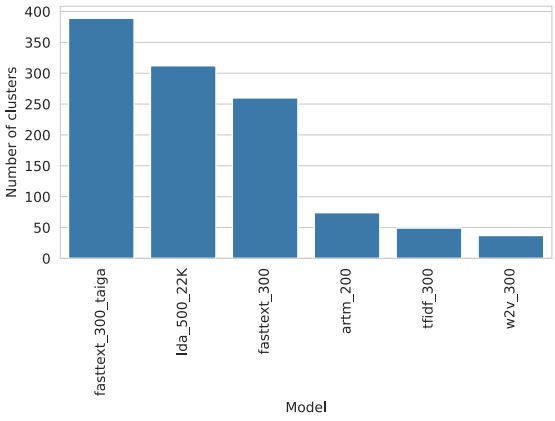


**Fig. 6.** Number of clusters divided on BIRCH for different vector representations.

eration not the geometrical position in space, but semantic similarity between words. This arithmetics corrupts the primary logic of K-means. The coordinates of word2vec are not interpretable, unlike ARTM.

Let us consider in more detail one sample tag - mobile developers. BIRCH + ARTM more frequently coincides with our markup, but K-means on the same model disparts cluster by 2 or more, which indicates a division more specificated on technology, like Android, iOS or Xamarin.

After analysis of performance results, the conclusion can be made, that for analyzing on technology level, the best solution is K-means+ARTM, that has shown 0.7384 on V-measure. However, if the task is set to try to understand the structure of dataset - it is worth to pay attention to word2vec embedding, it has shown 0.6846 V-measure scores on Affinity Propagation, which is the best result among others for this type of clusterization methods.

## 6 Conclusion

During the conducted experiments we tested such statistical methods of vector representation of texts as TF-IDF, probabilistic topic modeling, neural language models based on distributional semantics (word2vec, fasttext), and deep contextualized word representation (ELMo and multilingual BERT). They all went through several types of clustering: K-means, Affinity Propagation, BIRCH and agglomerative clustering.

The considered experiments show that the most interesting and applicable combinations are BIRCH+ARTM and K-means+ARTM, they define the occupation most clearly on utility dataset - texts of a small size with a specific theme. That confirms the effectiveness of additive regularization of topic models in the task of text clustering. Word2vec+Affinity Propagation also performed well. This neural word embedding can be combined with Affinity Propagation due to their orientation on semantic similarity. We have used many pre-trained models with a large vocabulary on different corpora, including multilingual ones, but their results are still much lower than of those learned on our corpora.

In our opinion, TF-IDF is not much inferior in terms of the quality of work to topic modeling, but on methods with an indefinite number of clusters it can degenerate into one cluster. It can be possibly expect by the fact that BIRCH algorithm eliminates "noise". That is why it defines the majority of samples as one "noise" cluster.

Evaluations by methods ARI, AMI and V-measure can be viewed independently and are interchangeable. We aim not only to define metrics, but also to analyze how various clustering methods in combination with different vector representation methods perform partitioning. We concluded that perhaps experts analyzed the professions guided by criteria that were different from those identified by clustering algorithms with an unspecified number of clusters.

Our study can be applied in the preparation of training courses, defining employer requests and also for various kinds of automated analyzers of the labor

market. It can also be in a point of interest for users looking for a work in a particular position, based on their skills.

In the future, we plan to expand the study towards ARTM tuning and tuning of neural word embeddings, and also research neighboring method Word network topic model [19].

## References


1. Ikudo, Akina, et al. *Occupational Classifications: A Machine Learning Approach.* No. w24951. National Bureau of Economic Research, 2018.
2. Boselli, Roberto, et al. "Using machine learning for labour market intelligence." *Joint European Conference on Machine Learning and Knowledge Discovery in Databases.* Springer, Cham, 2017.
3. Colombo, Emilio, Fabio Mercorio, and Mario Mezzanzanica. "Applying machine learning tools on web vacancies for labour market and skill analysis." (2018).
4. Wowczko, Izabela. "Skills and vacancy analysis with data mining techniques." *Informatics.* Vol. 2. No. 4. Multidisciplinary Digital Publishing Institute, 2015.
5. Spirin, Nikita, and Karrie Karahalios. "Unsupervised approach to generate informative structured snippets for job search engines." *Proceedings of the 22nd International Conference on World Wide Web.* ACM, 2013.
6. Muthyala, Rohit, et al. "Data-driven Job Search Engine Using Skills and Company Attribute Filters." 2017 *IEEE International Conference on Data Mining Workshops (ICDMW).* IEEE, 2017.
7. Deokar, Sanjivani Tushar. "Text documents clustering using k means algorithm." *International Journal of Technology and Engineering Science [IJTES]* 1.4 (2013): 282-286.
8. Zhu, Yan, Jian Yu, and Caiyan Jia. "Initializing k-means clustering using affinity propagation." *2009 Ninth International Conference on Hybrid Intelligent Systems.* Vol. 1. IEEE, 2009.
9. Guan, Renchu, et al. "Text clustering with seeds affinity propagation." *IEEE Transactions on Knowledge and Data Engineering* 23.4 (2011): 627-637.
10. Gencoglu, Oguzhan. "Deep Representation Learning for Clustering of Health Tweets." *arXiv preprint arXiv:1901.00439* (2018).
11. Parhomenko P.A., Grigorev A.A., Astrakhantsev N.A. A survey and an experimental comparison of methods for text clustering: application to scientific articles. Trudy ISP RAN/Proc. ISP RAS, 2017, vol. 29, issue 2, pp. 161-200
12. Chen, Juntian, Yubo Tao, and Hai Lin. "Visual exploration and comparison of word embeddings." *Journal of Visual Languages & Computing* 48 (2018): 178-186.
13. Naili, Marwa, Anja Habacha Chaibi, and Henda Hajjami Ben Ghezala. "Comparative study of word embedding methods in topic segmentation." *Procedia computer science* 112 (2017): 340-349.
14. Vorontsov, Konstantin, and Anna Potapenko. "Tutorial on probabilistic topic modeling: Additive regularization for stochastic matrix factorization." *International Conference on Analysis of Images, Social Networks and Texts.* Springer, Cham, 2014.
15. Vorontsov K. V., Potapenko A. A. Additive regularization of topic models // Machine Learning. 2015. Vol. 101, no. 1. P. 303323
16. Kutuzov, Andrey, and Elizaveta Kuzmenko. "WebVectors: a toolkit for building web interfaces for vector semantic models." *International Conference on Analysis of Images, Social Networks and Texts.* Springer, Cham, 2016.

17. Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).
18. Peters, Matthew E., et al. "Deep contextualized word representations." *arXiv preprint arXiv:1802.05365* (2018).
19. Zuo, Yuan, Jichang Zhao, and Ke Xu. "Word network topic model: a simple but general solution for short and imbalanced texts." *Knowledge and Information Systems* 48.2 (2016): 379-398.