Записки научных
семинаров ПОМИ
Том 499, 2021 г.

V. Malykh, V. Lyalin

IMPROVING CLASSIFICATION ROBUSTNESS FOR NOISY TEXTS WITH ROBUST WORD VECTORS

Abstract. Text classification is a fundamental task in natural language processing, and a huge body of research has been devoted to it. However, there has been little work on investigating noi se robustness for the developed approaches. In this work, we are bridging this gap, introducing results on noise robustness testing of modern text classification architectures for Engl ish and Russian languages. We benchmark the CharCNN and SentenceCNN models and introduce a new model, called RoVe, that we show to be the most robust to noise.

§1. Introduction

Numerous text classification applications, including sentiment analysis and intent recognition, are related to user-generated data, where correct spelling or grammatical sentences are far from guaranteed. Real world data contains naturally arising noise, i.e., typos and mistakes in the language. According to Cucerzan and Brill [5], the probability of a typo in a user-generated input is about 10%. However, spelling errors and typos are simply ignored in most works on text classification. In this work, we study the noise robustness of existing text classification models, experimenting with different noise models, both natural and artificial. As artificial noise we consider insertion and deletion of characters in a word.

Under the vast variety of spelling errors, classical text vectorization approaches such as a bag of words with one-hot or TF-IDF encoding encounters out-of-vocabulary problems. Although there exist successful applications to low-noise tasks on common datasets [6, 7], not all models work well with real-world data such as user comments or tweets. We have conducted experiments with two architectures developed in Yoon Kim's group: a character-based CNN and a word-based combitation of CNN and

Key words and phrases: word vectors, distributed representations, d natural language processing.

This work was supported by the National Technology Initiative and PAO Sberbank project ID 0000000007417F630002.

RNN. We also consider two modifications of the latter architecture: one modification replaces word embeddings in this architecture with *FastText* word vectors, and another one does this with *RoVe* word embeddings.

This work is organized as follows. In Section 2 we describe previous works on this topic and basic units that we use in our models. In Section 3, we introduce the models in question. Sections 4 and 5 are devoted to presenting the experimental setup and obtained results with discussion and interpretation, and Section 6 concludes the work.

§2. Related Work

In recent years, a number of approaches have been proposed in order to mitigate the noisy texts issue. These approaches are mostly based on neural networks. Most research on word embeddings is devoted to fixed letter representations for a word, but there is a line of work on character-based representations that would allow a model to be robust to typos. The *FastText* model presented by Mikolov et al. [1] generates word embeddings for unknown words on the fly, based on embeddings of constituent symbol n-grams. Another approach are *Robust Vectors* presented by Malykh et al. [2], where word embeddings are generated based on the bag-of-letters representation. There also exist character-level models such as the one presented by Yoon Kim et al. [3], where the embedding of a word is based on the embeddings of its constituent characters. In the work [15], the authors investigate medical concept normalization, which is in fact a classification problem for a large number of possible classes. They use character-level attention convolutional networks, and their models (precisely four independent and related models) are closely related to the one used in this work.

In [16], a model robust to word dropping was introduced for the sentiment analysis problem. The authors used a convolutional neural network (CNN) with additional regularization in order to achieve robustness. All words unknown to the model were represented as a single token.

To broaden the perspective, we should also mention non-neural network approaches such as, e.g., [14], where the authors used topic modeling in order to improve sentiment classification. But generally speaking, so far there has been little research on noisy texts classification that would emphasize noise robustness. In this work, we are aiming to bridge this gap and motivate additional research in this area.

But there were a little research body on noisy texts classification with emphasis for the noise robustness. By this work we are aiming to bridge the gap and seed additional research in this area.

2.1. Gated Recurrent Unit. The Gated Recurrent Unit (GRU) is a common RNN architecture that preserves a hidden state between timesteps and mitigates the vanishing gradients problem by preserving the constant error carousel. It was introduced in [11].

Formally, the GRU is defined as Equations 1.

$$\begin{aligned} \boldsymbol{u}_t &= \sigma(W_{xu}\boldsymbol{x}_t + W_{hu}\boldsymbol{h}_{t-1} + \boldsymbol{b}_u), \\ \boldsymbol{r}_t &= \sigma(W_{xr}\boldsymbol{x}_t + W_{hr}\boldsymbol{h}_{t-1} + \boldsymbol{b}_r), \\ \boldsymbol{h}'_t &= \tanh(W_{xh'}\boldsymbol{x}_t + W_{hh'}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1})), \\ \boldsymbol{h}_t &= (1 - \boldsymbol{u}_t) \odot \boldsymbol{h}'_t + \boldsymbol{u}_t \odot \boldsymbol{h}_{t-1}. \end{aligned} \tag{1}$$

where $\boldsymbol{x}_t$ denotes the input vector at time t; $\boldsymbol{h}_t$, the hidden state vector at time t; $W_{x\cdot}$ (with different second subscripts), matrices of weights applied to the input; $W_{h\cdot}$, matrices of weights in recurrent connections; $\boldsymbol{b}$, the bias vectors.

2.2. Attention in recurrent neural networks. The RNNs have commonly known flaw, they rapidly forget earlier timesteps, e.g. see [13]. To mitigate this issue an attention mechanism was introduced. The already classic approach is described in paper [12].

A soft alignment model produces weights α_{ti} that control how much each input word influences the resulting output vector. The score α indicates whether the network should be focusing on this specific word right now. υ is the text vector that summarizes all the information of words. Soft attention drastically improves translation and classification for longer sentences and is now the standard approach. More formally it is described in Equations 2.

$$\begin{aligned} \boldsymbol{\upsilon}_t &= \tanh\left(W_\omega \boldsymbol{h}_t + \boldsymbol{b}_\omega\right), \\ \boldsymbol{\alpha}_t &= \frac{\exp\left(\boldsymbol{\upsilon}_t^T \boldsymbol{u}_i\right)}{\sum\limits_{j=1}^{T} \exp\left(\boldsymbol{\upsilon}_j^T \boldsymbol{u}_i\right)}, \\ \boldsymbol{\upsilon} &= \sum_{t=1}^{T} \boldsymbol{\alpha}_t \boldsymbol{h}_t, \end{aligned} \tag{2}$$

where W_ω and $\boldsymbol{b}_\omega$ are parameters of hidden states linear transformation; and $\boldsymbol{u}_i$ is some external vector, in case of GRU it could be hidden state h_i at the end of a sequence. $\boldsymbol{u}_i$ could be considered as a vector of context, meaning that vectors which are closer to context one should have more weight.

2.3. Character-level Convolutional Networks. Convolutional neural networks have proven to be a useful tool for text classification [10]. They can be constructed on the basis of individual characters and hence represent any character sequence, thus alleviating the out-of-vocabulary problem.

The *convolution* itself is usually defined as the Hadamard multiplication of some matrix k with a patch from the input A, a matrix of the corresponding size, followed by summing the resulting matrix into a single value; formally,

$$b_{ij} = \sum\sum k \odot A_{[i-s_h,i+s_h]:[j-s_v:j+s_v]},$$

where b_{ij} is the resulting value, k is the weight matrix with size $(2\cdot s_v+1)\times(2\cdot s_h+1)$, which is called the *kernel*, s_v and s_h are vertical and horizontal kernel sizes respectively; A is an original matrix which the convolutional kernel slides over.

Convolutional networks working with texts are somewhat different from working with pictures. They are often thought of as 1D convolutions, although formally every basic token (usually a word or a character) has a vector embedding, so the input is a two-dimensional matrix. But the difference is that it does not make sense to break the embedding of a word into pieces, so convolutions in natural language processing receive as input an $s_h \times d$ window where d is the dimension of the embedding, and only s_h can vary [9].

§3. Models

In our experiments, we have compared the performance of the following models.

3.1. CharCNN. In this model, the text is represented as a sequence of one-hot symbols. The model consists of a character embedding layer and a convolutional layer with 256 filters, kernel size 15, and stride 2, followed by max-pooling with kernel size 64 and stride 32. After pooling, we apply

dropout and project the 256-dimensional hidden vector to 2 dimensions by a fully-connected layer. The architecture is presented on Figure 1.

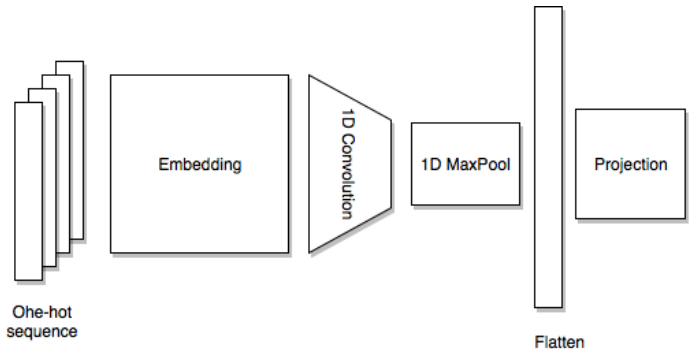

Figure 1. CharCNN.

3.2. FastText. In this model, the text is represented as a sequence if 300-dimensional vectors built using a pre-trained *FastText* model. Then this sequence is fed as input into a GRU layer with hidden state dimension 256. Next, dropout is applied to the last hidden state, and the resulting vector is projected into a 2-dimensional space.

3.3. CharCNN-WordRNN. This is a novel model architecture that we present in this work; it is similar to [3] but omits the highway layer present there. In the model, each word is represented as a sequence of one-hot symbols, and the text is represented as a sequence of word representations. Words are embedded via a convolutional layer with kernel size 5 and max-over-time pooling. The embeddings enter a GRU layer with hidden state dimension 128, and then the dropout and projection layers are applied as described in the previous section.

3.4. RoVe. Analogously to section 3.2, the text represented as a sequence 300-dimensional vectors but built using pre-trained RoVe model. It was introduced in work of Malykh [18]. We input this sequence into GRU layer with 256 hidden dim. Next, dropout is applied to the last hidden state and resulting vector is projected into 2 dimension space.

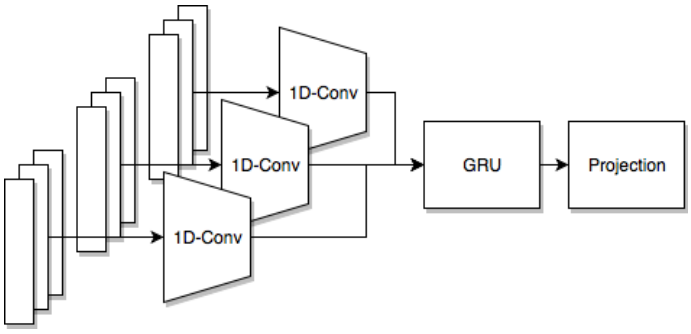

Figure 2. CharCNN-WordRNN

§4. Experiments

We have conducted three types of experiments:

- the train and test sets are spell-checked and artificial noise (section 4.2) in inserted;
- the train and test sets are not changed (with above mentioned exception for Russian corpus) and no artificial noise is added;
- the train set is spell-checked and noised, the test set is unchanged.

These experiment setups are meant to demonstrate the robustness of tested architectures to artificial and natural-born noise.

All models were trained with batch size 32, Adam optimizer and dropout before the last fully-connected layer with keep probability 0.5. A loss function is standard cross-entropy loss. CNN weights are initialized using X. Glorot initialization with normal distribution.

4.1. Datasets. We have conducted experiments on two datasets: Airline Twitter Sentiment for English language and SentiRuEval-2015 for Russian language.

Airline Twitter Sentiment consists of 14,485 tweets containing people's opinions on their experience with US airline companies. There are three classes in this dataset: positive, negative and neutral. The dataset is publicly available[1].

[1] `https://www.kaggle.com/crowdflower/twitter-airline-sentiment`

Since there is no published split for this dataset, we created one. We have shuffled the dataset and took first 75% as train set, next 15% as validation set and the last 15% are treated as test set in our experiments.[2]

SentiRuEval-2015 was introduced in work [17]. This dataset contains two sub-datasets from different domains. One domain set, so called "automotive", includes user reviews on automotive brands. It is divided to 217 in trainset and 201 in testset. Another domain set is "restaurant" one. It contains reviews of restaurants in Russia. Its trainset includes 201 reviews and its testset includes 203 reviews. All the reviews are marked up containing no sentiment, positive sentiment, negative sentiment, or both at once.

4.2. Noise Model. In order to demonstrate robustness against noise, we have performed spell-checking[3] for the above datasets and artificially introduced noise into our datasets. In the noise model, we introduce:

-
- the probability of inserting a letter after the current one, and
- the probability of a letter to be omitted

for every letter of the input alphabet and for each dataset. On every input string, we perform random letter insertions and deletions with modeled probabilities. Both types of noise are added at the same time. We test models with different levels of noise, from 0% (no noise) to 20%. According to [5], the real level of noise in user-generated texts is about 10-15%.

§5. Results

The results of training and testing models without either spell-checking or artificial noise are presented below.

The results in Table 1 are shown in order to demonstrate the tested models' behaviour in classic test environments with fixed natural noise. Unfortunately, this noise is uncontrollable, so we conducted two series of experiments with controllable noise, one where artificial noise was introduced in both training and test sets, and the other where it was only in the training set.

[2]Authors are open for sharing train/val/test split for the sake of reproducibility.

[3]We have used the publicly available enterprise-level spell-check engine *Yandex.Speller*.

Model	SentiRuEval-2015	Airline Twitter Sentiment
CharCNN	0.40	0.77
FastTextGRU	0.45	0.76
CharCNN-WordRNN	0.39	0.81
RoVe	0.38	0.78

Table 1. Results of the experiments on unchanged dataset. F_1-score on test set.

5.1. Airline Twitter Sentiment Dataset. On the Fig. 3 there are presented results in following environment: train set is cleared from natural noise and introduced the artificial one; test set is also a subject to the described transformation. We can see that FastText model is the best with no presented noise, but it is not that robust. RoVe model is overperforms FastText model from the noise level of 7.5%. From the noise level of 12.5% CharCNN and CharCNN-WordRNN models are also shoing better results by F_1-score.

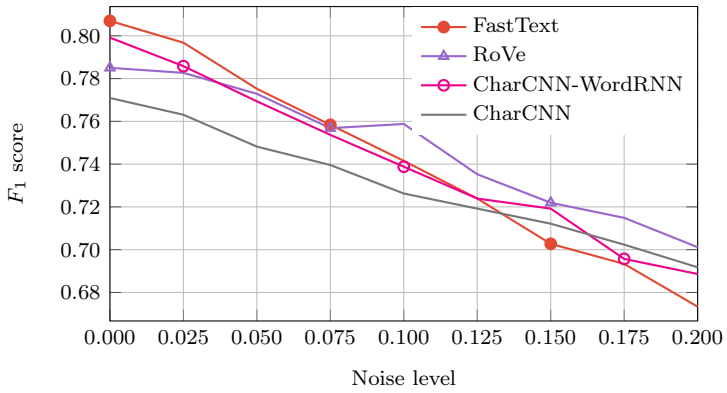

Figure 3. Airline Twitter Sentiment Dataset. Train on spell-checked and noised data, test on spell-checked and noised with the same noise level as train.

On the Fig. 4 there are presented results in following environment: train set is cleared from natural noise and introduced the artificial one; test set is remained unchanged. Here we can see a different picture: FastText model

again is the best on low levels of noise and again RoVe model outperforms FastText model, but in this experiment this is true only from 15% of noise in training data. Other models haven't shown durability in this experiment and have not exceed FastText model results.

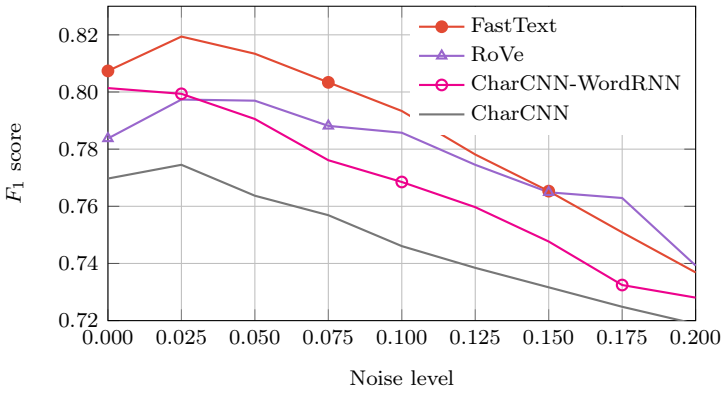

Figure 4. Airline Twitter Sentiment Dataset. Train on spell-checked and noised data, test on unchanged data.

5.2. SentiRuEval-2015 Dataset. On the Fig. 5 there are presented results in following environment: train set is cleared from natural noise and introduced the artificial one; test set is also a subject to described transformation. One can see that the behaviour of all models is remaining almost the same. FastText model shows best results on low-noised data, while RoVe model outperforms it from 15% of noise. We also could mention that CharCNN model outperforms FastText model on 17.5% of noise.

On the Fig. 6 there are presented results in following environment: train set is cleared from natural noise and introduced the artificial one; test set is remained unchanged. As we can see the demonstrated behaviour of the robustness properties is now more complex. FastText model is the best for all levels of noise, while RoVe model shows second result for almost every noise level, expect for 12.5%, where it is outperformed by CharCNN model.

Comparing two series of experiments for artificial noise induced only in the training set and in both training and test sets, one can see that the behaviour of all models is quite similar, with some differences arising

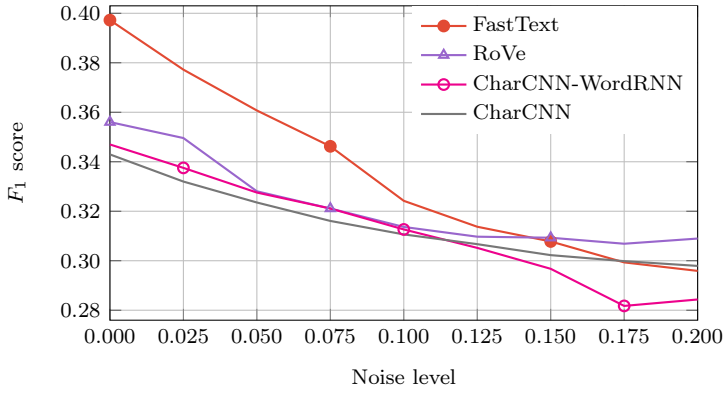

Figure 5. SentiRuEval-2015 Dataset. Train on spell-checked and noised data, test on spell-checked and noised with the same noise level as train.

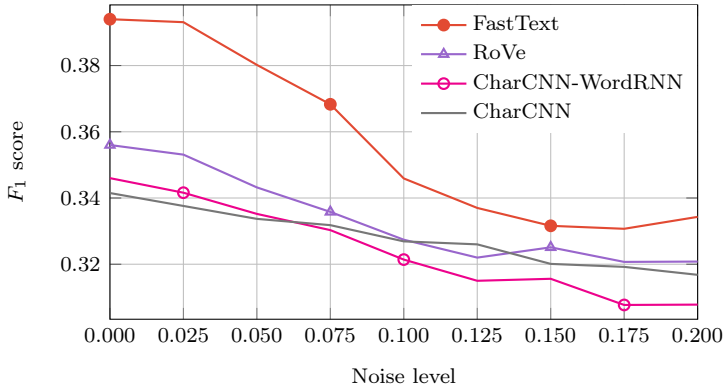

Figure 6. SentiRuEval-2015 Dataset. Train on spell-checked and noised data, test on unchanged data.

in the region of high noise probability. This fact can be interpreted as evidence that the proposed artificial noise model is a suitable substitute for natural user-generated noise: models trained with artificial noise generalize to the real world setting significantly better than trained in a noise-free

environment, and about the same as models trained on real user-generated datasets.

§6. Conclusion

We have evaluated noise robustness of common modern text classification architectures. Moreover, a proposed artificial noise is demonstrated to be adequate surrogate of natural noise in the data. Some models are performing better on highly noised training data, which could be explained by their internal adaptiveness to proposed noise, the presented RoVe model is among them. FastText model show the best performance for low noise data, while RoVe takes over from some (different from experiment to experiment) level of noise. This variable level of noise is keeping near the 10-15% of estimated natural noise, which fact allows us to state that RoVe model shows good adaptation ability to natural noise.

We see future directions of work in this field in three main domains: introducing and evaluating other noise models, testing more advanced architectures for text classification, and experimenting with more complex datasets that contain multi-class and/or multi-label classification.

The authors are grateful to Ilseyar Alimova for useful comments during the preparation of this paper.

References

1. A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, *Bag of Tricks for Efficient Text Classification*, `arXiv:1607.01759` (2016).
2. V. Malykh, *Robust Word Vectors: Embeddings for Noisy Texts*, `arXiv:1607.01759` (2018).
3. Y. Kim, Y. Jernite, D. Sontag, A. M. Rush, *Character-Aware Neural Language Models*, AAAI, 2016, pp. 2741–2749.
4. A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, Ch. Potts, *Learning Word Vectors for Sentiment Analysis*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, 2011, pp. 142–150.
5. S. Cucerzan and E. Brill, *Spelling correction as an iterative process that exploits the collective knowledge of web users*, Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 2004.
6. A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, *Bag of tricks for efficient text classification*, `arXiv:1607.01759` (2016).
7. J. Howard and S. Ruder, *Fine-tuned Language Models for Text Classification*, `arXiv:1801.06146` (2018).

8. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, E. Kaiser, I. Polosukhin, *Attention is all you need*, Advances in Neural Information Processing Systems, 2017, pp. 6000–6010.
9. X. Zhang, J. J. Zhao, Y. LeCun, *Character-level Convolutional Networks for Text Classification*, `arXiv:1509.01626` (2017).
10. Y. Kim, *Convolutional Neural Networks for Sentence Classification*, `arXiv:1408.5882` (2014).
11. K. Cho, B. van Merrienboer, D. Bahdanau, Y. Bengio, *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*, `arXiv:1409.1259` (2014).
12. D. Bahdanau, K. Cho, and Y. Bengio, *Neural Machine Translation by Jointly Learning to Align and Translate*, `arXiv:1409.0473` (2014).
13. Y. Bengio, P. Simard, P. Frasconi, *Learning long-term dependencies with gradient descent is difficult*, — IEEE transactions on neural networks, **5**, No. 2 (1994), 157–166.
14. E. Tutubalina, S. Nikolenko, *Inferring sentiment-based priors in topic models*, In Mexican International Conference on Artificial Intelligence, 2015, pp. 92–104.
15. J. Niu, Y. Yang, S. Zhang, Z. Sun, W. Zhang, *Multi-task Character-Level Attentional Networks for Medical Concept Normalization*, Neural Processing Letters, 2018, pp. 1–18.
16. Y. Li, T. Cohn, Y. Baldwin, *Learning robust representations of text*, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 2016, pp. 1979–1985.
17. N. V. Loukachevitch, et al., *SentiRuEval: Testing Object-oriented Sentiment Analysis Systems in Russian*, Proceedings of International Conference "Dialog", 2015.
18. V. Malykh, *Generalizable Architecture for Robust Word Vectors Tested by Noisy Paraphrases*, Supplementary Proceedings of the Sixth International Conference on Analysis of Images, Social Networks and Texts (AIST 2017), Moscow, Russia, 2017.

Поступило 12 января 2019 г.

St. Petersburg Department of
Steklov Institute of Mathematics,
nab. r. Fontanki, 27, 191023, St. Petersburg, Russia;
Moscow Institute of Physics and Technology,
9 Institutskiy per., 141701, Dolgoprudny, Russia;
Institute for Systems Analysis,
pr. 60-letiya Oktyabrya, 9, 117312, Moscow, Russia
E-mail: `valentin.malykh@phystech.edu`

Moscow Institute of Physics and Technology,
9 Institutskiy per., Dolgoprudny, 141701, Russia
E-mail: `lyalin@phystech.edu`