e-ISSN 1607-3274 Радіоелектроніка, інформатика, управління. 2020. № 4
p-ISSN 2313-688X Radio Electronics, Computer Science, Control. 2020. № 4

UDC 004.82

MULTITOPIC TEXT CLUSTERING AND CLUSTER LABELING USING CONTEXTUALIZED WORD EMBEDDINGS

Ostapiuk Z. V. – Postgraduate student of the Department of Software Engineering, Lviv Polytechnic National University, Lviv, Ukraine.

Korotyeyeva T. O. – PhD, Associate Professor of the Department of Software Engineering, Lviv Polytechnic National University, Lviv, Ukraine.

ABSTRACT

Context. In the current information era, the problem of analyzing large volumes of unlabeled textual data and its further grouping with respect to the semantic similarity between texts is emerging. This raises the need for robust text analysis algorithms, namely, clustering and extraction of key data from texts. Despite recent progress in the field of natural language processing, new neural methods lack interpretability when used for unsupervised tasks, whereas traditional distributed semantics and word counting techniques tend to disregard contextual information.

Objective. The objective of the study is to develop an interpretable text clustering and cluster labeling methods with respect to the semantic similarity that require no additional training on the user's dataset.

Method. To approach the task of text clustering, we incorporate deep contextualized word embeddings and analyze their evolution through layers of pretrained transformer models. Given word embeddings, we look for similar tokens across all corpus and form topics that are present in multiple sentences. We merge topics so that sentences that share many topics are assigned to one cluster. One sentence can contain a few topics, it can be present in more then one cluster simultaneously. Similarly, to generate labels for the existing cluster, we use token embeddings to order them based on how much they are descriptive of the cluster. To do so, we propose a novel metric – token rank measure and evaluate two other metrics.

Results. A new unsupervised text clustering approach was described and implemented. It is capable of assigning a text to different clusters based on semantic similarity to other texts in the group. A keyword extraction approach was developed and applied in both text clustering and cluster labeling tasks. Obtained clusters are annotated and can be interpreted through the terms that formed the clusters.

Conclusions. Evaluation on different datasets demonstrated applicability, relevance, and interpretability of the obtained results. The advantages and possible improvements to the proposed methods were described. Recommendations for using methods were provided, as well as possible modifications.

KEYWORDS: NLP, word embedding, text clustering, cluster labeling, BERT, keyword extraction, semantic similarity.

ABBREVIATIONS

AI is artificial intelligence;

BERT is Bidirectional Encoder Representations from Transformers;

BoW is Bag-of-Words;

CBOW is Continuous Bag-of-Words;

LDA is Latent Dirichlet Allocation;

NLP is natural language processing;

NN is a neural network;

STS is semantic textual similarity;

TF-IDF is a term frequency-inverse document frequency.

NOMENCLATURE

i^*, j^* are global indices of tokens from all sentences;

$tf_{i,j}$ is a *i*-th term's frequency in the *j*-th document;

df_i is a document frequency of *i*-th token;

N is a total number of documents;

N_t is a total number of tokens across all sentences;

i is a token index in the sentence;

j is a sentence index;

k is a transformer model layer index;

N_l is a total number of layers in the NN;

$e_{i^*}^k$ is an embedding from the *k*-th layer of the *i*-th token;

$stability_{t_{i^*}}$ is a *i*-th token's stability coefficient;

$stability_j^{th}$ is a threshold for stability coefficient in the *j*-th sentence;

N_s is a number of sentences in the dataset;

N_t^j is a number of tokens in the *j*-th sentence;

$e_{j,i}$ is a *i*-th token's embedding from the *j*-th sentence;

$e_{j,i}^k$ is an embedding for the *i*-th token from the *j*-th sentence from *k*-th layer;

s_j is a *j*-th sentence;

$t_{j,i}$ is a *i*-th token from the *j*-th sentence;

$\left|e_{j,i}\right|$ is a length of embedding;

N_{tc} is a total number of topics in the dataset;

tc_t is a *t*-th topic cluster;

S_t is a set of sentence indices in the *t*-th topic cluster;

T_t is a set of tokens in the *t*-th topic cluster;

$j_{t1,t2}$ is a Jaccard similarity between a set of sentence indices in *t*1-st and *t*2-nd topic clusters;

JM is a square similarity matrix build from Jaccard similarity values;

DOI 10.15588/1607-3274-2020-4-10

z is a z-th cluster with sentences;

N_{sc} is a total number of sentence clusters;

SM is a square similarity matrix build from cosine similarity values;

SV_{i^*} is a similarity vector for the i-th token;

s_{j^*} is an index of sentence for the i-th token;

$\overline{SV_{i^*}}$ is an average for a similarity vector SV_{i^*};

S_{i^*} is a variance of similarity vector values SV_{i^*};

$rank_{i^*}$ is a rank of the i-th token.

INTRODUCTION

The problem of dealing with textual data has been a subject of interest for many years now. The need for robust and efficient approaches is growing, as the amount of reviews, feedbacks, posts, and other texts that need to be addressed is getting bigger. Since this kind of data is often unlabeled, the most urgent tasks to solve are clustering and keywords extraction. More often than not, collections of texts are not labeled or require many human-hours to label and those labels are still subjective.

One of the most obvious applications of text clustering is the processing of user feedbacks or reviews. This problem is actual nowadays, and according to the latest surveys [1] corporations are planning to start using AI at the start of this decade to address many customer-specific issues. The research [1] also shows that NLP is the most needed field for companies because they lack solutions for dealing with written communication.

The main challenge of processing texts by a computer is obtaining their numerical representations. This process, also called text embedding, is often performed through word-counting techniques that represent sentence as bag-of-word. This approach, although proven effective when dealing with supervised machine learning tasks, is conceptually unable to capture language semantics. Thus, using such text representation can result in mediocre clustering solutions.

Another, more advanced approach, is using pretrained (or training from scratch) NNs. Novel large models are pretrained on huge amounts of text with specific learning tasks (self-supervised approach). This means that they can be used with small unlabeled datasets – something that was hardly possible with models that need to be trained from scratch.

While they achieve state-of-the-art results, it is still an open issue how to cluster texts using output embeddings. This problem has two aspects: how to pool embedding for each token to capture sentence meaning, and how to interpret these embeddings. Even though it is agreed that deep contextualized embeddings can be good input features for clustering, we still have no means of interpreting these clusters – why the model thinks these texts belong together.

The object of study is the process of grouping texts by their semantic similarity.

The subject of study are algorithms for textual data comparison using contextualized embeddings from pretrained transformer models.

The purpose of the work is to develop an approach for texts clustering with respect to the semantic similarity between them. The study is also focused on the extraction of keywords from texts to be used for topic finding and further comparison. Another aim is to address the interpretability of resulting clusters and to develop cluster labeling technique.

1 PROBLEM STATEMENT

In the corpus of N_s sentences, we need to find N_{sc} sentence clusters. Each sentence $s_j, j = \overline{1, N_s}$ consists of N_t^j tokens $t_{j,i}$ each of which has embedding $e_{j,i}^k$ assigned from the k-th layer of the transformer model. One sentence cluster should be defined by one or more topics tc_t, $t = \overline{1, N_{tc}}$. To be interpretable, the topic tc_t should consist of the set of related tokens T_t and set of respective sentence indices S_t in which tokens in T_t are found. Given all topics, tc_t, $t = \overline{1, N_{tc}}$ we need to merge them into final N_{sc} clusters that contain sentences that share the most topics.

For the task of cluster labeling, a dataset with N_s sentences is given. Each of the sentences contains N_t^j tokens $t_{j,i}$ with embedding $e_{j,i}^k$ taken from the k-th layer of the transformer model. The output should consist of a list of cluster labels t_{i^*} – tokens or their lemmas/dictionary form.

2 REVIEW OF THE LITERATURE

In order to compare two pieces of text one needs to convert them into fixed-size numerical vectors. One of the most popular algorithms for this purpose is bag-of-words. It is used in a wide range of machine learning tasks [2]. BoW vectors consist of elements that denote the number of occurrences of some term in a text. While BoW can be successfully used in supervised tasks, it fails to capture semantics and therefore performs poorly in unsupervised tasks like clustering or keywords extraction. A study [3] was performed to overcome the hard-mapping consequences of BoW.

Another slightly improved metric for representing text in vector space is called TF-IDF. The assumption behind the TF-IDF formula:

$$idf_i = \log\left(\frac{N}{df_i}\right),$$
$$TFIDF = tf_{i,j} idf_i, j = \overline{1, N}. \quad (1)$$

DOI 10.15588/1607-3274-2020-4-10

is that words that occur frequently across all documents are not significant and their importance needs to be reduced (inverse document frequency). On the other hand, words that occur frequently in some documents but are mostly absent in others are considered to be more important (term frequency).TF-IDF is also used to extract important keywords from text based on the previously described assumptions [4].

Both BoW and TF-IDF vectors suffer from high-dimensionality and extreme sparsity. Moreover, words that comprise input texts are represented in a one-hot encoding manner, making them orthogonal to each other. To overcome these and other problems, a Word2Vec model was presented in 2013 by Mikolov et al. [5, 6]. It essentially performs dimensionality reduction. This model allows obtaining vectors that can be compared in embedding space and yield semantically justified similarity scores. However, this and other conceptually similar models preserve no contextual information, and for one word there is always single embedding.

To obtain semantic similarity of variable length sentences or other language structures, information from all words needs to be combined/pooled. Usually, to compare two documents some similarity measure between averaged embeddings is used, but the results are often mediocre. An alternative similarity measure using matrix norms was proposed and applied to targeted marketing tasks [7]. This approach uses pair-wise similarity matrices which leads to $n(n-1)/2$ operations of similarity calculation.

Different pooling techniques are used to create one embedding for the text from its words, one of the most successful is simple averaging with weighting [8]. Nonetheless, the order of the words is not taken into account.

To create embeddings for a document, regardless of its length, Doc2Vec was developed by Le and Mikolov [9]. The disadvantage of Doc2Vec is that document vectors are hard to interpret. To obtain good results, one needs to train this model with their dataset, which is a limitation for many relatively small collections.

Another approach to cluster documents is topic modeling. The most popular is Latent Dirichlet Allocation, LDA [10]. These generative models are often used for data exploration [11]. There are few popular topic modeling algorithms [12], but they all share one underlying assumption that the document consists of a fixed number of topics and each topic is defined by a collection of words. An advantage of topic modeling approaches over traditional clustering is that distribution of topics over documents is obtained. Thus, we can assign one document to different topics with some probability.

In 2017 Vaswani et al. presented the revolutionary transformer model [13]. Based on transformers, a BERT model was introduced in 2018 by Devlin et al. [14]. BERT was pretrained on a massive data set. The self-attention mechanism allows BERT to produce contextualized word embedding and perform word sense disambiguation, solving the polysemy issue.

Although BERT, when first presented, beat many benchmarks, it was still an open problem on how to obtain good and robust sentence-level embeddings. Among BERT's training tasks was next sentence prediction objective, so it is possible to feed into the model two texts and perform sentence regression like STS. However, a computational overhead is caused by $n(n-1)/2$ evaluations.

There are many possible pooling strategies, some of which were described in the original paper [14]. Jawahar et al. proved that different layers encode different linguistic properties of the English language [15], so pooling can be performed not only on the last embeddings.

Sentence-BERT model was developed in 2019 by Reimers and Gurevych [16] and is responsible for the state-of-the-art performance on STS benchmark [17]. The authors used siamese and triplet NNs to fine-tune BERT. As a result, the SBERT model can be used to obtain a single vector per sentence.

A study performed by Wang and Kuo in 2020 [18] brought insights into the evolution of word embeddings through layers of BERT-based models. Authors developed the SBERT-WK model for sentence embedding and presented new metrics that integrate data from all layers – alignment, novelty, and word importance measures. This study showed that BERT-like models can bring insights into textual data not only through embeddings but through their interlayer patterns.

As for unsupervised keywords extraction, a popular RAKE algorithm [19] can extract key phrases and rank them. Also, it can be used to generate adequate stopword list if enough data is provided. A recent algorithm YAKE [20] outperforms many state-of-the-art alternatives. Both are quick and robust, however, operate on statistical metrics and co-occurrence data of words that can be a limitation as keywords cannot be compared using embeddings. Unlike cross-corpus TF-IDF, RAKE and YAKE applicability to cluster labeling is also limited. It requires additional means of selecting keywords from all documents that can be ranked as cluster labels.

3 MATERIALS AND METHODS

Methods, described in this paper, rely on the novel token metric – stability coefficient. The stability coefficient is aimed at capturing the degree of token embedding adjustment to the semantics of the sentence.

SBERT [16] averages last layer embeddings $e_{i^*}^{N_l}$, $i^* = \overline{1,n}$ to obtain sentence vector. Unlike the BERT model's final embedding, vectors from SBERT are very similar to each other. This can be explained by fine-tuning details of SBERT, specifically by the objective function that aimed at minimization of the cosine similarity

$$\cos\left(e_{i^*}, e_{j^*}\right) = \frac{e_{i^*} \cdot e_{j^*}}{\left|e_{i^*}\right|\left|e_{j^*}\right|} \qquad (2)$$

between semantically similar sentences.

DOI 10.15588/1607-3274-2020-4-10

As a result, the influence of the context makes final SBERT embeddings not usable as a representation for described later methods. We suggest using embedding from the ninth or tenth layer (Fig. 1).

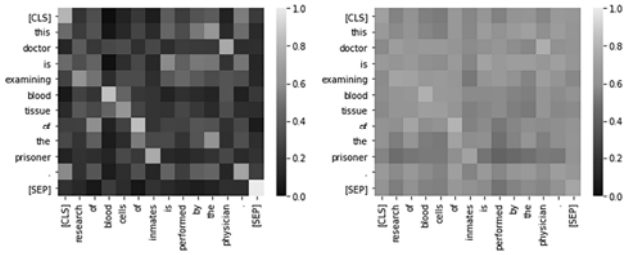

Figure 1 – Pair-wise comparison using cosine similarity between tokens from ninth and last layers respectively

On Fig. 1 there are pairs that have high similarity (lighter shades): “doctor” – “physician”, “prisoner” – “inmates” etc. However, many word-pairs consist of prepositions, punctuation, and other common tokens. To eliminate them we cannot use TF-IDF formula (1) because that would make our approach corpus-dependent and not usable for small datasets.

On Fig. 2 it is shown that common tokens change more (are more “diluted” by other embeddings). The original sentence is “This doctor is examining blood tissue of the prisoner.” In Fig. 1, we demonstrate patterns of interlayer similarity for salient (first row) and common (second row) tokens.

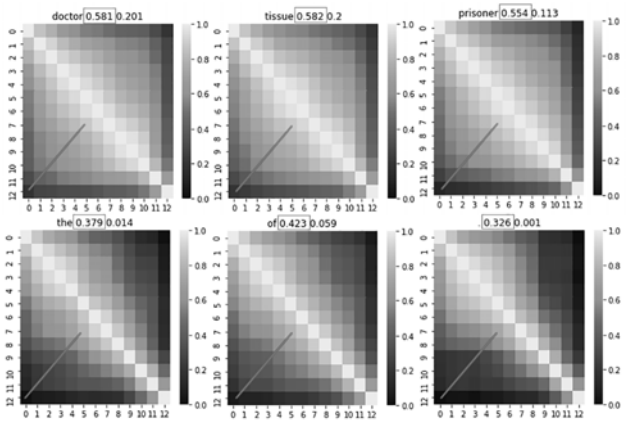

Figure 2 – Heat maps of cosine similarity between embeddings from different layers

To calculate the stability coefficient we sum values from these similarity maps across anti-diagonal starting from the middle as indicated by the line in Fig. 2.

$$stability_{t_{i^*}} = \frac{\left(\sum_{k=0}^{6} \cos\left(e_{i^*}^{k}, e_{N_l - i^*}^{k}\right)\right)}{0.5 N_l} \qquad (3)$$

We normalize the value, thus, the theoretical maximum for the stability coefficient is one. In practice, embeddings are changing rapidly, so for tokens like “.”, “the”, “of” etc. (see Fig. 2) we observe smaller values of stability coefficient (avg. 0.39); for salient tokens, we observe average stability coefficient of 0.56. The threshold $stability_j^{th}$ can be hard-coded, however, we recommend using the average stability coefficient from the sentence.

Two techniques were developed (clustering and cluster labeling) each of which shares the first three steps:

1. Obtain embeddings $e_{j,i}, j = \overline{1, N_s}, i = \overline{1, N_t^j}$ for every token $t_{j,i}$ from sentences s_j. The index of layer $k = \overline{1, N_l}$, from which embeddings are taken, is a hyperparameter. We suggest using the ninth or tenth layer.

2. For each $t_{j,i}$ get $stability_{t_{j,i}}$ using formula (3).

3. For each sentence s_j calculate threshold $stability_j^{th}$ or use hard-coded value. Using this value, remove tokens that satisfy $stability_{t_{j,i}} < stability_j^{th}$.

Our clustering method continues with the following steps (hyperparameter and algorithm choices are described in the section with experiments):

1. Having filtered embeddings $e_{j,i}$ for each token $t_{j,i}$ from each sentence s_j, we need to cluster them using clustering or community detection algorithm of choice. Objects being clustered are high-dimensional embeddings $e_{j,i}$. For BERT and SBERT there are models with $|e_{j,i}| = 768$ (suffix “base”) and $|e_{j,i}| = 1024$ (suffix “large”) dimensions in one embedding.

2. We now have N_{tc} clusters each containing some amount of tokens. These clusters are called topics. We store sentence index *j* with every token embedding $e_{j,i}$, so we can determine what sentences are represented in what clusters. Apart from token embeddings $e_{j,i}$, each topic cluster $tc_t, t = \overline{1, N_{tc}}$ contains a set of respective sentence indices $tc_t = \langle S_t; T_t \rangle$.

3. From each of the N_{tc} clusters we find Jaccard similarity between respective S_t:

$$j_{t1,t2} = j_{t2,t1} = \frac{|S_{t1} \cap S_{t2}|}{|S_{t1} \cup S_{t2}|}. \qquad (4)$$

4. From computed Jaccard similarities matrix is formed $JM_{t1,t2} = j_{t1,t2}$. The *JM* matrix can be interpreted as an adjacency matrix because it contains distance

DOI 10.15588/1607-3274-2020-4-10

measures calculated by the formula (4) between topics $tc_t, t = \overline{1, N_{tc}}$.

5. Merge topics that share the most sentences into one cluster. As a result of the clustering/community detection algorithm of choice, we obtain N_{sc} groups with sentences c_z, $z = \overline{1, N_{sc}}$ that share some topics, therefore can be considered as one cluster inside a particular dataset.

6. Each cluster c_z obtained on the previous step contains a set of sentences and topics tc_t that were merged into c_z. Since topics are essentially collections of tokens, we can obtain labels for the cluster by taking unique tokens from each of the tc_t.

Another method based on the stability coefficient is cluster labeling. For a collection of sentences, we can order tokens by their representative qualities for the current dataset, limit the resulting set, and obtain cluster labels. To do so, we propose the following method (follows described earlier first three steps):

1. Create a square similarity matrix $SM_{i^*,j^*} = \cos\left(e_{i^*}, e_{j^*}\right), i^*, j^* = \overline{1, N_t}$. It consists of cosine similarity values between each token embedding $e_{i^*}, i^* = \overline{1, N_t}$.

2. For each token t_{i^*} extract similarity vector SV_{i^*}:

$$SV_{i^*} = \left\{ SM_{i^*,j^*} \mid 1 \le j^* \le N_t \wedge s_{j^*} \ne s_{i^*} \right\}. \quad (5)$$

The vector SV_i is obtained by taking the *i*-th row of *SM* and excluding tokens from the current token's sentence.

3. Obtain a single value metric from the vector of similarities SV_{i^*}. We propose the following metrics, each of which is evaluated in the results section and justified in the discussion section:

4.

$$\overline{SV_{i^*}} = \frac{\sum_{j^*} SV_{i^*,j^*}}{\left|SV_{i^*}\right|}; \quad (6)$$

$$S_{i^*} = \frac{\sum_{j^*} \left(SV_{i^*,j^*} - \overline{SV_{i^*}} \right)^2}{\left|SV_{i^*}\right| - 1}; \quad (7)$$

$$rank_{i^*} = \overline{SV_{i^*}}\, S_{i^*}. \quad (8)$$

5. Optionally, lemmatize and take unique terms.

6. Order tokens by the chosen metric and limit results.

4 EXPERIMENTS

We obtain pretrained models from Hugging Face's (a popular NLP library) server [21]. In our experiments, we use PyTorch implementations of SBERT model "bert-base-nli-stsb-mean-tokens" [16]. The letter is fine-tuned on the AllNLI [22] dataset, then on the train set of STS benchmark [17]. SBERT authors claim that this model is specifically well suited for semantic similarity.

To check how our approach performs on real-life data we experiment with widely used NLP dataset Reuters-21578 [23] for text categorization.

First, we perform forward pass of input sentences through the model and calculate stability coefficients using formula (3) for each token from each sentence. We follow the second and third steps as described in the previous section. Since BERT-like models use WordPiece tokenizer [24], some words can be deconstructed into subwords ("cellphone" is split to "cell" and "##phone"). We complete each token if it happens to be a subword (starts with "##") and average respective embeddings from the tenth layer.

For the fourth and fifth steps, to perform grouping we use agglomerative clustering [25]. We use cosine similarity as a metric to calculate distances between clusters. As a linkage algorithm, the "average" method was used. We chose the agglomerative clustering algorithm because it allows setting threshold of similarity metric, unlike other algorithms that require a number of clusters to be specified beforehand. As a threshold, we use the empirical value calculated by multiplying maximum within-cluster distance from the obtained linkage matrix by 0.7 (we use SciPy [26] implementation). As a result of hierarchical clustering, we obtain groups with semantically similar tokens and their sentence indices.

We perform the sixth step as described in the section with methodology. We interpret the *JM* matrix obtained on the seventh step as a weighted adjacency matrix – a matrix whose elements store weights between pairs of topics. We use the Louvain community detection algorithm [27]. In other words, on this step, we merge topics that share many sentences. After this, we take unique tokens from topics as labels for obtained clusters.

To compare our results we implemented LDA topic modeling as in [12]. We used Gensim's [28] implementation of LDA – LdaMulticore with 5 topics. A sentence is assigned to a topic if the respective probability from a topic distribution over texts is higher than 0.5.

To evaluate the cluster labeling technique we follow steps described earlier. Again, we chose the tenth layer as an embedding source.

To compare the performance of our technique we implemented TF-IDF based ranking of n-grams within a cluster of sentences (arguably, the most popular approach among NLP practitioners) as proposed by Shahzad Qaiser and Ramsha Ali [29]. We exclude words that have DF higher than 0.8 and limit the number of features to 10000. We use lemmatization (reducing word inflection) as a preprocessing step implemented in NLTK's WordNetLemmatizer module [30].

5 RESULTS

The results of our clustering technique are evaluated on the synthetic dataset to cover as much cases as possible

DOI 10.15588/1607-3274-2020-4-10

(polysemy, sentence with multiple possible assignments) (Table 1). Our evaluating dataset can be divided into four clusters: “phone”, “man eating bakery”, “doctor in prison”, “space”. The last sentence, however, can be categorized into both “phone” and “space” clusters. Also, some of the sentences from “doctor in prison” mention “phone” related topics.

The final clusters are presented in Table 2. We implemented the LDA method [3, 11] to compare results (Table 3).

To evaluate our cluster labeling technique we use categories from the Reuters-21578 dataset (Table 4) as clusters and calculate average similarity (6), variance (7), and word rank (8) for extracted key tokens using stability coefficient (3).

Table 1 – Extracted topics from sentences

#	Sentence	Topics
1.	I like my phone	{ your, my }; { cell, phone, cellphone }
2.	My phone is not good	{ your, my }; { cell, phone, cellphone }
3.	Your cellphone looks great	{ your, my }; { cell, phone, cellphone }
4.	A man is eating food	{ man, he }; { eating, piece, food }
5.	A man is eating a piece of bread	{ man, he }; { eating, piece, food }; { bread, pasta }
6.	A man is eating pasta	{ man, he }; { eating, piece, food }; { bread, pasta }
7.	He went to prison cell with a cell phone to draw blood cell samples from patients	{ cell, phone, cellphone }; { man, he }; { prison, inmates, cell }; { samples, test, felons, patients, blood }
8.	He went to prison cell with a cellphone to draw blood cell samples from felons	{ cell, phone, cellphone }; { man, he }; { prison, inmates, cell }; { samples, test, felons, patients, blood }
9.	He went to prison cell with an Android to test inmates	{ man, he }; { prison, inmates, cell }; { samples, test, felons, patients, blood }
10.	SpaceX launched astronauts to the moon	{ moon, spacex, astronauts, launched, space, floating, rockets}
11.	The invention of reusable rockets was a key step in commercial space travel	{ moon, spacex, astronauts, launched, space, floating, rockets }; { founder, invention, reusable }
12.	Elon Musk is the founder of SpaceX	{ moon, spacex, astronauts, launched, space, floating, rockets }; { founder, invention, reusable }
13.	My phone is floating in space	{ your, my }; { cell, phone, cellphone }; { moon, spacex, astronauts, launched, space, floating, rockets }

Table 2 – Annotated clusters with texts. Tokens that triggered assignment of the sentence to the current cluster are underlined

#	Sentence clusters’ topics	Sentences
1.	{ my, your }; { cell, phone, cellphone }	I like my phone; My phone is not good; Your cellphone looks great; He went to prison cell with a cell phone to draw blood cell samples from patients; He went to prison cell with a cellphone to draw blood cell samples from felons; My phone is floating in space
2.	{ man, he }	A man is eating food; A man is eating a piece of bread; A man is eating pasta; He went to prison cell with a cell phone to draw blood cell samples from patients; He went to prison cell with a cellphone to draw blood cell samples from felons; He went to prison cell with an Android to test inmates
3.	{ eating, piece, food }; { bread, pasta }	A man is eating food; A man is eating a piece of bread; A man is eating pasta
4.	{ prison, inmates, cell }; { samples, test, felons, patients, blood }	He went to prison cell with a cell phone to draw blood cell samples from patients; He went to prison cell with a cellphone to draw blood cell samples from felons; He went to prison cell with an Android to test inmates
5.	{ moon, spacex, astronauts, launched, space, floating, rockets }; { reusable, founder, invention }	SpaceX launched astronauts to the moon; The invention of reusable rockets was a key step in commercial space travel; Elon Musk is the founder of SpaceX; My phone is floating in space

Table 3 – LDA topics extracted from sentences

Topic (top 10 terms)	Sentences
spacex, moon, launch, astronaut, phone, man, eat, cell, good, space	I like my phone; My phone is not good; SpaceX launched astronauts to the moon;
cell, go, prison, draw, sample, blood, phone, cellphone, patient, felon	I like my phone; My phone is not good; He went to prison cell with a cell phone to draw blood cell samples from patients; He went to prison cell with a cellphone to draw blood cell samples from felons; He went to prison cell with an Android to test inmates;
space, step, invention, travel, commercial, key, reusable, rocket, phone, eat	I like my phone; My phone is not good; The invention of reusable rockets was a key step in commercial space travel;
spacex, bread, elon, eat, piece, man, founder, musk, phone, space	I like my phone; My phone is not good; A man is eating a piece of bread; Elon Musk is the founder of SpaceX;
phone, man, eat, cellphone, great, look, float, pasta, space, food	I like my phone; My phone is not good; Your cellphone looks great; A man is eating food; A man is eating pasta; My phone is floating in space

DOI 10.15588/1607-3274-2020-4-10

Table 4 – Cluster labeling results on Reuters-21578 dataset

Category	Top 10 cluster labels			
	BERT-based ranking			TF-IDF ranking
	Rank	Average similarity	Variance	
tea	tea, export, trade, import, coffee, exporter, cocoa, production, in, soybean	pct, commodities, hazelnut, ctc, competitive, exporter, importer, mln, countertrade, kernels	tea, cocoa, coffee, in, export, corn, and, to, rubber, wheat	said, tea, tonne, trade, mln, export, pct, india, production, countertrade
strategic-metal	smelter, ore, copper, mine, production, uranium, metallurgical, niobium, mineworker, mining	zccm, concentrates, stockpile, stockpiled, seminconductor, storeage, zirconium, alcad, steelworkers, dlrs, cominco	zinc, copper, mine, ore, smelter, uranium, coal, mining, gold, production	said, pct, year, smelter, stockpile, local, mine, contract, last, cominco
housing	unit, pct, home, housing, a, houses, mln, january, rate, to	pct, mln, seasonally, dlrs, completions, resale, unit, rate, insurance, mortgage	january, february, homes, housing, houses, unit, to, a, pct, family	pct, unit, mln, january, start, housing, family, rate, said, fell

6 DISCUSSION

Our clustering approach assigns sentences to clusters based on extracted keywords that form topics. This means that anyone willing to use it can substitute SBERT/BERT embedding with some other word vectors. We, however, use these contextualized embeddings to make use of different meanings of words that depend on the containing sentence. For example, there are topics { cell, phone, cellphone } and { prison, inmates, cell }. Both of them contain the word "cell" but with a different meaning. This allows assigning sentence that contains, for example, "prison cell" to correct cluster with "prison" topic, not the "mobile" one.

The thresholding stability coefficient essentially filters out stopwords, and we end up with many nouns and verbs that can represent clusters in some datasets. Consequently, very detailed, fine-grained clusters are obtained. This property is desirable for low-level analysis of texts but can be overwhelming for bigger datasets. For example, we obtained cluster with topic { man, he }. While it is a valid cluster with sentences that match its topic, usually prepositions are not that interesting as cluster topics. To improve topics, one can use synonym filtering or compute the mean vector from all the embeddings and chose the closest one as a single representation for the topic.

In comparison with popular topic modeling approach LDA, our method is easier to configure (no need to select optimal topic number or iteration number) and it is more interpretable. Moreover, LDA seems to perform poorly on smaller datasets. An advantage of LDA is that it scales better to bigger datasets.

As for cluster labeling, Table 4 shows that all three approaches to ranking cluster labels yielded decent and adequate results. Words that to authors mind are better candidates for cluster labels are underlined. Ideally, key tokens in the cluster must be similar to as many other tokens as possible – this condition is satisfied by summing values of similarity vector SV_{i*} (5) and normalizing the sum (6) – "Average similarity" column in Table 4.

However, intuitively, tokens that best describe cluster should not be a bit similar to every other token but have "spikes" of similarity. Summing across similarity vector SV_i will not be a good measure here because a bit of similarity to other tokens can overcome "spikes" of similarity. Instead, we calculate the variance S_{i*} of similarity vector SV_{i*} values. This metric is used in the column "Variance". To incorporate both values we simply multiply them – "Rank" column.

The results obtained using variance S_{i*} require further cleaning (removing stopwords "in", "and", "to"). Average similarity, as theorized, puts on top tokens that are domain-specific, however, not cluster-specific. Words like "pct", "competitive", "mln" are not stopwords, but they are not representative and descriptive of clusters presented in Table 4. On the other hand, combining two metrics in the "Rank" measure by multiplying them resulted in most descriptive labels.

Labels obtained using TF-IDF formula (1) contain fewer cluster-specific words. As seen from results, TF-IDF labels need more filtering of common words like "said", "fell" that are spread in the dataset because it is taken from the news. In our approach, they are not present in the result.

The conceptual similarity between TF-IDF formula and our ranking approach needs to be pointed out as well. Much like in TF-IDF formula (1), we obtain token rank (8) by multiplying metric that indicates how much "spread" current token is (6) (a conceptual equivalent to term frequency $tf_{i,j}$), and how special it is for the given cluster (7) (an equivalent to inverse document frequency idf_i). The important difference is that rank operates on semantic similarity properties of the token within its sentence and cluster, whereas TF-IDF incorporates word-counting statistical measures.

CONCLUSIONS

In this research, a problem of clustering textual data with respect to semantic similarity was addressed. Proposed solutions deal effectively with small datasets and require no additional training on user's data – a frequently arising limitation of the majority of datasets. The results show that obtained clusters are interpretable and justified by topics that are represented by sentences. The cluster labeling technique proved to be adequate and yield stable results for different dataset sizes.

The scientific novelty of the study is a new text clustering algorithm that assigns text to multiple clusters. The assignment results can be easily interpreted by a human through the respective topics. A novel token property –

DOI 10.15588/1607-3274-2020-4-10

stability coefficient – was developed. The bigger the coefficient – the greater the token stability and its impact on final embeddings.

Also, novel metric – token rank – for tokens in the cluster is proposed and evaluated. Token rank is used in the cluster labeling method and incorporates both how similar token is to all cluster content and how "noticeable" in comparison to common words it is.

The practical significance is that developed methods apply to text mining and analysis of web content. Fine-grain properties of the multitopic clustering method can be successfully leveraged in tasks where one does not have access to a lot of data. Cluster labeling technique can be used for data exploration by engineers to grasp the high-level content of the corpus. It can be used to improve the indexing of textual data by extracting labels and using them for filtering and search.

Proposed methods are extensible and can be used as frameworks. For example, one can use different embedding source, choose different clustering algorithms for topic detection and sentence merging, or extract initial keywords for cluster labeling using different algorithm before applying ranking. By choosing different index of the layer to obtain embeddings from, one can vary the influence of the word context.

The prospects for future study include gaining a more in-depth understanding of which tokens contribute the most to final embeddings by researching self-attention mechanisms in transformer models. Also, additional research is needed to efficiently eliminate low-level topics and rank them according to their relevance to the corpus.

ACKNOWLEDGEMENTS

This work is prepared as a part of the thesis on the text mining approaches for master's degree in computer science. We express appreciation for the technical support to Lviv Polytechnic National University.

REFERENCES

1. Gareiss R., There's Nothing Artificial About It, Nemertes Research, Mokena, IL, Quarterly Rep. DN7575, 2019.
2. Zhang Y., Jin R. and Zhou Z. Understanding bag-of-words model: a statistical framework, *International Journal of Machine Learning and Cybernetics*, Vol. 1, No. 1–4, pp. 43–52, 2010. DOI: 10.1007/s13042-010-0001-0.
3. Zhao R. and Mao K., Fuzzy Bag-of-Words Model for Document Representation, *IEEE Transactions on Fuzzy Systems*, 2018, Vol. 26, No. 2, pp. 794–804, DOI:10.1109/dexa.2010.32.
4. Wartena C., Brussee R. and Slakhorst W. Keyword Extraction Using Word Co-occurrence, in *Proc. of 21st International Conference on Database and Expert Systems Applications (DEXA)*, 2010. DOI: 10.1109/dexa.2010.32.
5. Mikolov T., Chen K., Corrado G. and Dean J. Efficient Estimation of Word Representations in Vector Space, *arXiv*: 1301.3781 [cs.CL], Sep. 2013.
6. Mikolov T., Sutskever I., Chen K., Corrado G. and Dean J. Distributed Representations of Words and Phrases and their Compositionality, *Advances in Neural Information Processing Systems*, 2013, Vol. 26.
7. Brück T. vor der and Pouly M. Text Similarity Estimation Based on Word Embeddings and Matrix Norms for Targeted Marketing, in *Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019, Vol. 1, Minneapolis, USA, pp. 1827–1836. DOI: 10.18653/v1/n19-118.
8. Arora S., Liang Y. and Ma T. A Simple but Tough-to-Beat Baseline for Sentence Embeddings, in *Proc. of 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.
9. Le Q. and Mikolov T. Distributed Representations of Sentences and Documents, in *Proc. of 31st International Conference on International Conference on Machine Learning (ICML)*, Beijing, China, 2014, pp. 1188–1196.
10. Blei D., Ng A. and Jordan M. "Latent dirichlet allocation", *The Journal of Machine Learning Research*, 2003, Vol. 3, No. 1, pp. 993–1022.
11. Tong Z. and Zhang H. A Text Mining Research Based on LDA Topic Modelling, *Computer Science & Information Technology (CS & IT)*, 2016. DOI: 10.5121/csit.2016.60616.
12. Alghamdi R. and Alfalqi K. A Survey of Topic Modeling in Text Mining, *International Journal of Advanced Computer Science and Applications*, 2015, Vol. 6, No. 1. DOI: 10.14569/ijacsa.2015.060121.
13. Vaswani A. et al. Attention is all you need, in *Proc. of 31st International Conference on Neural Information Processing Systems (NIPS)*, Long Beach, USA, 2017, pp. 6000-6010.
14. Devlin J., Chang M., Lee K. and Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in *Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Minneapolis, USA, 2019, pp. 4171–4186. DOI: 10.18653/v1/n19-1423.
15. Jawahar G., Sagot B. and Seddah D. What Does BERT Learn about the Structure of Language?, in *Proc. of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, Florence, Italy, 2019, pp. 3651–3657. DOI: 10.18653/v1/p19-1356.
16. Reimers N. and Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, in *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, 2019, pp. 3982–3992. DOI: 10.18653/v1/d19-1410.
17. Cer D., Diab M., Agirre E., Lopez-Gazpio I. and Specia L. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation, in *Proc. of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, Vancouver, Canada, 2017, pp. 1–14. DOI: 10.18653/v1/s17-2001.
18. Wang B. and Kuo C. SBERT-WK: A Sentence Embedding Method by Dissecting BERT-Based Word Models, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2020, Vol. 28, pp. 2146–2157. DOI: 10.1109/taslp.2020.3008390.
19. Rose S., Engel D., Cramer N. and Cowley W. Automatic Keyword Extraction from Individual Documents, *Text Mining*, 2010, pp. 1–20, DOI: 10.1002/9780470689646.ch1.
20. Campos R., Mangaravite V., Pasquali A., Jorge A., Nunes C. and Jatowt A. YAKE! Keyword extraction from single documents using multiple local features, *Information*

DOI 10.15588/1607-3274-2020-4-10

Sciences, 2020, Vol. 509, pp. 257–289, DOI: 10.1016/j.ins.2019.09.013.
21. Wolf T. et al, HuggingFace's Transformers: State-of-the-art Natural Language Processing *arXiv*: 1910.03771 [cs.CL], Jul. 2020.
22. Bowman S., Angeli G., Potts C. and Manning C. A large annotated corpus for learning natural language inference, in *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Lisbon, Portugal, 2015, pp. 632–642, 2015. DOI: 10.18653/v1/d15-1075.
23. Lewis D., Yang Y., Rose T. and Li F. RCV1: A New Benchmark Collection for Text Categorization Research, *Journal of Machine Learning Research*, 2004, Vol. 5, No. 5, pp. 361–397.
24. Wu Y. et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, *arXiv*: 1609.08144 [cs.CL], Oct, 2016.
25. Ackermann M., Blömer J., Kuntze D. and Sohler C., Analysis of Agglomerative Clustering, *Algorithmica*, Vol. 69, No. 1, pp. 184–215, 2012. DOI: 10.1007/s00453-012-9717-4.
26. Virtanen P. et al., SciPy 1.0: fundamental algorithms for scientific computing in Python, *Nature Methods*, Vol. 17, No. 3, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.
27. Blondel V., Guillaume J., Lambiotte R. and Lefebvre E., Fast unfolding of communities in large networks, *Journal of Statistical Mechanics: Theory and Experiment*, 2008. Vol. 2008, No. 10, p. P10008, DOI: 10.1088/1742-5468/2008/10/p10008.
28. Řehůřek R. and Sojka P. Software Framework for Topic Modelling with Large Corpora, in *Proc. of the 7th Conference on Language Resources and Evaluation (LREC)*, Valletta, Malta, 2010, pp. 45–50. DOI: 10.13140/2.1.2393.1847.
29. Qaiser S. and Ali R. Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents, *International Journal of Computer Applications*, Vol. 181, No. 1, pp. 25–29, 2018. Available: 10.5120/ijca2018917395.
30. Loper E. and Bird S. "NLTK", in *Proc. of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics*, Philadelphia, USA, 2002, pp. 63–70. DOI: 10.3115/1118108.1118117.

Received 02.09.2020.
Accepted 27.10.2020.

УДК 004.82

КЛАСТЕРИЗАЦІЯ ТЕКСТІВ ІЗ ВИОКРЕМЛЕННЯМ ТЕМ ТА АНОТАЦІЯ КЛАСТЕРІВ ЗА ДОПОМОГОЮ КОНТЕКСТУАЛЬНИХ СЛІВ-ВЕКТОРІВ

Остапюк З. В. – магістрант, кафедра програмного забезпечення, Національний університет «Львівська політехніка», Львів, Україна.

Коротєєва Т. О. – канд. техн. наук, доцент кафедри програмного забезпечення, Національний університет «Львівська політехніка», Львів, Україна.

АНОТАЦІЯ

Актуальність. У сучасній інформаційній ері все частіше виникає проблема аналізу великих обсягів текстових даних та їх групування із урахуванням семантичної схожості. Як результат, збільшується необхідність в надійних алгоритмах аналізу тексту, а саме – для кластеризації та виокремлення ключових слів із текстових даних. Незважаючи на недавній прогрес у галузі опрацювання мови, результати нових нейронних методів складно інтерпретувати при використанні для завдання кластеризації, тоді як традиційні методи розподіленої семантики та підрахунку слів, як правило, не враховують контекстну інформацію.

Метою роботи є розробити методи кластеризації тексту, результати яких можна легко інтерпретувати, та анотації кластерів із врахуванням семантичної подібності, які не потребують додаткового навчання на наборах даних користувача.

Метод. Щоб вирішити завдання кластеризації тексту, ми використовуємо контекстуалізовані слова-вектори та аналізуємо їх еволюцію між шарами попередньо натренованих моделей трансформерів. Ми шукаємо схожі лексеми у всьому корпусі за допомогою слів-векторів та формуємо теми, які можуть бути присутні у кількох реченнях. Ми об'єднуємо теми так, що речення, які поділяють багато тем, присвоюються одному кластеру. Оскільки одне речення може містити декілька тем, воно може бути присутнім у кількох кластерах одночасно. Аналогічно, для створення анотацій для існуючого кластера ми використовуємо слова-вектори, щоб упорядкувати слова залежно від того, наскільки добре вони описують кластер. Для цього ми пропонуємо нову міру відповідності кластеру – ранг слова.

Результати. Описано та реалізовано новий підхід кластеризації тексту. Він здатний віднести один текст до одного та більше кластерів на основі семантичної подібності з іншими текстами групи. Розроблено та застосовано підхід до виокремлення ключових слів як для кластеризації тексту, так і для завдання анотації кластерів. Отримані кластери анотовані та можуть бути інтерпретовані через терміни, з яких сформовані відповідні теми.

Висновки. Оцінка на різних наборах даних продемонструвала застосовність, відповідність та легкість інтерпретації отриманих результатів. Описано переваги та можливості вдосконалення запропонованих методів. Були надані рекомендації щодо використання методів, а також можливі їх модифікації.

КЛЮЧОВІ СЛОВА: NLP, слова-вектори, кластеризація тексту, анотування кластерів, BERT, виокремлення ключових слів, семантична схожість.

DOI 10.15588/1607-3274-2020-4-10

УДК 004.82

КЛАСТЕРИЗАЦИЯ ТЕКСТОВ ИЗ ИЗВЛЕЧЕНИЕМ ТЕМ И АННОТАЦИЯ КЛАСТЕРОВ ПРИ ПОМОЩИ КОНТЕКСТУАЛЬНИХ СЛОВ-ВЕКТОРОВ

Остапюк З. В. – магистрант, кафедра программного обеспечения, Национальный университет «Львовская политехника», Львов, Украина.

Коротеева Т. О. – канд. техн. наук, доцент кафедры программного обеспечения, Национальный университет «Львовская политехника», Львов, Украина.

АННОТАЦИЯ

Актуальность. В современную информационную эру возникает проблема анализа больших объемов текстовых данных и их дальнейшего группирования на основе семантического сходства. В результате растёт потребность в надежных алгоритмах анализа текста, а именно – кластеризации и извлечении ключевых данных из текстов. Несмотря на недавний прогресс в области обработки текста, новым нейронным методам не хватает интерпретируемости при использовании в задачах кластеризации, тогда как традиционные методы распределенной семантики и подсчета слов, как правило, игнорируют контекстную информацию.

Целью исследования является разработка интерпретируемых методов кластеризации текста и аннотации кластеров с учетом семантического сходства, которые не требуют дополнительного обучения на пользовательском наборе данных.

Метод. Чтобы решить задачу кластеризации текста, мы используем контекстуализированные слова-векторы и анализируем их эволюцию через слои предварительно обученных моделей трансформеров. С помощью слов-векторов мы ищем похожие токены во всем корпусе и формируем темы, которые могут присутствовать в нескольких предложениях. Мы объединяем темы так, чтобы предложения, которые разделяют многие темы, были отнесены к одному кластеру. Поскольку одно предложение может содержать несколько тем, оно может присутствовать в нескольких кластерах одновременно. Аналогичным образом, чтобы генерировать аннотации для существующего кластера, мы используем слова-векторы, и упорядочиваем слова их в зависимости от того, насколько хорошо они описывают кластер. Для этого мы предлагаем новую меру соответствия кластеру – ранг слова.

Результаты. Был описан и реализован новый подход к кластеризации текста. Он может относить текст к разным кластерам на основе семантического сходства с другими текстами в группе. Подход с извлечением ключевых слов был разработан и применен как в задачах кластеризации текста, так и в задачах аннотации кластеров. Полученные кластеры содержат аннотирование темы и могут быть интерпретированы через термины, из которых сформированные эти темы.

Выводи. Оценка на разных наборах данных продемонстрировала применимость, соответствие и интерпретируемость полученных результатов. Описаны преимущества и возможные улучшения предложенных методов. Даны рекомендации по использованию методов, а также возможные модификации.

КЛЮЧЕВЫЕ СЛОВА: NLP, слова-векторы, кластеризация текста, аннотация кластеров, BERT, извлечения ключевых слов, семантическая схожесть.

ЛІТЕРАТУРА / ЛИТЕРАТУРА

1. There’s Nothing Artificial About It: quarterly report (final): DN7575 / Nemertes; R. Gareiss. – Mokena, 2019. – 12 p.
2. Zhang Y. Understanding bag-of-words model: a statistical framework / Y. Zhang, R. Jin, Z. Zhou // International Journal of Machine Learning and Cybernetics. – 2010. – № 1. – P. 43–52. DOI: 10.1007/s13042-010-0001-0.
3. Zhao R. Fuzzy Bag-of-Words Model for Document Representation / R. Zhao, K. Mao // IEEE Transactions on Fuzzy Systems. – 2017. – Vol. 14, № 8. DOI: 10.1109/TFUZZ.2017.2690222.
4. Wartena C. Keyword Extraction Using Word Co-occurrence / C. Wartena, R. Brussee, W. Slakhorst // 21st International Conference on Database and Expert Systems Applications (DEXA), Bilbao, Spain, August 30 – September 3, 2010: proceedings. – Bilbao: IEEE, 2010. – P. 54–58. DOI: 10.1109/DEXA.2010.32.
5. Efficient Estimation of Word Representations in Vector Space / T. Mikolov, K. Chen, G. Corrado, J. Dean // arXiv. – 2013. – Vol. abs/1301.3781.
6. Distributed Representations of Words and Phrases and their Compositionality / [T. Mikolov, I. Sutskever, K. Chen et al.] // Advances in Neural Information Processing Systems. – 2013. – № 26.
7. Vor der Brück T. Text Similarity Estimation Based on Word Embeddings and Matrix Norms for Targeted Marketing / T. Vor der Brück, M. Pouly // The 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Minneapolis, USA, June, 2019: proceedings. – Minneapolis : ACL, 2019. – Vol. 1. – P. 1827–1836. DOI: 10.18653/v1/N19-1181.
8. Arora S. A Simple but Tough-to-Beat Baseline for Sentence Embeddings / S. Arora, Y. Liang, T. Ma // The 5th International Conference on Learning Representations (ICLR), Toulon, France, April 24–26, 2017: proceedings. – Toulon : OpenReview.net, 2017.
9. Le Q. Distributed Representations of Sentences and Documents / Q. Le, T. Mikolov // The 31st International Conference on Machine Learning (ICML), Beijing, China, 2014: proceedings. – Beijing : PMLR, 2014. – Vol. 32. – P. 1188–1196.
10. Blei D. Latent Dirichlet Allocation / D. Blei, A. Ng, M. Jordan // Journal of Machine Learning Research. – 2003. – Vol. 3. – P. 993–1022. DOI: 10.1162/jmlr.2003.3.4-5.993.
11. Tong Z. A Text Mining Research Based on LDA Topic Modelling / Z. Tong, H. Zhang // The 6th International Conference on Computer Science, Engineering and Information Technology (CSEIT), Vienna, Austria, May 21–22, 2016: proceedings. – Vienna : CSIT, 2016. – P. 201–210. DOI: 10.5121/csit.2016.60616.
12. Alghamdi R. A Survey of Topic Modeling in Text Mining / R. Alghamdi, K. Alfalqi // International Journal of Advanced Computer Science and Applications. – 2015. –

DOI 10.15588/1607-3274-2020-4-10

Vol. 6, № 1. – P. 147–153. DOI: 10.14569/IJACSA.2015.060121.
13. Attention Is All You Need / [A. Vaswani, N. Shazeer, N. Parmar et al.] // The 31st International Conference on Neural Information Processing Systems (NIPS), Long Beach, USA, December 4–9, 2017: proceedings. – Long Beach : Curran Associates Inc., 2017. – P. 6000–6010.
14. BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding / [J. Devlin, M. Chang, K. Lee, K. Toutanova] // The 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Minneapolis, USA, June 2–7, 2019: proceedings. – Minneapolis: ACL, 2019. – P. 4171–4186. DOI:10.18653/v1/N19-1423.
15. Jawahar G. What does BERT learn about the structure of language? / G. Jawahar, B. Sagot, D. Seddah // The 57th Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy, July 28 – August 2, 2019: proceedings. – Florence : ACL, 2019. – P. 3651–3657. DOI: 10.18653/v1/P19-1356.
16. Reimer N. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks / N. Reimer, I. Gurevych // The 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, November 3–7, 2019: proceedings. – Hong Kong : ACL, 2019. – P. 3982–3992. DOI: 10.18653/v1/D19-1410.
17. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation / [D. Cer, M. Diab, E. Agirre et al.] // The 11th International Workshop on Semantic Evaluation (SemEval-2017), Vancouver, Canada, August, 2017: proceedings. – Vancouver: ACL, 2017. – P. 1–14. DOI: 10.18653/v1/S17-2001.
18. Wang B. SBERT-WK: A Sentence Embedding Method by Dissecting BERT-based Word Models / B. Wang, J. Kuo // IEEE/ACM Transactions on Audio, Speech, and Language Processing. – 2020. – Vol 28. – P. 2146–2157. https://arxiv.org/pdf/2002.06652.pdf DOI: 10.1109/TASLP.2020.3008390.
19. Automatic Keyword Extraction from Individual Documents / [S. Rose, D. Engel, N. Cramer, W. Cowley] // Text Mining: Applications and Theory. – Padstow : John Wiley & Sons, 2010. – (Mathematics). – P. 1–20. DOI: 10.1002/9780470689646.ch1.
20. YAKE! Keyword Extraction from Single Documents using Multiple Local Features / [R. Campos, V. Mangaravite, A. Pasquali et al.] // Information Sciences. – 2020. – № 509. – P. 257–289. DOI: 10.1016/j.ins.2019.09.013.
21. HuggingFace's Transformers: State-of-the-art Natural Language Processing / [T. Wolf, L. Debut, V. Sanh et al.] // arXiv. – 2019. – Vol. abs/1910.03771.
22. A large annotated corpus for learning natural language inference / [S. Bowman, G. Angeli, C. Potts, C. Manning] // The 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), Lisbon, Portugal, September, 2015: proceedings. – Lisbon : ACL, 2015. – P. 632–642. DOI: 10.18653/v1/D15-1075.
23. RCV1: A New Benchmark Collection for Text Categorization Research / [D. Lewis, Y. Yang, T. Rose, F. Li] // Journal of Machine Learning Research. – 2004. – Vol. 5, №5. – P. 361–397.
24. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation / [Y. Wu, M. Schuster, Z. Chen et al.] // arXiv. – 2016. – Vol. abs/1609.08144.
25. Analysis of Agglomerative Clustering / M. Ackermann, J. Blomer, D. Kuntze, D. Sohler // Algorithmica. – 2012. – Vol. 69, № 1. – P. 184–215. DOI: 10.1007/s00453-012-9717-4.
26. SciPy 1.0: fundamental algorithms for scientific computing in Python / [P. Virtanen, R. Gommers, T. Oliphant et al.] // Nature Methods. – 2020. – Vol 17. – P. 261–272. DOI: 10.1038/s41592-019-0686-2.
27. Fast unfolding of communities in large networks / [V. Blondel, J. Guillaume, R. Lambiotte, E. Lefebvre] // Journal of Statistical Mechanics: Theory and Experiment. – 2008. – Vol. 2008, № 10. DOI: 10.1088/1742-5468/2008/10/P10008.
28. Rehurek R. Software Framework for Topic Modelling with Large Corpora / R. Rehurek, P. Sojka // The 7th Conference on Language Resources and Evaluation (LREC), Valletta, Malta, May 22nd, 2010: proceedings. – Valetta: ELRA, 2010. – P. 45–50. DOI: 10.13140/2.1.2393.1847.
29. Qaiser S. Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents / S. Qaiser, R. Ali // International Journal of Computer Applications. – 2018. – Vol. 181, №1. – P. 25–29. DOI: 10.5120/ijca2018917395.
30. Loper E. NLTK: The Natural Language Toolkit / E. Loper, S. Bird // Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, Philadelphia, USA, July, 2002: proceedings. – Philadelphia : ACL, 2002. – P. 63–70. DOI: 10.3115/1118108.1118117.

DOI 10.15588/1607-3274-2020-4-10