Architecture and Implementation of an Intelligent News Analysis System

Yury Gapanyuk, Igor Latkin, Sergey Chernobrovkin, Aleksey Leontiev, Grigory Ozhegov, Alexander Opryshko, and Maxim Myalkin

Bauman Moscow State Technical University, Moscow, Russia,
gapyu@bmstu.ru

Abstract. The paper discusses the developing of information system for analysis and aggregation of news from news servers and social resources. The basic purpose of the system is to provide relevantly aggregated news information for the user. The paper reviewed the used techniques for gathering and storing of news information. The paper discusses the used methods for news processing such as news online clustering and news sentiment analysis. The architecture and implementation principles of developed system are given.

Keywords: online news processing, online clustering, sentiment analysis, metagraph, TF-IDF, word mover's distance, convolutional neural network, support vector machine.

1 Introduction

The job of journalists and editors is to react as quickly as possible to hot news, so they have to examine a lot of publications that appear on the Internet in a short time. Moreover, each event can be covered in many articles and publications, and therefore the amount of data becomes sufficiently large. In addition to the news, there are also social networks where users can express their attitude to certain news, discuss events, come to their "own" conclusion and so on. All this information (news items, articles, social posts) should be read, compared and analyzed by the editors of many news agencies. However, often the editor can't process the whole data stream, so we need to provide a short description of these news, various additional information, using which editor would be able to assess if it can be valuable and interesting, and also to tie discussions in social networks to the news, to provide maximum information, bring various statistical data about the social posts and much more.

To solve these problems, we have developed the information system that is discussed in details in this paper.

2 The system architecture

2.1 Overview

The article [1] considers the news processing system in comparison with existing systems. The system proposed in the article allows editors to view a summary of news articles. The main advantage of the system considered further is that, in addition to news sources, it also processes social networks data. People express their attitude to news in the comments. Sentiment analysis of these comments is one of the advantages of the system.

2.2 The high-level system architecture

The designed system should consist of several independent modules, each of which must solve its own task. Among these tasks, we can distinguish: collecting and storing data, news clustering, selecting text attributes and its characteristics, aggregation of data for the same information event and providing information in a format convenient for the editor.

Thus, it is possible to divide the system into three main subsystems: a collection subsystem, a subsystem for executing text processing tasks and a subsystem for displaying information. The high-level system structure is represented at Fig. 1.

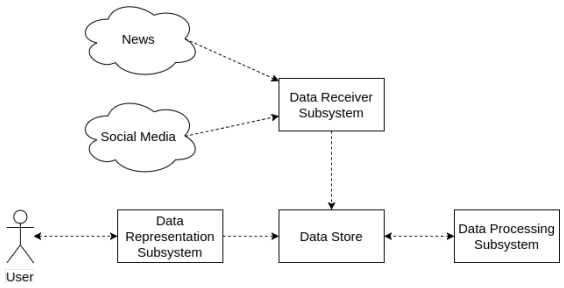


Fig. 1: The high-level system structure.

2.3 Data collection and storage

The workflow of this subsystem is represented in the diagram in Fig. 2.

All collected data from the Web (news articles and social posts and comments) are stored inside document-oriented database MongoDB. After being collected, news are transferred to the queue for the processing subsystem using RabbitMQ as a messaging broker, where task routing is used to define the type of processing for each individual news article.

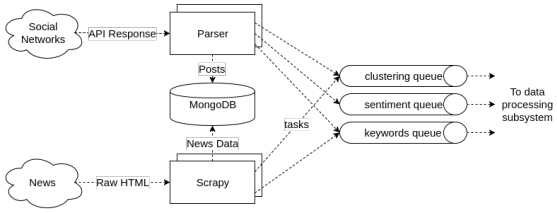


Fig. 2: Diagram of data collection subsystem

2.4 Selecting attributes of text and its characteristics

The word processing system consists of several subsystems, each of which performs its own task. However, all subsystems operate on a common principle.

When the news is received, a task is formed to analyze this news in all the subsystems, but, formally, three tasks are created: the task for clustering, the task for determining the emotional coloring of social posts, and the selection of keywords.

All tasks are resource-intensive, therefore, it is necessary to ensure maximum concurrency in the implementation of all phases in order to reduce the average waiting time for news processing. In production there is a requirement to process not less than 20,000 news per day. So the news processing time must be less than 5 seconds.

To accommodate all these requirements, the processing subsystem architecture, displayed in the Fig. 3 was developed. Each worker processes the news and adds the meta information to the database

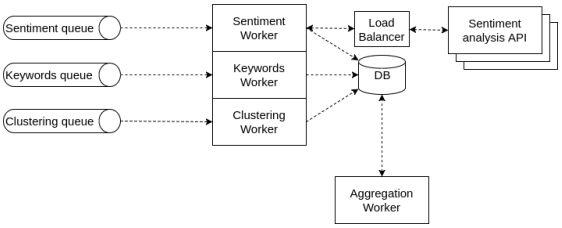


Fig. 3: Processing subsystem scheme

The diagram shows three task queues that are responsible for their own processing types. Tasks then processed by their corresponding handlers.

Due to the fact that task for sentiment analysis is quite resource intensive in terms of using the CPU during training and using RAM both during training and operation of the trained model it was decided to perform a sentiment analysis of text in a separate microservice where the necessary models were placed in

memory, and the interaction of workers with this microservice is carried out through the HTTP API.

In addition to processing each news, it is necessary to identify the aggregated characteristics of the cluster. For this purpose, an aggregator is added to the scheme. Its task is to get updated news clusters and aggregate the characteristics of cluster's elements. Processed clusters are displayed in the editor's interface.

3 The brief description of metagraph approach

In this paper, we will discuss system from two levels point of view. The higher level is a structure description layer using metagraph approach and the lower level is an implementation layer.

According to paper [2] metagraph approach may be considered as a higher-level structural framework for representation of dynamical complex graph structures. In this paper, we have adopted the metagraph approach for the structure of the NLP-processing system representation.

The metagraph is described as follows: $MG = \langle V, MV, E \rangle$, where MG – metagraph; V – set of metagraph vertices; MV – set of metagraph metavertices; E – set of metagraph edges.

Metagraph vertex $v_i = \{atr_k\}, v_i \in V$, where atr_k – attribute. Metagraph edge $e_i = \langle v_S, v_E, \{atr_k\} \rangle, e_i \in E$, where v_S – source vertex (metavertex) of the edge; v_E – destination vertex (metavertex) of the edge; atr_k – attribute.

The metagraph fragment is defined as $MG_i = \{ev_j\}, ev_j \in (V \cup E \cup MV)$, where ev_j – an element that belongs to union of vertices, edges and metavertices. The metagraph metavertex: $mv_i = \langle \{atr_k\}, MG_f \rangle, mv_i \in MV$, where mv_i – metagraph metavertex; atr_k – attribute, MG_f – metagraph fragment.

The metagraph itself is not more than a complex data structure. To process and transform metagraph data the metagraph agents are used. There are two kinds of metagraph agents: the metagraph function agent ag^F and the metagraph rule agent ag^R.

The metagraph function agent serves as a function with input and output parameter in form of metagraph: $ag^F = \langle MG_{IN}, MG_{OUT}, AST \rangle$, where ag^F – metagraph function agent; MG_{IN} – input parameter metagraph; MG_{OUT} – output parameter metagraph; AST – abstract syntax tree of metagraph function agent in form of metagraph.

The metagraph rule agent uses rule-based approach: $ag^R = \langle MG, R, AG^{ST} \rangle, R = \{r_i\}, r_i : MG_j \rightarrow OP^{MG}$, where ag^R – metagraph rule agent; MG – working metagraph, a metagraph on the basis of which the rules of agent are performed; R – set of rules r_i; AG^{ST} – start condition (metagraph fragment for start rule check or start rule); MG_j – a metagraph fragment on the basis of which the rule is performed; OP^{MG} – set of actions performed on metagraph.

The distinguishing feature of metagraph agent is its homoiconicity which means that it can be data structure for itself.

Thus, metagraph approach is a good basis for complex graph structures and their transformations representation. Now we can describe the elements of architecture of modules of proposed system in terms of metagraph approach.

4 The implementation of clustering module

4.1 The metagraph representation of clustering module

The metagraph representation of clustering module is given at Fig. 4

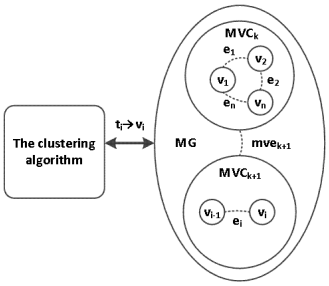


Fig. 4: The metagraph representation of clustering module.

The clustering module may be implemented as a metagraph rule agent which generates the output metagraph MG. The news text messages t_i are transformed into metagraph vertices v_i, which may be annotated with attributes (source text message, auxiliary clustering metrics).

The output metagraph MG contains a set of metavertices-clusters $MVC = \{MVC_k\}$. Each metavertex-cluster contains earlier clustered vertices-messages v_1, v_2, $\ldots$ v_n. Depending on used clustering algorithm vertices-messages may be connected with necessary edges e_1, e_2, $\ldots$ e_n.

Clustering metagraph rule agent may add new metavertex-cluster MVC_{k+1} and then add new vertex-message v_i to this cluster. For new vertex-message v_i necessary edges inside the cluster may be added (edge e_i is shown as an example). For new metavertex-cluster MVC_{k+1} necessary edges also may be added connecting new cluster with other clusters (edge mve_{k+1} is shown as an example).

The rule system of clustering metagraph rule agent contains two rules. When the news text messages t_i arrives then it is transformed into metagraph vertices v_i. Then depending on the current state of the metagraph MG one of the two following rules are performed:

1. $(MG, v_i) \rightarrow Alg(MG, v_i) = MVC_k; MVC_k + v_i; MVC_k + \{e_i\}$
   Using clustering algorithm for new vertex v_i the appropriate metavertex-cluster MVC_k is selected. Then new vertex v_i with necessary edges $\{e_i\}$ are adding to selected cluster.

2. $(MG, v_i) \rightarrow Alg(MG, v_i) = \emptyset; MG+MVC_{k+1}; MG+\{mve_{k+1}\}; MVC_{k+1}+v_i; MVC_{k+1}+\{e_i\}$
   If appropriate metavertex-cluster cannot be selected then new metavertex-cluster MVC_{k+1} and set of necessary edges $\{mve_{k+1}\}$ are adding to metagraph. Then new vertex v_i with necessary edges $\{e_i\}$ are adding to new cluster.

The clustered metagraph structure may be considered as a basis for next stages of text processing. It may be transformed with other metagraph agents.

4.2 Algorithm of the clustering module

The detailed algorithm *Alg* for news clustering module may be represented as follows:

1. Initializing an empty set of clusters.
2. For each news:
   2.1. Preprocess text. Split the text into words, delete punctuation (commas, dots, colons, etc.), remove stop words (prepositions, conjunctions, interjections), put the word into the normal form, for example, in the infinitive for the verb, in the masculine singular for nouns and adjectives.
   2.2. Increase the count of scanned news.
   2.3. Increase the count of number of all words included in the text of the news.
   2.4. Calculate minhash from the news and add it to the index.
   2.5. Determine the most similar news using the LSH procedure.
   2.6. For each news:
      2.6.1. Create a pair of news from step 2 and step 2.6.
      2.6.2. Define WMD between the first and second news from the pair.
      2.6.3. Define the vector TF-IDF of the first and second news.
      2.6.4. Determine the cosine of the angle between the vectors obtained in the previous step.
      2.6.5. Find the arithmetic mean between WMD and the cosine metric obtained in the previous step. Consider this number as the distance between the news.
   2.7. Find the minimum distance between processed news and all similar news.
   2.8. If this distance is less than specified threshold, add the news to the cluster that contains the news with the shortest distance, otherwise create a new cluster with the current news article.

The algorithm doesn't need any additional data for preprocessing or training to perform news clustering in the streaming mode. In this case, setting the number of clusters is not necessary, the algorithm splits the entire set of texts into clusters, depending on the threshold parameter defined in step 2.8 of the algorithm. The speed required in this task is achieved by using the LSH technique to approximate the distance between news articles. This technique is described in details in [10].

In other words, the algorithm assigns a new element of the stream to the closest cluster.

Let's review some algorithms steps in more detail. In clause 2.6.3. texts represented as vectors of TF-IDF coefficients of their words. This measure is based on the frequency of occurrence of a term (word or literal) in the text and in the entire set of texts. Use of this for the task of determining the relevance of a document to a query is described in the article [11].

Practically, any other metric can be used, but experiments show that the best option is the conversion of texts into TF-IDF vectors.

During the experiments, optimal thresholds were chosen, under which the quality of the algorithm was maximal. Of course, theese thresholds are optimal for the test set of data. In the ideal case, the threshold should vary depending on the amount and nature of the data in the system. For test data we took and manually marked clusters of news articles and posts in social networks. Quality was assessed using the F1 metric.

To calculate the F1-score, we need to determine the number of False / True Positive / Negative examples. However, in the case of clustering, it is difficult to determine the method for obtaining the values of FP, FN, TP, and TN, since the number of clusters is unknown in advance and it is impossible to determine the correspondence between the resulting cluster and the cluster from the initial set.

Therefore, it is possible to break all the initial data into a set of all possible pairs of texts belonging to the same cluster. In the same way, we will split the resulting set of clusters, as described in [12].

As a result, there are two sets of pairs, where each pair is represented by two texts referring to the same cluster. We denote the pairs obtained from the original set as G, and the pairs from the resulting set as C, so: $TP = |G \cap C|$, $FP = |C \setminus G|$, $FN = |G \setminus C|$, $F1 = \frac{2\ TP}{2\ TP+FN+FP}$.

Figure 5 shows that the F1-score was near 0.78 when using the Jaccard metric against approx. 0.82 when using TF-IDF weighting texts on the same dataset.

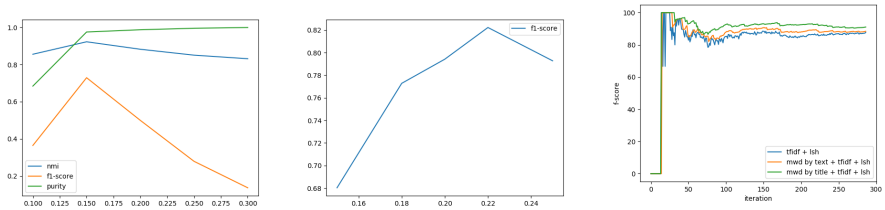
This approach works well, since it takes into account the frequency of words in the texts, but this is a purely statistical approach.

To take into account the semantic component, step 2.6.2 was added to the algorithm. At this step, the measure of the distance WMD (Word Mover's Distance) between the titles of news is estimated. WMD is an extension of the EMD task for texts and described in details in [3].

The figure 5c shows the results of using WMD.

Since counting WMD of large texts takes a long time, it was decided to evaluate WMD using the news titles. At the same time, the quality of clustering was improved, F1-score for the variant with WMD for the title and TF-IDF for the text was improved by 6%.

With the increase in the number of news, the processing time of one news is increasing, as it is required to compare a new element with an increasing number of already existing elements. To stabilize the time of one iteration, steps 2.4 and 2.5 were added to the algorithm.



(a) Jaccard distance measure

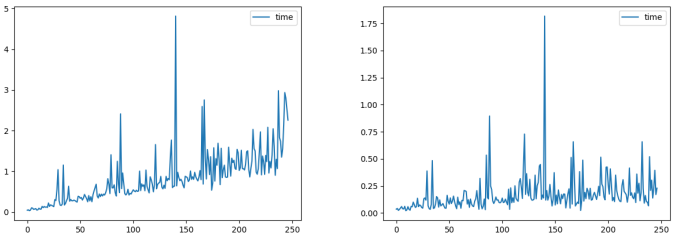
(b) TF-IDF vectorizing and cosine distance measure

(c) Comparison of clustering results when using WMD + TF-IDF and only TF-IDF

Fig. 5: Clustering quality

In these steps, the LSH method is implemented, which allows to approximate the distance measure with the MinHashing operation.

The figure 6 shows a comparison of the processing time of one news using LSH and without it depending on the iteration number.



(a) Iteration time without LSH

(b) Iteration time with LSH

Fig. 6: Comparison of the processing time of one news when using LSH and without it.

The iteration time without using LSH grows with each iteration, while on the second figure the time is much less (approx. 0.25s) and does not grow, that is, stable. This is extremely important in the system, since the iteration time directly affects the speed of news appearance in the editor's interface.

5 Selecting key phrases

Keywords combinations are extremely important for the editor, as they allow to understand quickly the essence of news articles without reading the whole text.

First, the POS-tagging of the text is performed, that is, for each word the most likely part of the speech is exposed. After this, using the regular expres-

sions by parts of the speech, the necessary word combinations are selected. For example, “adjective + preposition + noun” or “adjective + noun”. For the news, regular expressions like “preposition + noun” and “number + noun” are also important, since they allow to identify phrases like “in Moscow”, “June 21”, etc.

However, it is not enough just to select word combinations for all news, it is important to define them for the news cluster, for one information event, so aggregation is also applied for this task.

To solve this problem, the following algorithm is used in the aggregation module:

1. For each news and its phrases we create a “normalized” set of word combinations: each word in each phrase is put into its normal form.
2. For the entire news cluster, the frequencies of normalized word combinations are calculated and sorted in descending order.
3. 10 most common normalized word combinations are taken.
4. Normalized word combinations are converted to the original ones.

Therefore, for each cluster a set of the most popular word combinations is formed. Normalization in step 1 required because phrases that differ only in the case or conjugation of individual words should be considered as one word combination.

Additionally, the NER task (highlighting named entities) can be set, since the news often refers to the classical entities of this task (geographic locations, persons, corporations).

6 The implementation of sentiment analysis module

Sentiment analysis in the news analysis system is a core component that helps a reporter to better understand opinion of people about a specific news article throughout the social networks. The clustering algorithm discussed earlier is used to combine social content with the corresponding news. When the clustering finished sentiment analysis of each individual social post happens providing the reporter with a feedback about people’s opinion in the Internet to better analyze news and events. Various aggregated statistics can be built to understand areal, demographic or other distributions of users’ attitude to specific events.

6.1 The metagraph representation of sentiment analysis module

The metagraph representation of sentiment analysis module is given at Fig. 7.

The stage I is entities detection. The result of stage I is a metagraph containing entities-metavertices (the only metavertex ME is represented at Fig. 7) with included entities-vertices $E_1 \ldots E_N$.

The stage II is opinion target expression. After stage II the sentences related to entities are detected. The Fig. 7 shows metavertex SE containing sentences. Metavertices ME and SE and their inner vertices are connected with corresponding edges.

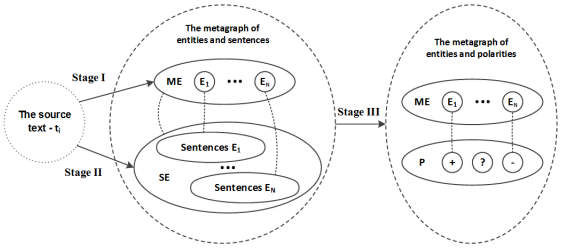


Fig. 7: The metagraph representation of sentiment analysis module.

The stage III is polarity detection. After stage III all entities are connected with corresponding polarities. Polarities are shown as metavertex P containing three vertices-polarities: positive (+), negative (-) and neutral (?). In our implementation, the neutral polarity is not used.

6.2 Sentiment analysis experiments

The task of sentiment analysis can be formulated as a problem of binary classification of text (using positive and negative polarities). As input, such classifier is supplied with a set of texts $T = \{T1, T2, \ldots, Tn\}$, at the output there is a label (1 or 0) assigned to a positive or negative text. The binary mark in practice is replaced by the probability that the text is positive.

For training and testing models we used two common datasets: IMDB Large Movie Database [6] and SAR14 [7]. Both datasets contain reviews from the IMDB site, but SAR14 has a lot more examples. Evaluating models was carried out for each dataset independently.

Two main approaches used in this system will be considered: the SVM-based approach and the convolutional neural network approach. Before model training and predicting new values, we need to preprocess the data and present it in a form that is convenient for the model. All texts passed through preprocessing — clearing of unnecessary data, bringing words to normal form, dividing into words, etc.

The next step is to words vectorization, i.e. encode text data to numeric. This operation can be performed in different ways, but the most efficient is transformation of words into a certain linear space in which the semantic proximity of words can be expressed in the distance (cosine or Cartesian) between the vectors representing the given words. At the moment there are two popular word vectorizing algorithms — Word2Vec [8] and GloVe [9].

Let's now consider each of the models for sentiment analysis separately.

6.2.1 SVM Support Vector Machine (SVM) is a linear method of classification, the main principle is to maximize the "margin" between objects of different

classes. It is a very popular model for solving natural language processing tasks [4].

Two methods of encoding text data were considered: TF-IDF for texts, and also average vectors of Word2Vec (both GloVe and Google Word2Vec). In addition, we searched for the best free hyperparameter C of the function that is optimized during SVM learning procedure:

$$\begin{cases} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \to \min_{w,b,\xi_i} \\ c_i(\mathbf{w}\cdot\mathbf{x_i} - b) \geq 1 - \xi_i, \quad 1 \leq i \leq n \\ \xi_i \geq 0, \quad 1 \leq i \leq n \end{cases}$$

C is responsible for how it's permissible to algorithm to make mistakes in learning, i.e. how wide or narrow should be the gap between classes. The larger the value of C, the smaller the gap, and vice versa. The table 1 shows the results of the experiments.

Table 1: SVM performance

	Dataset	**Word vectors**	**Test set results**	C **value**
1	imdb	TF-IDF	0.889880	0.196
2	imdb	Word2Vec	0.857280	1.347
3	imdb	GloVe:100	0.798680	0.714
4	imdb	GloVe:300	0.833040	0.100
5	sar14	TF-IDF	0.921253	0.311
6	sar14	Word2Vec	0.891255	0.709
7	sar14	GloVe:100	0.850590	0.491
8	sar14	GloVe:300	0.876588	0.945

The best quality was shown by the algorithm with $C = 0.196$ with the result of 0.89 on the test sample for the IMDB dataset and $C = 0.311$ with the result 0.92 on the test sample for the SAR14 dataset.The TF-IDF vectorization of the texts showed the best result on both datasets.

6.2.2 CNN We used model described in details in [5]. This model can be extended by applying not only one filter, but several, for example, of different sizes. This allows us to train the model to distinguish the attributes of different properties from sentences.

All the attributes are then fed to the fully connected softmax layer, where the output is the prediction of probability of tonality of text. As a regularization, Dropout layers are used, the main task of which is to leave a certain predetermined probability *keep_prob* in the neural network or not. As the vectorization of words, we compared the pretrained Wor2Vec Google, pretrained GloVe vectors, and the vectors learned in the process of network training itself. Using cross-

validation, the optimal network structure was chosen (in terms of the size of filters and the number of them).

The graph of the model is as follows is in Fig. 8. Modeling results are shown in the table 2.

Table 2: CNN results

	Dataset	**Word vectors**	**Test set results**	**Configuration**
1	imdb	ad-hoc	0.8849	[3, 5] x 128
2	imdb	Word2Vec	0.8935	[3, 5] x 128
3	imdb	GloVe:300	0.8755	[3, 5] x 128
4	sar14	ad-hoc	0.9206	[3, 5, 7] x 100
5	sar14	Word2Vec	0.9202	[3, 5, 7] x 128
6	sar14	GloVe:300	0.9109	[3, 5, 7] x 128

The best result was achieved on both datasets with pretrained vectors using the configuration [3, 5] x 128, meaning that 2 parallel feature cards with filter sizes of 3 and 5, respectively, are used. Each feature card contains 128 filters. On Fig. 9 the learning curves of the best models are shown. The loss function is not growing too fast and actually decreasing for some number of epochs, so we can be sure that the model is actually learning and loss function tends to its minimum.

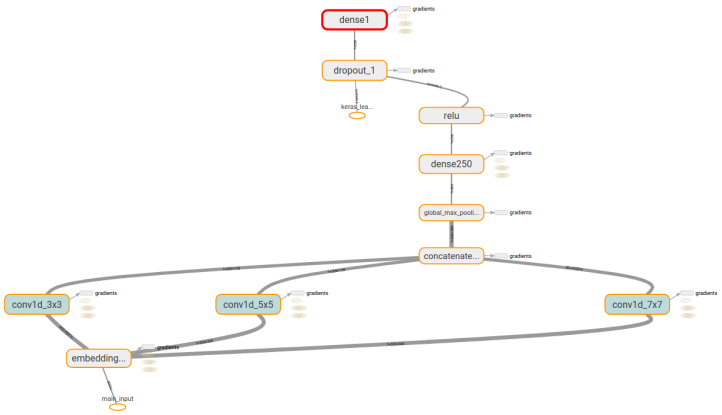
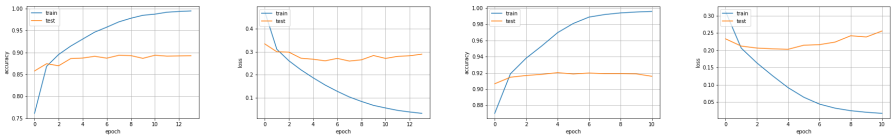


Fig. 8: CNN Graph model



(a) IMDB acc (b) IMDB loss (c) SAR14 acc (d) SAR14 loss

Fig. 9: CNN learning curves (Google Word2Vec)

So, in the developed sentiment analysis subsystem we use an ensemble of the described SVM and CNN models to achieve the best performance. The comparison of results achieved in the baseline SVM model described in [13] [14] [15] [16], SVM and CNN model discussed in this paper and also a designed ensemble model is shown in table 3.

Table 3: Results comparison

Model	**Accuracy**
SVM (Pang B.:2002)	0.829
CNN (Kim Y.:2014)	0.815
NBSVM (Wang and Manning:2012)	0.790
MV-RNN (Socher et al.:2012)	0.790
SVM	0.8899
CNN	0.8935
Ensemble	**0.9109**

7 System preview

The figure 10 shows the layout of the system interface. This page displays the most recent articles with all the clustered news from different resources. Additionally, basic information about the news and related graphics are displayed. If the news is received from social networks, sentiment analysis of its comments is conducted and displayed in the percentage of positive and negative reviews.

8 Conclusion

In this paper, we presented architecture and implementation principles of an intelligent news analysis system. The system consists of three main subsystems: a collection subsystem, a subsystem for executing text processing tasks and a subsystem for displaying information. The implementation of clustering module and sentiment analysis module are discussed in details. The operational principles of these modules are represented using metagraph approach. As the future work, we plan to improve the quality of implemented modules and add new modules such as aspect-based sentiment analysis module.

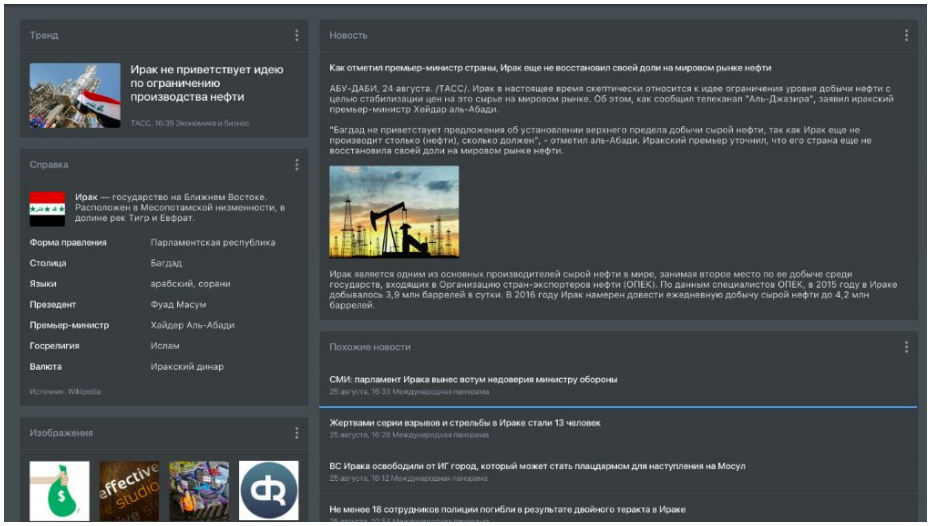


Fig. 10: Interface preview

References

1. Radev, D., Otterbacher, J., Winkel, A., Blair-Goldensohn, S.: NewsInEssence: summarizing online news topics. Communications of the ACM 48.10 (2005): 95-98.
2. Fedorenko Yu.S., Gapanyuk Yu.E.: Multilevel neural net adaptive models using the metagraph approach. Optical Memory and Neural Networks, 2016.
3. Matt J. Kusner et al.: From Word Embeddings To Document Distances, 2015.
4. Sida Wang, Christopher D. Manning: Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. Association for Computational Linguistics (ACL), 2012.
5. Kim Yoon: Convolutional Neural Networks for Sentence Classification, 2014.
6. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. Association for Computational Linguistics, 2011.
7. Nguyen, Dai Q., Nguyen, Dat Q., Vu, T., Pham, S.B.: Sentiment Classification on Polarity Reviews: An Empirical Study Using Rating-based Features. Association for Computational Linguistics, 2014.
8. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space arXiv preprint arXiv:1301.3781, 2013.
9. Pennington, J., Socher, R., Manning,. C.D.: GloVe: Global Vectors for Word Representation, 2014.
10. Jeffrey D. Ullman, Anand Rajaraman, Jure Leskovec: Mining of Massive Datasets, 2010.
11. Juan Ramos: Using TF-IDF to Determine Word Relevance in Document Queries, 2003.
12. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze: Introduction to Information Retrieval, Cambridge University Press, 2008.
13. Pang B., Lee L., Vaithyanathan S.: Thumbs Up? Sentiment Classification Using Machine Learning Techniques, Proceedings of EMNLP. 2002. P. 7986.

14. Kim Y.: Convolutional Neural Networks for Sentence Classification. 2014.
15. Sida Wang and Christopher D. Manning: Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. 2012.
16. Socher, R., Perelygin, A., Wu, J.Y., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.: Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. 2013.