# Convolutional Neural Networks for Unsupervised Anomaly Detection in Text Data

Oleg Gorokhov, Mikhail Petrovskiy(✉), and Igor Mashechkin

Computer Science Department of Lomonosov Moscow State University, MSU, Vorobjovy Gory, Moscow 119899, Russia
owlman995@gmail.com, {michael,mash}@cs.msu.su

**Abstract.** In this paper, we discuss the problem of anomaly detection in text data using convolutional neural network (CNN). Recently CNNs have become one of the most popular and powerful tools for various machine learning tasks. CNN's main advantage is an ability to extract complicated hidden features from high dimensional data with complex structure. Usually CNNs are applied in supervised learning mode. On the other hand, unsupervised anomaly detection is an important problem in many applications, including computer security, behavioral analytics, etc. Since there is no specified target in unsupervised mode, traditional CNN's objective functions cannot be used. In this paper, we develop a specific CNN architecture. It consists of one convolutional layer and one subsampling layer, we use RBF activation function and logarithmic loss function on the final layer. Minimization of the corresponding objective function helps us to calculate the location parameter of the features' weights discovered on the last network layer. We use $l_2$-regularization to avoid degenerate solution. Proposed CNN has been tested on anomalies discovering in a stream of text documents modeled with well-known Enron dataset, where proposed method demonstrates better results in comparison with the traditional outlier detection methods based on one-class SVM and NMF.

**Keywords:** Anomaly detection · Text mining · Convolutional neural network · One-class classification · Regularization · SVM · NMF

## 1 Introduction

Nowadays the interest to anomaly detection is increasing. The essence of anomaly detection problem is in searching the data whose behavior or characteristics are different from the normal ones [2]. Thus, solving the anomaly detection problem, we define criteria, describing standard behavior. If the data correspond to the criteria we call them normal. Otherwise, the data are considered as abnormal (anomalous). Anomaly detection is widely used in different areas. Recently anomaly detection in text data began to be of particular interest. This is because


H. Yin et al. (Eds.): IDEAL 2017, LNCS 10585, pp. 500–507, 2017.
https://doi.org/10.1007/978-3-319-68935-7_54

the most of information, used by humans, are texts, and finding anomalies in such information helps to solve computer security problems, to prevent information leaks, to understand clients' interests and expectations, etc.

Each word in a user's document is interpreted only in its context, therefore some traditional document-to-document similarity measures demonstrate poor quality in anomaly detection tasks [6]. Thus, there is an open problem of anomaly detection in text data the user interacts with. This paper is devoted to this topic. Classical approaches to anomaly detection in text data use the following technique. A classifier is trained on normal data only, and then it is used for separating normal and abnormal data [2]. It is worth noting, that methods based on support vector machine (SVM) and latent semantic analysis (LSA) are widely used nowadays [9,13]. The usage of convolutional neural networks (CNN) becomes popular for the text analysis. However, CNNs are mostly used for supervised classification tasks [1,7]. In this paper, we propose adaptation of the existing CNN approach for the unsupervised anomaly detection.

## 2 Anomaly Detection in Text Data

According to various researches [1,2,6,7,9,13] the anomaly detection scheme for text data consists of the stages shown in the Fig. 1.

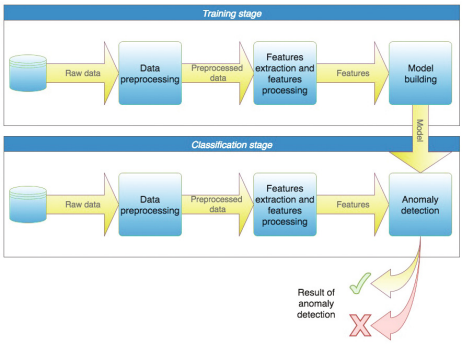


**Fig. 1.** General scheme for anomaly detection

At the first stage the data preprocessing [13] is used. After that a document is represented in the form of a sequence of words or basic word forms. Next stage is features extraction. Traditional methods represent a document as a vector of frequency characteristics of document's terms. The size of the vector is equal to the size of the dictionary [9,13]. But these methods do not take into account the initial order of words, though the order of words is important in case of using CNNs. In our approach, the alternative method based on vector representation of terms is applied [1,7,8].

At the final stage the one-class classification model is to be built. Usually it leads to the reduction of the anomaly detection problem to the optimization problem [2]. Traditional approaches to the model construction are presented by one-class SVM and LSA-based methods, discussed in [3,9,13]. Recently there is a growing interest in application of neural networks to the problem of unsupervised anomaly detection. For now, autoencoders neural networks such as replicator neural networks (RNN) presented in [4] are used for these purposes. In this research, we adapt convolution neural networks traditionally used for supervised learning to the problem of unsupervised anomaly detection [1,7].

## 3 Proposed Approach

Initially convolutional neural networks were widely used in computer vision tasks. However, it has been noticed that they also give fairly good results in the natural language processing [7]. Standard CNN is the multilayer neural network, where one or a few hidden layers are convolutional, i.e. they allow getting features from the input data by applying a filter (convolution operation). The proposed version of the architecture of a convolutional neural network is based on the research presented in [1,7].

The classical convolutional neural network has two basic operations: convolution and subsampling. The convolution operation involves the transformation of a consistent group of features according to a particular rule.

Consider a text corpus consisting of $m$ documents. As it was mentioned earlier, instead of bag-of-words features vector of the document, we use a sequence of vectors corresponding to each term in the document. In this case, each document is represented as the following vector:

$$x_{1:n} = x_1 \oplus x_1 \oplus ... \oplus x_n, \tag{1}$$

where $x_j \in \mathbb{R}^k$, $j = 0, 1, ..., n$, is the vector corresponding to the $j$th term, $k \in \mathbb{N}$ is fixed feature size, and $\oplus$ is concatenation operator.

Let $h$ be a window size, that is the length of the subsequence of features, which each filter is applied to. This length is set at the stage of the network architecture design. Now consider a filter that is a vector $w \in \mathbb{R}^{hk}$. This filter is applied to a window consisting of $h$ terms. Application of the filter to the window is described in the following way:

$$c_i = f(w \cdot x_{i:i+h-1} + b), \tag{2}$$

where $b \in \mathbb{R}$ is a bias term and $f$ is a non-linear function. Thus, by applying this filter to one window, we get an abstract feature $c_i$. The specified filter is applied to all possible windows of successive terms in the document. Thus, we obtain the following feature map:

$$c = (c_1, c_2, ..., c_n), \tag{3}$$

where $c_i \in \mathbb{R}$ is a feature extracted from $x_{i:i+h-1}$ using filer $w$.

The pooling (subsampling) operation is performed after convolution. It implements a non-linear compression of the feature map. As a rule, the maximum function is used for pooling operation. In this case, the subsampling extracts the most significant feature from the resulting map. So we get only one, the most important feature for each filter. Thus, there is the only one most significant element from the feature map corresponding to each individual filter. To determine several features from the document, we use a variety of different filters of different length. In general case, a consecutive alternation of convolution-subsampling pairs is used in the convolutional network. In our case, only one convolution and one subsampling layers are used.

The final layer follows convolutional and subsampling layers. At the last layer, it is necessary to produce a certain mapping of the feature space into a space of smaller dimension.

The considered architecture of the neural network consists of the layer that maps words into low-dimensional vectors, the convolution layer, the subsampling layer, and the output layer that uses the dropout to avoid overfitting. Convolutional, pooling, and final layers are shown in Fig. 2.

In the described approach, the index of the term in the internal vocabulary (i.e., $k = 1$) stands for the value of the corresponding term. The first layer of the neural network represents the algorithm that compresses the terms by combining

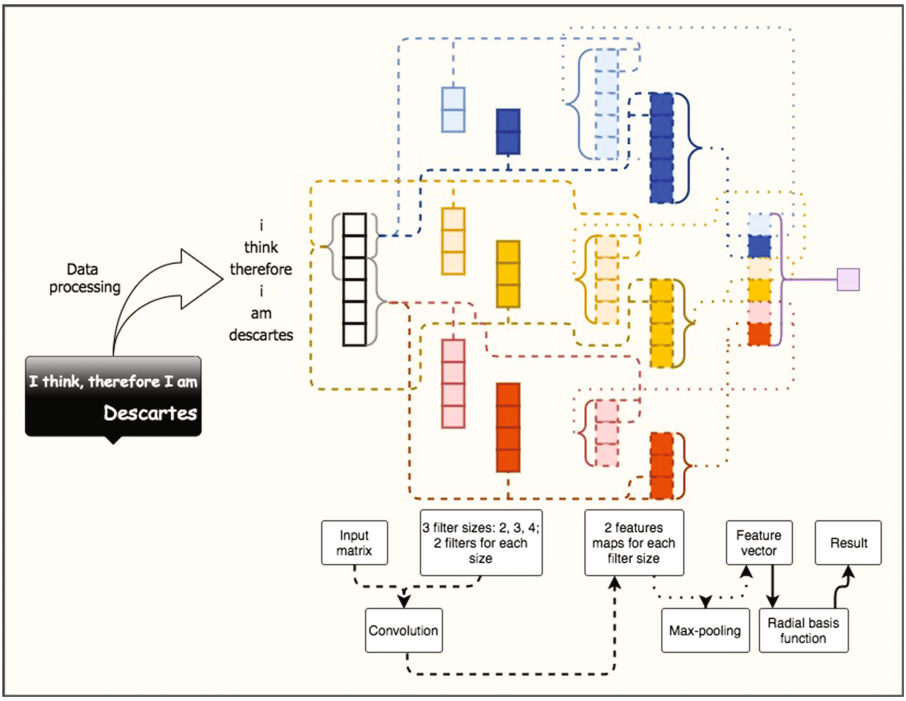


**Fig. 2.** CNN for text data analysis

them into groups according to the remainders after division of their indices by the number equal to the size of the resulting space of the processing data. For example, if the vocabulary contains 10 terms, and it is needed to be compressed to 5 elements, the indices are combined into the following five groups: [[0, 5], [1, 6], [2, 7], [3, 8], [4, 9]].

The convolutional layer is a collection of convolutional "sub-layers", each of which uses filters of a certain size. A rectifier is used as an activation function on each "sublayer". All weights on this layer are initialized using the truncated normal distribution. Biases are initialized as constants. These initial assumptions are based on the experiments carried out in [7]. Each convolutional "sublayer" is followed by a layer that compresses the feature map with subsampling operation. Features with maximum value are selected in the chosen neural network architecture. The convolution layer and the subsampling layer use several filters for windows of different sizes, which allow obtaining a vector of features. The outputs of all "sublayers" which perform subsampling are combined into one vector. Thus, at this stage we have the features vector, used as the basis of the anomaly detection in the user's work. The final layer detects abnormal data basing on the constructed features.

Dropout [1,7] is used for regularization at the last layer. Its sense is the following. At the beginning, a set of synaptic connections ($SC$) for dropout is chosen. Then the probability $p$ of $s_i$ is removed on the training stage (the synaptic weight of the corresponding connection is equal to zero) is assigned to each $s_i \in SC$. This method avoids the co-adaptation of hidden layer neurons in the learning stage, since some connections may be accidentally broken during the training process, and all the scales cannot be adjusted to each other. This regularization method is described in details in [12].

We also use $l_2$-regularization for the coordinates of the scatter of feature values center, that is weights of synaptic connections on the last layer. As it is said before, the last layer unites all the features into one vector, which is used for the anomalous data identification.

The neural network is designed to solve the problem of a one-class classification, therefore the last layer has only one output in the proposed solution. We use the Hypothesis of compactness, assuming that all feature vectors values for the normal training samples must be close to each other and form compact cloud in the discovered feature space. According to this assumption, we construct the output layer in such a way that the obtained result depends directly on the distance of the feature vector to a center of the distribution of features found in the learning process. The distance is constructed on the convolutional layer. We propose to use the radial-basis function as the activation function on the last layer and logarithmic loss function $-\ln(\cdot)$. As a result, we obtain a simple form of the quadratic objective function with $l_2$-regularization:

$$g(t) = (t - c)^2 + \alpha||t||_2, \tag{4}$$

where $c$ is the feature vector (3), $t$ are coordinates of the center of distribution of the convolutional features $c$, and $\alpha$ is a $l_2$-regularization parameter.

Thus, in the learning process, minimizing the objective function (4), we find a location parameter, that is the center of distribution of convolutional features $c$ with $l_2$-regularization $\alpha$. Regularization is applied to the coordinates of the center of distribution, i.e. to the vector $t$.

## 4 Experiments

Experimental evaluation of our method is presented in the form of comparison with the most popular classical methods, such as the one-class SVM [3,9], RNN [4], and a method based on the orthonormal non-negative matrix factorization (ONMF) [11]. ROC AUC is used for the performance evaluation.

### 4.1 Experimental Setup

Comparison of the anomaly detection algorithms is carried out on well-known ENRON dataset (see [5] for the detailed description). It includes emails obtained from 150 ENRON employees in 2000 and 2001 years. In this paper, we use the ENRON dataset with all attachments. Documents in text formats (DOC, RTF, PDF) attached to letters are considered as text data for the experiment. We select only those users, who have the cumulative size of all text files not less than 1 GB, to provide representative samples. Thus, 15 users are considered, they have 11941 text documents from 2000 to 2001. Data for each user are divided into 6-week experimental periods with 2-week step. Two weeks are used for the user model construction, two weeks for metaparameters selection and another two weeks for the testing set. We also impose an additional restriction that the number of documents in the training sample should be greater than 20. Thus, 118 experimental periods are obtained.

### 4.2 Results

To compare the performance of the proposed method with one-class SVM, we choose radial basis function as kernel function and tuned SVM meta-parameters. The best results are obtained for binary term weights (i.e., 0 if the term is not found in the document and 1 otherwise), as well as for tf-idf weights [9]. We have discovered that the most accurate results are achieved when only terms with the largest average weights in the text corpus are taken into account. We select 30–50 terms with the largest average weights.

An experimental estimation of the method based on the nonnegative orthonormal factorization is given in [10]. RNN with architecture based on [4] is also used for comparison. RNN consists of input and output layers of the same size equal to the size of the input space. Besides, there are several hidden layers organized by “hourglass” scheme. In our experiments, we choose the number of the hidden layers and their activation functions. The best results are obtained for one hidden layer with tanh activation function [4] and for three hidden layers

with linear activation functions on the outer layers and tanh activation function on the hidden (middle) layer.

In the proposed solution, it is necessary to vary the following CNN parameters: the number of filters used for feature vectors construction, number of learning epochs and the value of the $l_2$-regularization coefficient $\alpha$. In a series of experiments, it is found that optimal number of learning epochs is 400. For the chosen number of learning epochs, a series of experiments is carried out for selection of the $l_2$-regularization coefficient, as well as the number of filters of a certain size [1,7]. The results of all the experiments carried out are shown in Table 1.

Thus, it should be noted that the best result for the proposed method we have with $l_2$-regularization factor of 10,000, and with 70 filters corresponding to the window sizes of 3, 4 and 5.

**Table 1.** Results of experiments. *bin* means binary weights for terms, *tf-idf* means tf-idf weights, $l_2$ is $l_2$-regularization coefficient, *filters* stands for number of filters.

| Classification method | Median ROC AUC |
|---|---|
| SVM (bin, 30 words) | 0.8862 |
| *SVM (bin, 50 words)* | *0.8921* |
| SVM (tf-idf, 30 words) | 0.89 |
| SVM (tf-idf, 50 words) | 0.8783 |
| ONMF (Euclidean norm) | 0.8996 |
| *ONMF (max norm)* | *0.9065* |
| *RNN (one hidden layer)* | *0.7576* |
| RNN (three hidden layers) | 0.7298 |
| Proposed method ($\alpha = 0.5$, filters = 70) | 0.8088 |
| Proposed method ($\alpha = 100$, filters = 70) | 0.8801 |
| Proposed method ($\alpha = 10000$, filters = 50) | 0.9 |
| **Proposed method ($\alpha = 10000$, filters = 70)** | **0.9207** |

## 5 Conclusion

In the paper, the new algorithm for anomaly detection is developed. The algorithm solves one-class classification problem with the use of convolutional neural networks. Experimental evaluation of the proposed method shows better result in comparison with several methods traditionally used for anomaly detection problem.

**Acknowledgment.** This research is supported by the RFBR Grant No. 16-29-09555.

## References


1. Britz, D.: Implementing a CNN for text classification in tensorflow (2015). http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/
2. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 15:1–15:58 (2009)
3. Clifton, L., Clifton, D.A., Zhang, Y., Watkinson, P., Tarassenko, L., Yin, H.: Probabilistic novelty detection with support vector machines. IEEE Trans. Reliab. **63**(2), 455–467 (2014)
4. Hawkins, S., He, H., Williams, G., Baxter, R.: Outlier detection using replicator neural networks. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (eds.) DaWaK 2002. LNCS, vol. 2454, pp. 170–180. Springer, Heidelberg (2002). doi:10.1007/3-540-46145-0_17
5. Enron email dataset. www.cs.cmu.edu/./enron/
6. Kannan, R., Woo, H., Aggarwal, C.C., Park, H.: Outlier detection for text data: An extended version. CoRR abs/1701.01325 (2017)
7. Kim, Y.: Convolutional neural networks for sentence classification. CoRR abs/1408.5882 (2014)
8. Lee, J.Y., Dernoncourt, F.: Sequential short-text classification with recurrent and convolutional neural networks. CoRR abs/1603.03827 (2016). http://arxiv.org/abs/1603.03827
9. Manevitz, L.M., Yousef, M.: One-class SVMS for document classification. J. Mach. Learn. Res. **2**, 139–154 (2001)
10. Mashechkin, I.V., Petrovskii, M.I., Tsarev, D.V.: Machine learning methods for analyzing user behavior when accessing text data in information security problems. Mosc. Univ. Comput. Math. Cybern. **40**(4), 179–184 (2016)
11. Mirzal, A.: Converged algorithms for orthogonal nonnegative matrix factorizations. CoRR abs/1010.5290 (2010)
12. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
13. Tsarev, D.V., Petrovskiy, M.I., Mashechkin, I.V., Korchagin, A.Y., Korolev, V.Y.: Applying time series to the task of background user identification based on their text data analysis. Proc. Inst. Syst. Program. **27**(1), 151–172 (2015)