Hierarchical Attentional Hybrid Neural Networks for Document Classification

Jader Abreu[⋆], Luis Fred[⋆], David Macêdo, and Cleber Zanchettin

Centro de Informática
Universidade Federal de Pernambuco
50.740-560, Recife, PE, Brazil
`{jaoa,lfgs,dlm,cz}@cin.ufpe.br`

Abstract. Document classification is a challenging task with important applications. The deep learning approaches to the problem have gained much attention recently. Despite the progress, the proposed models do not incorporate the knowledge of the document structure in the architecture efficiently and not take into account the contexting dependent importance of words and sentences. In this paper, we propose a new approach based on a combination of convolutional neural networks, gated recurrent units and attention mechanisms for document classification tasks. The main contribution of this work is the use of convolution layers after the word embedding layer. The datasets IMDB Movie Reviews and Yelp were used in experiments and the proposed method improves the results of the current attention-based approaches.

Keywords: Text classification · Attention mechanisms · Document classification · Convolutional Neural Networks.

1 Introduction

Text classification is one of the most classical and important tasks in the machine learning field. The document classification, which is essential to organize documents for retrieval, analysis, and curation, is traditionally performed by classifiers such as Support Vector Machines or Random Forests [20] [21]. As in different areas, the deep learning methods are presenting a performance quite superior to traditional approaches in this field [5] [8]. Deep learning is also playing a central role in Natural Language Processing (NLP) through learned word vector representations [6]. The word vectors are feature extractors that are encoding semantic features of words in their dimensions generating neural language models.

In this case, the approach for text classification is similar to figure classification, and the only difference is that instead of pixel values we have a matrix of word vectors. Convolutional Neural Networks (CNN) originally popularized for text classification by employing convolutions directly to the text sentences are

[⋆] Authors contributed equally and are both first authors.

also effective for NLP [5], such as sentence modeling [8], named entity recognition [13] and sequence to sequence learning [10]. The convolutions create fixed size contexts representations, although the effective context size of the network can easily be made larger by stacking several layers.

In most NLP tasks for document classification, the proposed models do not incorporate the knowledge of the document structure in the architecture efficiently and not take into account the contexting dependent importance of words and sentences. Besides, these models are mostly based on recurrent neural networks although convolutional neural networks have strong potential for feature extraction. Much these approaches don't select qualitatively informative words and sentences since some words are more informative than others in a document [9].

BAI et al. [2] propose a convolutional architecture which simulates RNNs with a very long memory by adopting a combination of dilated and causal convolutions with residual connections. A new type of hidden unit, the GRU unit [4], inspired by the LSTM [15] [14] but simpler to compute and implement was proposed. Different from LSTM which has a memory cell and four gating units that adaptively control the information flow, the GRU has only two gating units.

In contrast, attentional mechanisms have emerged in NLP recently, allowing modeling information dependencies without regard to their distance between words in the input sequences. The approach was first introduced for machine translation where the target word generated by the decoder at each time step is aligned with all the words in the source sentence by selectively focusing on parts of input during translation [1]. With regarding with it, attention can provide insight into which words or sentences contribute to a classification decision [22]. Furthermore, attention approaches have been used in others tasks such as text summarization [11], speech recognition [19] and machine comprehension [12]. Some recent approaches also rely on RNN based bi-directional encoders to build representations of both past and future contexts in sentences, combined with attentional mechanisms for document classification [9].

In this paper, we propose a new approach to document classification based on CNN and using GRU hidden units. We also used attentional mechanisms to improve the model performance by selectively focusing the network on important parts of the text sentences during the model training. We call our model Hierarchical Attentional Hybrid Neural Networks (HAHNN). By employs a stack of deep layers to provide specialized understanding at each level of the document hierarchy HAHNN improves the document classification accuracy. We joined the Hierarchical Attentional [9], and the Temporal Convolutional Network (TCN) [2] for text classification and we get an improvement in the results. The main contribution of HAHNN is the convolution layer after the word embedding layer.

In Section II we present related works. Section III contains the structure of the model. Additionally, Section IV shows the experiments and results are described in Section V. Section VI presents some final remarks.

2 Related Works

In the literature exists a variety of methods for document and text classification. More recent works employed deep learning methods. A hierarchical neural architecture was proposed by [9], whose structure mirrors the hierarchical structure of the documents. The intuition underlying the model is that not all parts of a document are equally relevant to represent the document. Moreover, determining relevant sections involves modeling the interactions among the words and not just their presence in the text.

A problem in approaches such as word2vec [6] is that the model does not consider the morphology of the words, applying a distinct vector to each word. This is a limitation, especially for languages with large vocabularies and many rare words. These languages contain many word forms that rarely occur in the training corpus, making it difficult to learn good word representations. However, many word formations follow the same language rules, and it is possible to improve vector representations for morphologically rich languages by using the character level information. In [3] a solution was proposed by learning representations for character n-grams, and representing words as the sum of the n-gram vectors. The mentioned paper introduces an extension of the continuous skip-gram model [7] by taking into account subword information.

For the semantic modeling of sentences was introduced the Dynamic Convolutional Neural Network (DCNN) by [8] . It uses a global pooling operation over linear sequences named k-Max Pooling and applied in the network after the last convolutional layer to guarantees that the input to the fully connected layers is independent of the length of the input sentence. The k-max pooling operation makes it possible to pool the k most active features in a sequence.

To helps the RNN to remember long-term information was created Gated Recurrent Unit - GRU, a more straightforward version of computing and implementing then LSTM was proposed by [4]. It has two gates (reset and update) that effectively allows the hidden state to drop any information considered irrelevant, allowing a more compact representation. The GRU works similarly to the memory cell in the LSTM network.

A systematic evaluation of generic convolutional and recurrent architectures for sequence modeling, motivated by some recent results with convolutional neural networks is presented in [2]. These results suggest that convolutional architectures outperform the recurrent neural networks on tasks such as audio synthesis and machine translation. They also presented the Temporal Convolutional Network (TCN).

3 Hierarchical Attentional Hybrid Neural Networks

The HAHNN model combines convolutional neural networks, gated recurrent unit, and attention mechanisms to document classification. Figure 1 presents the proposed architecture. Initialy the first layer of HAHNN is a word embedding layer pre-processed (black circles in the Figure 1). The second layer (green circles) are the CNN layers, that consist of a convolutional layer with multiple filter widths and feature maps. The model of this layer uses multiple filters (with varying window sizes) to obtain multiple features by the convolutions. We also used Temporal Convolutional Network (TCN) with causal and dilated convolutions [2]. In some experiments, we have applied K-Max Pooling [8] to extract the k most active features in a sequence and apply Batch Normalization [18] for regularization too, although not has improved the model performance. We also apply a Dropout layer [17] for regularization and concatenation. In the next layers, we have a word encoder applying the word attention on word level context vector. In sequence, we have a sentence encoder applying the sentence attention on sentence-level context vector. The last use Softmax working with the document vector that predicts the probability distribution over classes. The inclusion of CNN after the word embedding layer is the most contribution of this work.

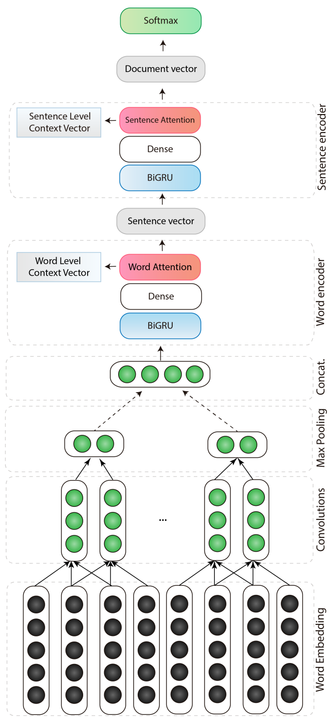


Fig. 1: Our HAHNN Architecture include an CNN layer after embedding layer. In addition, we have created an variant which includes an temporal convolutional layer [2] after embedding layer. Font: Autor.

In our model, we are using FastText approach [3] in word embedding initialization to representing words as the sum of the n-gram vectors. By taking into account subword information, it is possible to obtain representations of rare words. Therefore, taking the word *where* and $n = 3$ as an example, we are able to represent the character n-grams:

$$where =< wh, whe, her, ere, re >$$

And the special sequence:

$$< where >$$

In the n-grams above note that the sequence $< her >$, corresponding to the word *her* is different from the *tri*-gram *her* from the word *where*.

We investigate two variants of our architecture, the basic version and applying a TCN after the embedding layer. The goal is to simulate RNNs with very long memory by adopting a combination of dilated and causal convolutions with residual connections. The distinguishing characteristics of TCNs are: 1) the convolutions in the architecture are causal, meaning that there is no information leakage from future to past representation; 2) the architecture can take a sequence of any length and map it to an output sequence of the same length, just as with an RNN.

Since a simple causal convolution is only able to look back at the history of the information with a linear size in the depth of the network, it is challenging to apply causal convolutions on sequence tasks, especially those requiring longer history. The application of dilated convolutions that enable an exponentially large receptive field was an excellent solution to this problem implemented by TCN.

More formally, for a 1-D sequence input $\mathrm{x} \in \mathbb{R}^n$ and a filter $f : \{0, ..., k-1\} \rightarrow \mathbb{R}$, the dilated convolution operation F on element s of the sequence is defined as

$$F(s) = (x *_d f)(s) = \sum_{i=o}^{k-1} f(i) \cdot \mathbf{x}_{s-d \cdot i} \tag{1}$$

where d is the dilatation factor, k is the filter size, and $s - d \cdot i$ accounts for the past information direction. Dilation is thus equivalent to introducing a fixed step between every two adjacent filter maps. When $d = 1$, a dilated convolution reduces to a regular convolution. The use of larger dilation enables an output at the top level to represent a wider range of inputs, thus effectively expanding the receptive field.

The model takes into account that the different parts of a document have no similar relevant information. Moreover, determining the relevant sections involves modeling the interactions of the words, not just their isolated presence. Therefore, to include the sensitivity to this fact the model includes two levels of attention mechanisms [1]. One such structure at the word level and other at the sentence level, which let the model pay more or less attention to individual words and sentences when constructing the representation of the document.

The strategy consists of different parts: 1) A word sequence encoder and a word-level attention layer; and 2) A sentence encoder and a sentence-level attention layer. In the word encoder, the model uses bidirectional GRU [1] to produce annotations of words by summarizing information from both directions. Therefore, it incorporates the contextual information in the annotation. The attention levels let the model pay more or less attention to individual words and sentences when constructing the representation of the document [9].

Given a sentence with words $w_{it}, t \in [0, T]$ and an embedding matrix W_e, the bidirectional GRU contains the forward $GRU\overrightarrow{f}$ which reads the sentence s_i from w_{i1} to w_{iT} and a backward $GRU\overleftarrow{f}$ which reads from w_{iT} to w_{i1}:

$$x_{it} = W_e w_{it}, t \in [1, T], \tag{2}$$

$$\overrightarrow{h_{it}} = \overrightarrow{GRU}(x_{it}), t \in [1, T], \tag{3}$$

$$\overleftarrow{h_{it}} = \overleftarrow{GRU}(x_{it}), t \in [T, 1]. \tag{4}$$

An annotation for a given word w_{it} is obtained by concatenating the forward hidden state and backward hidden state, i.e., $h_{it} = [\overrightarrow{h_{it}}, \overleftarrow{h_{it}}]$, which summarizes the information of the whole sentence.

We use the attention mechanism to evaluates words that are important to the meaning of the sentence and to aggregate the representation of those informative words into a sentence vector. Specifically,

$$u_{it} = tanh(W_w h_{it} + b_w) \tag{5}$$

$$\alpha_{it} = \frac{exp(u_{it}^{\top} u_w)}{\sum_t exp(u_{it}^{\top} u_w)} \tag{6}$$

$$s_i = \sum a_{it} h_{it} \tag{7}$$

The model measures the importance of a word as the similarity of u_{it} with a word level context vector u_w and learns a normalized importance weight α_{it} through a softmax function. After that, the architecture computes the sentence vector s_i as a weighted sum of the word annotations based on the weights. The word context vector u_w is randomly initialized and jointly learned during the training process.

Given the sentence vectors s_i, and the document vector, the sentence attention is obtained:

$$\overrightarrow{h_{it}} = \overrightarrow{GRU}(s_i), i \in [1, L], \tag{8}$$

$$\overleftarrow{h_{it}} = \overleftarrow{GRU}(s_i), i \in [L, 1]. \tag{9}$$

The proposed solution concatenates $h_i = [\overrightarrow{h_i}, \overleftarrow{h_i}]$ h_i which summarizes the neighbor sentences around sentence i but still focus on sentence i. To reward sentences that are relevant to correctly classify a document, the solution again use attention mechanism and introduce a sentence level context vector u_s using it to measure the importance of the sentences:

$$u_{it} = tanh(W_s h_i + b_s) \tag{10}$$

$$\alpha_{it} = \frac{exp(u_i^\top u_s)}{\sum_i exp(u_i^\top u_s)} \tag{11}$$

$$v = \sum a_i h_i \tag{12}$$

In the above equation, v is the document vector which summarizes all the information of sentences in a document. Similarly, the sentence level context vector u_s can be randomly initialized and jointly learned during the training process.

The last layer of the model is a Softmax layer that predicts the probability distribution over classes. The output of the sentence attention layer is passed to a fully connected softmax layer. Assuming q is a vector of the inputs to the output layer and there are k output labels. The softmax function is defined as follows:

$$softmax(q) = \frac{exp(q)}{\sum_{j=1}^{k} exp(q_j)} \tag{13}$$

The source code of the method and the performed experiments is available at GitHub [1].

4 Experiments

We evaluate the proposed model on two document classification datasets. We use 90% of the data for training and the remaining 10% for the validation. We split documents into sentences and tokenize each sentence. We trained the FastText [3] in an unsupervised way to obtain the word embedding on the training and validation splits and used the word embedding to initialize the embedding layer. The word embedding has dimension 200, and the GRU dimension is 50. We use Adam optimizer to train all models with a learning rate of 0.001.

The datasets used are the IMDb Movie Reviews [2] which containing a set of 25,000 highly polar movie reviews for training, and 25,000 for testing. Classification involves detecting positive/negative reviews. The second dataset is the Yelp 2018 [3] which users give ratings and write reviews about businesses and services on Yelp. These reviews and ratings help other Yelp users to evaluate a business or a service. The dataset is for multi-class classification (ratings from 0-5 stars) and contains around 5M full review text data. We fix in 500k the evaluated size for the computational purpose.

The IMDb Movie reviews dataset have classes well balanced. However, Yelp reviews have some imbalanced classes where the majority class is *1star*. This lack of balanced classes introduces a bias and turn more challenging to train the models. In the next section we show and analyze the results.

[1] https://github.com/luisfredgs/cnn-hierarchical-network-for-document-classification
[2] http://ai.stanford.edu/ amaas/data/sentiment/
[3] https://www.yelp.com/dataset/challenge

5 Results

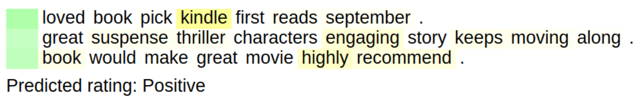
Our initial objective was to improve the accuracy obtained by [9] using an initialization of the embedding layer with a word vector trained by the semantic model of [3]. Other objectives were to analyze the impact of the use of pre-trained word vectors using the same approach of [3]. However, none of these changes brought significant results.

We have tried some different settings, but our best result was based on convolutional layers with multiple filter widths and feature maps introduced after the embedding layer.

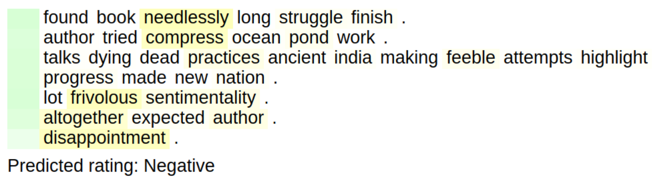
Table 1: Results in classification accuracies compared with [9].

Method	Accuracy on validation set	
	Yelp five classes, 500 k reviews	IMDb two classes, 50 k reviews
HN-ATT [9]	72.73	89.02
Our model (CNN)	**73.28**	92.26
Our model with TCN	72.63	**95.17**

To available the model we compare our results with [9], fixing Yelp (2018) dataset in 500k reviews divided into five classes. We use IMDb dataset with 50k reviews divide into two classes. Table 1 presents the results. The approach [9] obtained accuracy in the validation set of 72,73%, and the proposed model obtained 73,28%. We have evaluated in IMDb too, with 50k reviews and two classes. The approach [9] have accuracy in the evaluation group of 89,02%, and us best accuracy have 95,17% using the TCN layer.


loved book pick kindle first reads september .
great suspense thriller characters engaging story keeps moving along .
book would make great movie highly recommend .
Predicted rating: Positive

(a) A positive example of visualization of a strong word in the sentence.


found book needlessly long struggle finish .
author tried compress ocean pond work .
talks dying dead practices ancient india making feeble attempts highlight
progress made new nation .
lot frivolous sentimentality .
altogether expected author .
disappointment .
Predicted rating: Negative

(b) A negative example of visualization of a strong word in the sentence.

Fig. 2: Visualization of attention weights computed by the proposed model

We can see an improvement of the results in Yelp with our approach using CNN, an improvement in the results of IMDb with our both approaches and a relevant result in IMDb using our approach using with TCN. In the next subsections we can see more results visualizations.

5.1 Attention Weights Visualizations

An advantage provided by the attention mechanism is the visualizations of the important features by allowing to extract and visualize the attention weights. In order to validate that our model is able to select informative sentences and words in a document, we visualize the attention layers in Figure 2.

There is an example of the attention visualizations for a positive and negative class in a test review. Every line is a sentence. Green denotes the sentence weight, and yellow denotes the word weight in determining the sentence meaning. There are a greater focus on more important features like "highly recommended" in Figure 2 (a) and "disappointment" in Figure 2 (b).

This is important to analyze if our approach made important the correct sentence and word in document classification. In the next section we can analyze the visualization of word embedding.

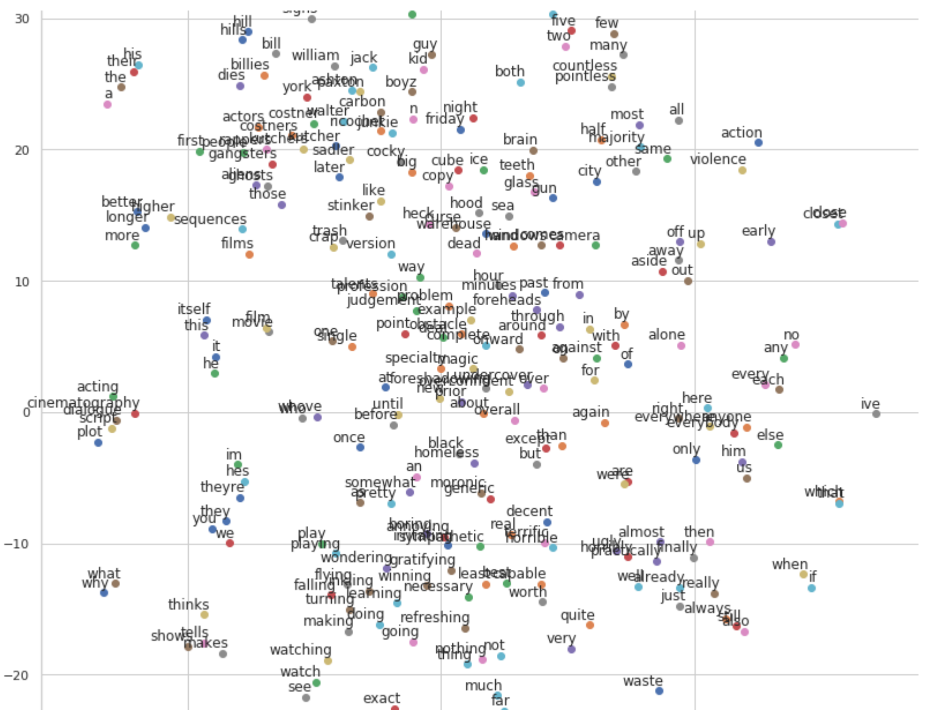


Fig. 3: Words grouped by similarity using [3] approach. Note that words such as *half* and *majority* was mapped to proximate points in geometric space because of the similarity of their meanings.

5.2 Word Embedding visualization

In experiments, we did not verify a significant improvement by use of FastText [3] against another approach as word2vec. Nevertheless, we decided to maintain fastText over other methods. In figure 3 we show similar words grouped into regions relatively close on IMDb dataset using FastText.

Note that similar words are mapped to closest points in the geometric space, where semantically similar words have similar vectors. The similarity is determined by comparing the word vectors or the "word embeddings". This means that words such as *half* and *majority* should have similar word vectors to the word *most* (because of the similarity of their meanings), whereas the word *jack* should be quite distant. For example the words "his", "then" , "the" and "a" in top-left of figure 3, and "Watching", "Watch" and "see" in the bottom area are grouped by similarity use, not only by the letters.

6 Final Remarks

In this work, we presented HAHNN architecture for document classification. The method combines attention mechanisms in both word and sentence level for better incorporate the document structure in the model and improves accuracy. Besides, we combine CNN and GRU layers for better performance. In HAHNN, the CNN play an important role in the feature extraction, whereas the bidirectional GRU get annotations of words by summarizing information from both directions, and therefore incorporate more contextual information. The inclusion of CNN after the word embedding layer is the most contribution of this work.

We evaluated the proposed model with 500k reviews of Yelp 2018 divided into five different classes and evaluated too with two classes of 50k reviews of IMDb. One can be seen in Table 1 we obtained an improvement in all the results compared with the model basis [9]. With these results, we can conclude that our architecture produces a consistent improvement for document classification. As aforementioned, in our experiments, we have used 500k reviews of Yelp (2018) dataset divided into five classes, and IMDb dataset with 50k reviews divide into two classes.

In future works, we want to apply CNN's with self-attention [16] for learning long-distance dependencies between the words in the sentence and want to use other strategies and scores to analyze the results.

Acknowledgements

This work was supported in part by FACEPE, CNPq, and CETENE (Brazilian research agencies). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan XP GPU used for this research.

References

1. BAHDANAU, Dzmitry; CHO, Kyunghyun; BENGIO, Yoshua. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.
2. BAI, Shaojie; KOLTER, J. Zico; KOLTUN, Vladlen. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271, 2018.
3. BOJANOWSKI, Piotr et al. Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606, 2016.
4. CHO, Kyunghyun et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.
5. KIM, Yoon. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882, 2014.
6. MIKOLOV, Tomas et al. Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. 2013. p. 3111-3119.
7. Tom Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In Adv. NIPS
8. KALCHBRENNER, Nal; GREFENSTETTE, Edward; BLUNSOM, Phil. A convolutional neural network for modeling sentences. arXiv preprint arXiv:1404.2188, 2014.
9. YANG, Zichao et al. Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016. p. 1480-1489., San Diego, CA, USA.
10. Gehring., et al. Convolutional Sequence to Sequence Learning. arXiv:1705.03122v3 2017
11. M. Rush., et al. A Neural Attention Model for Abstractive Sentence Summarization. arXiv:1509.00685 (2015)
12. Seo., et al. Bidirectional Attention Flow for Machine Comprehension arXiv:1611.01603 (2016)
13. Jason P.C. and Chiu, Eric Nichols. Named Entity Recognition with Bidirectional LSTM-CNNs arXiv:1511.08308 (2015)
14. Alex Graves. 2012. Supervised Sequence Labelling with Recurrent Neural Networks. Studies in Computational Intelligence. Springer
15. S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. Neural Computation, 9(8):17351780
16. Vaswani, A., Shazeer, N., Parmar, N., et al. 2017. Attention Is All You Need arXiv e-prints arXiv:1706.03762.
17. Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1):19291958, 2014.
18. Ioffe, S., & Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv e-prints , arXiv:1502.03167.
19. Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. End-to-end attention-based large vocabulary speech recognition. arXiv:1508.04395, 2015.

20. Schlag, S., Schmitt, M., Schulz, C. 2018. Faster Support Vector Machines. arXiv e-prints arXiv:1808.06394.
21. Tin Kam Ho. 1998. The Random Subspace Method for Constructing Decision Forests. IEEE Trans. Pattern Anal. Mach. 832844.
22. Du, Changshun, and Lei Huang. "Text Classification Research with Attention-based Recurrent Neural Networks. International Journal of Computers Communications Control 13.1 (2018): 50-61.