Document Classification by Using Hybrid Deep Learning Approach

Bui Thanh Hung(✉)

Data Analytics and Artificial Intelligence Laboratory
Engineering - Technology Faculty, Thu Dau Mot University,
6 Tran Van On Street, Phu Hoa District, Thu Dau Mot City,
Binh Duong Province, Vietnam
hungbt.cntt@tdmu.edu.vn

Abstract. Text classification is an essential component in a variety of applications of natural language processing. While the deep learning-based approach is becoming more popular, using vectors of word as an input for the models has proved to be a good way for the machine to learn the relation between words in a document. This paper proposes a solution for the text classification using hybrid deep learning approaches. Every existing deep learning approach has its own advantages and the hybrid deep learning model we are introducing is the combination of the superior features of CNN and LSTM models. The proposed models CNN-LSTM, LSTM-CNN show enhanced accuracy over another approach.

Keywords: Document classification · CNN · LSTM · Hybrid deep learning

1 Introduction

Text classification is a traditional topic for natural language processing, in which one needs to assign predefined categories to free-text documents. This task is essential in several applications of natural language processing such as web searching, information filtering, sentiment analysis, etc.… Therefore, it has attracted a remarkable attention from many researchers. Feature representation is a key problem in text classification. The common features based on the bag-of-words model, where unigrams, bigrams, n-grams or some exquisitely designed patterns are typically extracted as features. The range of text classification research goes from designing the best features to choosing the best possible machine learning classifiers. Nowadays, almost all techniques of text classification are based on words, in which simple statistics of some ordered word combinations usually perform the best.

The application of deep learning approaches to text classification or natural language processing in large scale has been explored in literature. Without any knowledge on the syntactic or semantic structures of a language, based on discrete embedding of words, these approaches have been proven to be competitive to traditional models.

In this paper, we focus on hybrid deep learning approached based on Long Short Term Memory (LSTM) and Convolution Neural Network (CNN). We propose two hybrid deep learning models: CNN-LSTM and LSTM-CNN; both of which are

P. C. Vinh and A. Rakib (Eds.): ICCASA 2019/ICTCC 2019, LNICST 298, pp. 167–177, 2019.
https://doi.org/10.1007/978-3-030-34365-1_13

combination of the superior features of CNN and LSTM models. We use pre-trained word embeddings to extract and weight useful affective information in accordance with their contribution to the document classification. This paper will proceed to compare both of those approaches, as well as using either CNN or LSTM separately.

The rest of this paper is organized as follows: Sect. 2 introduces related work on the document classification. Section 3 describes in details, the different approaches and how we applied hybrid deep learning models to the document classification problem. Section 4 shows the experiments, as well as discussion related to the results. Finally, Sect. 5 summarizes our work and future directions.

2 Related Works

There have been a number of studies in the world that have shown positive results in text classification [5, 8, 12, 14]. Three topics: feature engineering, feature selection and using different types of machine learning algorithms are mainly focused on traditional text classification works. The most widely used feature for feature engineering topic is the bag-of-words feature. In addition, part-of-speech tags, noun phrases features have been designed as complex features for machine learning models. Deleting noisy features and improving the classification performance are the advantages of feature selection. However, traditional approaches still have its own weakness which is data scarcity; and this has big impact on the classification accuracy.

Recently, deep neural networks, representation learning and neural models for learning word representations have inspired new ideas for solving text classification problem. Kim [13] used convolution neural networks (CNN) for text classification. Zhang et al. [12] applied a character-level CNN model. Johnson et al. [8] proposed a high-dimensional one hot vector. Socher et al. [9] used recursive neural networks for text classification. Tai et al. [7] applied the structure of a sentence and used a tree structured LSTMs for classification. There are a few works that combine LSTM and CNN structure to for sentence classification. Zhou et al. [3] used a CNN to get a sentence vector and then a recurrent neural network to compose the sentence vectors to get a document vectors. Bui [2] used hybrid deep learning mode CNN-LSTM for Vietnamese keyword extraction.

Our research follows a different approach. We use combination of the superior features of CNN and LSTM in two hybrid deep learning models: CNN-LSTM and LSTM-CNN. Hybrid deep learning LSTM-CNN, CNN-LSTM models are used for text classification. We have done the experiment and also made the comparison with other models which have been published before in the same dataset.

3 The Proposed Model

The proposed model for this work is shown in Fig. 1. The model includes three main steps: Word Embedding, Hybrid Deep Learning Models and Classification. Details of the functionality and responsibility of each layer in our model are as follows:

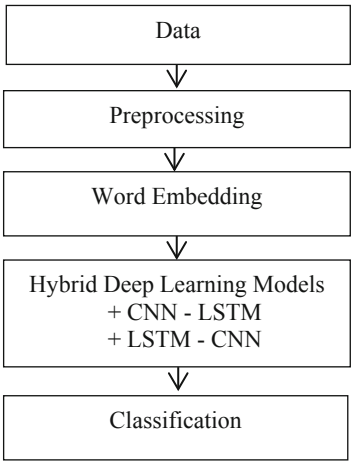

Fig. 1. The proposed model

3.1 Word Embedding

The idea of Word embedding is to capture with them as much as possible structure of the word such as the semantical/morphological/ context/hierarchical/ etc. information and convert it to vectors. Using vector representation for words has been widely known in recent days and Tomas Mikolov's Word2vec algorithm [11] is the most famous one.

We used Skip-gram [11] in this research. Objective of the Skip-gram model is to predict the contexts of a given target-word. Using a window of an arbitrary size n—by capturing n words to the left of the target and n words to its right retrieves the contexts which are immediate neighbours of the target. Skip-gram model is shown in Fig. 2.

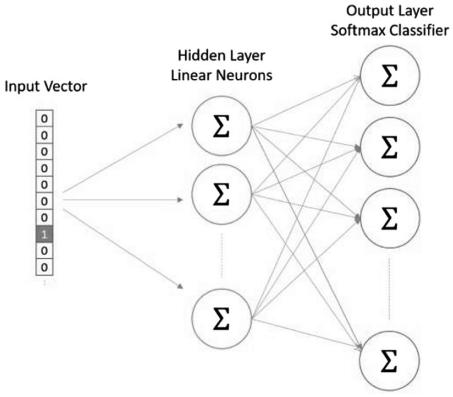

Fig. 2. Skip-gram model

3.2 Hybrid Deep Learning Models

Machine learning methods could be classified into three categories including supervised, semi-supervised and unsupervised. In the wider family of machine learning methods, Deep learning is a member. Recurrent Neural Networks (RNNs) [1, 9] and Convolutional Neural Networks (CNN) [4, 8]. [13] are two most well-known types of deep neural networks. We will describe about Long Short Term Memory (LSTM) [10], Convolutional neural networks (CNN) and Convolutional Neural Networks combined with Long Short Term Memory in hybrid deep learning models: CNN-LSTM, LSTM-CNN as follows:

LSTM

A recurrent neural network (RNN) is a type of advanced artificial neural network. RNNs can use their internal state (memory) to process sequences of inputs [1, 9]. RNNs have shown great successes in many natural language processing tasks. RNNs connect previous information to present task, since the gap between the relevant information and the place that it needs is small, RNNs can learn to use the past information. However, RNNs seem to fail to connect the information since there is still a gap between the relevant information and the words we expect it in terms of Long-Term dependencies, so RNNs has not yet proved to be workable.

A modification of the Recurrent neural networks is Long short term memory (LSTM) [10]. Comparing with the regular feed forward neural network, LSTM ables to retain the knowledge about previous outputs because of the feedback loop present in its architecture. A diagram of a simple LSTM cell is shown in Fig. 3. A large network is made by combining individual cells together. The memory is represented by the cell unit. Five main elements: an input gate i, an output gate o, a forget gate f, a recurring cell state c and hidden state output h compose in the cell. Given a sequence of vectors $(x_1, x_2, \ldots, x_n)$, σ is the logistic sigmoid function, the hidden state h_t of LSTM at time t is calculated as follows [10]:

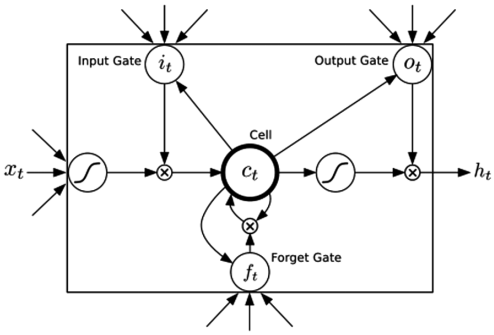

Fig. 3. The long short term memory cell

$$h_t = o_t * tanh(c_t) \tag{1}$$

$$o_t = tanh(Wx_0 x_t + Wh_0 h_{t-1} + Wc_0 c_t + b_o) \tag{2}$$

$$c_t = f_t * c_{t-1} + i_t * tanh(Wx_c x_t + Wh_c h_{t-1} + b_c) \tag{3}$$

$$f_t = \sigma\left(Wx_f x_t + Wh_f h_{t-1} + Wc_f c_{t-1} + b_f\right) \tag{4}$$

$$i_t = \sigma(Wx_i x_t + Wh_i h_{t-1} + Wc_i c_{t-1} + b_i) \tag{5}$$

CNN

CNN includes several major architectural components which are paired with each other in multi-story structure that is: Convolution, Pooling, Activation function (ReLu, Sigmoid, Tanh), and Fully connected [4, 8, 13]. This model is shown in Fig. 4.

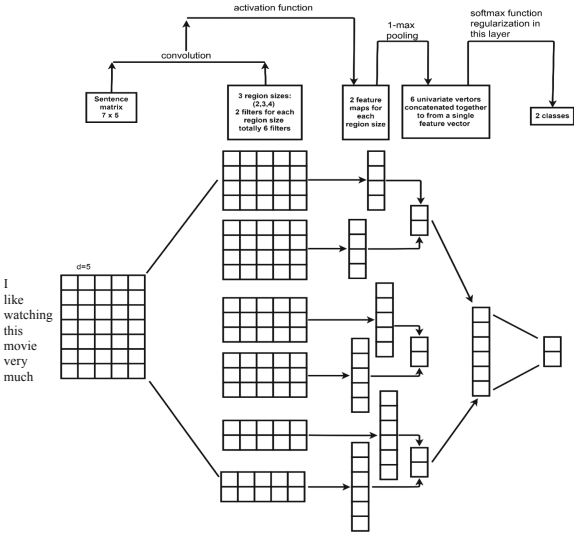

Fig. 4. Convolutional Neural Network (CNN) model

CNN-LSTM

The CNN model focused on extracting spatial information as features, whereas the LSTM model focused on extracting temporal information (from the past to the future). CNN-LSTM model [2, 3, 6], therefore, can combine and use both types of information together. The basic idea is to use the CNN model first to extract the spatial information. Then, instead of flatten them, we feed them directly to the LSTM model for classification. Figure 5 shows CNN-LSTM model.

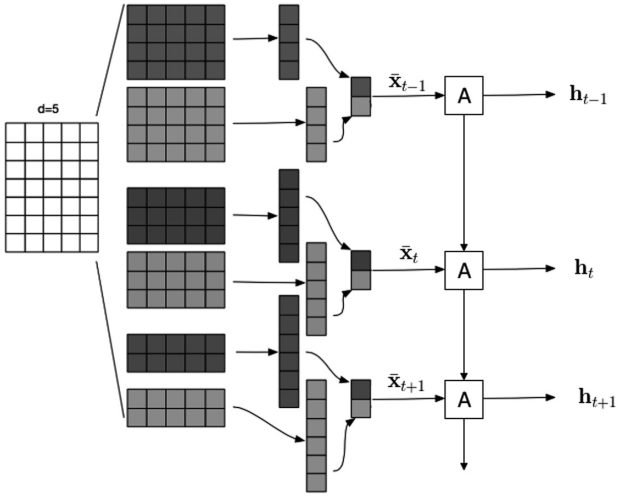

Fig. 5. CNN-LSTM model

LSTM-CNN

LSTM-CNN model combines both types of information: temporal information and spatial information together. The inputs are fed into two LSTM layers at first to capture the temporal information. Next the CNN model focused on extracting spatial information. After the max pooling layer, data is reshaped to a vector again and passed into a fully connection layer with dropout operation before feeding into the output layer. Using the dropout operation to prevent over-fitting and achieve better generalization. To calculate the probabilities of different classes, in the output layer, softmax function is chosen as the active function. The class with the highest probability will be final prediction for inputs. Figure 6 shows LSTM-CNN model.

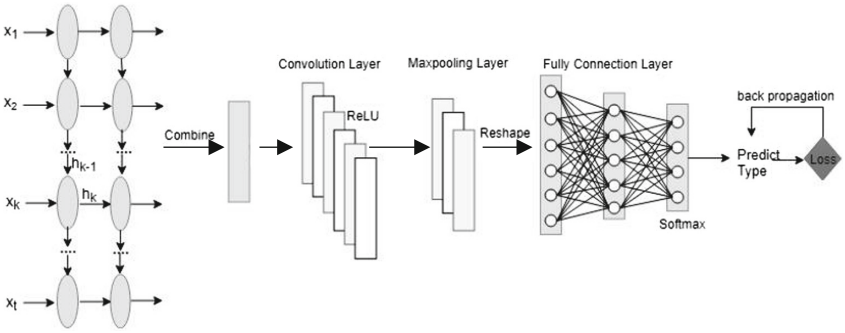

Fig. 6. LSTM-CNN model

3.3 The Classification

Following by [6], with the text vector learned from the CNN-LSTM, LSTM-CNN model, the output layer defined as:

$$y = (W_d x_t + b_d) \tag{6}$$

where

x_t: the text vector learned from the CNN-LSTM, LSTM-CNN models
y: the degree of document class of the target text
W_d, b_d: the weight and bias associated with the CNN-LSTM, LSTM-CNN models. By minimizing the mean squared error between the predicted document class and actual document class, the CNN-LSTM, LSTM- CNN model are trained. The loss function is defined as follows:

$$L(X, y) = \frac{1}{2n} \sum_{k=1}^{n} \binom{n}{k} \left\| h\left(x^i\right) - y^i \right\|^2 \tag{7}$$

$X = \left\{x^1, x^1, x^2, \ldots, x^m\right\}$: a training set of document matrix
$y = \left\{y^1, y^2, \ldots, y^m\right\}$: a document class ratings set of training sets.

4 Experiments

4.1 Dataset

We conducted a model experiment and its comparison with other models has been published before. We used 7 datasets, gathered by Zhang [12]. These data sets focus on two main aspects: emotional analysis and document classification; let us describe more detail about these datasets.

- DBPedia 2014 (DBP) includes structured information extracted from Wikipedia, this dataset has 14 labels headlines and abstracts of articles on Wikipedia.
- Yahoo! Answers (Yahoo A) includes questions and answers from Yahoo! Webscope, this dataset has 10 labels.
- AG's news (AG) includes titles and descriptions of the articles newspapers from more than 2000 different sources. This dataset has 4 labels according to the category of each article.
- Amazon reviews (Amazon) includes user product reviews on the "Amazon" website. Similar to Yelp, this dataset is divided into 2 episodes: the first episode (Amazon Polarity) consists of 2 labels "positive" and "negative" and the second episode (Amazon Full) has the number of labels corresponding to the star number of the review.

- Yelp reviews 2015 (Yelp) includes location reviews on the "Yelp" website, this data source is divided into 2 Episodes: the first volume (Yelp Polarity) consists of 2 labels, "positive" and "negative" and the second episode (Yelp Full) has the number of labels corresponding to the star number of the review.

Detailed information about the seven datasets is shown in Table 1. Datasets have data for training and testing separately.

Table 1. Overview of the dataset

Dataset	DBP	Yahoo A	AG	Amazon F	Amazon P	Yelp P	Yelp F
Train	560K	1.4M	120K	3M	3.6M	560K	650K
Test	70K	60K	7.6	560K	400K	38K	50K
Classes	14	10	4	5	2	2	5

We preprocessed and cleaned datasets by the following steps:

- Eliminate non-alphabetical characters, numeric characters except spaces
- Separating words, creating dictionaries for numbering words, dictionaries with a maximum size of 50000 words (tokenize)
- Convert words to 300 vector size (based on pre-train word2vec- FastText) [5] and set equal length sentences using padding technique. The maximum length of the sentence is 2.5 times the average length of all sentences in the filtered data set, but will be the lowest limit of 64 and the highest is 150.
- Words not included in the word vector set will be randomly generated with the mean (center) and standard deviation equal to the mean and standard deviation of the vector matrix of the dictionary (50000 words)

4.2 Experiment Setting

The models are set as follows:

- Adam Optimizer is used as the optimization algorithm, the cost function is softmax, learning rate initialized to 0.0005, size Mini-Batch 128.
- Loss calculation algorithm: Categorical cross-entropy.
- The maximum number of training sessions (epoch) is 20. During training, learning rate will be reduced 10 times if accuracy is not improved (minimum 5e-6) and stop training after 5 times with no improvement in accuracy (2 for too large dataset).
- We used word2vec- FastText published in [5] as pre-trained word embeddings.

The parameters of each model and details about how to apply each model in the document classification are described as follows:

LSTM: The input will be fed with 500 time steps. At each time step, one 300-dimensions word embedding will be fed, the result of the current time step will be affected by the result of previous time steps. This way, the temporal information will be extracted by the LSTM, and the final result will be judged based on this information as well.

CNN: to apply this model, we first need to treat the input as a 2-d image, whose each row contains a word, which is also a 300-dimension vector. The CNN model will have a window with size (*x;* 300) that moves along the rows (moves along each words). 300 is a constant, which is the same as the word-embedding size. *x* is the kernel size, which is either 2, 3 or 4 in our case. This means we are grouping together 2, 3 or 4 words to get our relevant spatial information. The result will be the spatial information that was extracted. We can then flatten them and feed them to another fully-connected layer. Finally, the output of that layer can be turned into document classification.

CNN-LSTM: For applying this model, the first part (CNN) is done similar to the CNN approach. The output of the CNN part will have the spatial information of groups of words. For a kernel size of 4, which means 4 words per group, CNN will gives an output of 125 features, from the original input of 500 words (each word is a 300-dimensions vector). We then feed these features into the LSTM, one feature per time steps. This brings the number of time steps to 125 as well. The final result should have the temporal information of the spatial information of the original input. We can then turn it into document classification.

LSTM-CNN: In this model, the first part (LSTM) is done similar to the LSTM approach. The output of the LSTM part will have the temporal information of groups of words. We use same parameters of CNN model. After the max pooling layer, data is reshaped to a vector again and passed into a fully connection layer with dropout operation before feeding into the output layer. Softmax function is chosen as the active function. We can then turn probability of temporal information of the spatial information of the original input into document classification.

4.3 Evaluation

We used accuracy score to evaluate the models. This score is calculated by Eq. in (8):

$$Accuracy = \frac{\#number\ of\ true\ predict\ labels}{\#number\ of\ predict\ lables} \tag{8}$$

We compared our models with character-level Convolutional Neural Networks (CNN) [12] (*) and LSTM-bigram (h = 10) [5] (**) since they are attractive models for text classification and did experiment in the same datasets. We used the results of these models published by the authors. In character-level Convolutional Neural Networks (CNN) we used five models named as: Lg. Full Conv. Th., Lg. w2v Conv. Th., LSTM, Sm. Lk. Conv, BoW TFIDF and ngrams TFIDF. Table 2 shows the results and performance of our Hybrid Deep Learning approach in seven datasets compared with the others. The bold number indicates which dataset approaches gives the best result for a particular model (for a column). From the results we saw that our proposed model CNN-LSTM produced the best results in four datasets: AG, Amazon F, Amazon P and Yelp F and LSTM-CNN got the best results in DBP and Yahoo A datasets. To compare the results in seven datasets between our models with the others, we calculated the sum of accuracy of each model in the seven datasets. The result is shown in Fig. 7. From the Fig. 7, we saw that CNN-LSTM model be the first rank, following is LSTM-bigram

and LSTM-CNN. To explore more about our hybrid deep learning models CNN-LSTM and LSTM-CNN, we compared in training time per epoch of each model in DBP dataset. By our experiments, CNN-LSTM is three times faster than the LSTM-CNN.

Table 2. Accuracy of our models compared with other, numbers are in percentage.

Model	DBP	Yahoo A	AG	Amazon F	Amazon P	Yelp F	Yelp P
CNN-LSTM	98.68	72.40	**92.60**	**61.32**	**95.48**	**64.34**	95.50
LSTM-CNN	**98.87**	**73.61**	92.32	59.98	95.02	62.25	95.53
LSTM, bigram (**)	98.60	72.30	92.50	60.20	94.60	63.90	**95.70**
Lg. Full Conv. Th. (*)	98.45	70.42	90.49	59.46	94.49	61.96	95.12
Lg. w2v Conv. Th. (*)	98.63	68.77	90.09	56.25	94.20	60.42	95.37
LSTM (*)	98.55	70.84	86.06	59.43	93.90	58.17	94.74
Sm. Lk. Conv (*)	98.15	69.98	89.13	56.34	94.15	58.59	94.46
BoW TFIDF (*)	97.37	71.04	89.64	55.26	91.00	59.86	93.66
ngrams TFIDF (*)	98.69	68.51	92.36	52.44	91.54	54.80	95.44

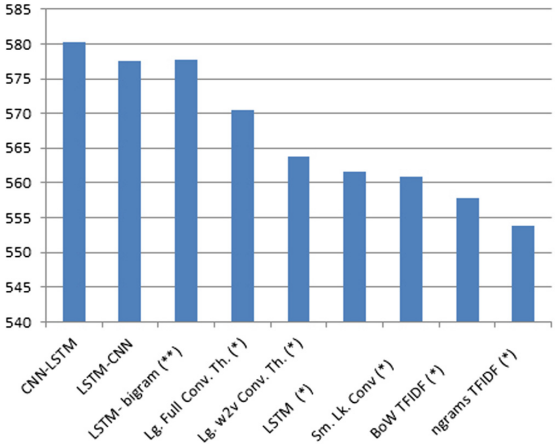

Fig. 7. Sum of accuracy of each model in the seven datasets

5 Conclusion

This paper has presented a solution for document classification using hybrid deep learning approaches. In our experiments, testing with public dataset is positive; the hybrid deep learning approach shows the better result comparing with other approaches. The result shows that CNN-LSTM model is better LSTM-CNN and also performs much faster than LSTM-CNN. In future work, we would like to apply in other deep learning models and explore more features to improve the results.

References

1. Graves, A.: Supervised Sequence Labelling with Recurrent Neural Networks, vol. 385. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-24797-2
2. Hung, B.T.: Vietnamese keyword extraction using hybrid deep learning methods. In proceedings of the 5th NAFOSTED Conference on Information and Computer Science - NICS (2018)
3. Zhou, C., Sun, C., Liu, Z., Lau, F.: A C-LSTM neural network for text classification. arXiv preprint arXiv:1511.08630 (2015)
4. Conneau, A., Schwenk, H., Barrault, L., LeCun, Y.: Very deep convolutional networks for natural language processing. CoRR, vol. abs/1606.01781 (2016)
5. Grave, E., Mikolov, T., Joulin, A., Bojanowski, P.: Bag of tricks for efficient text classification. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL, Valencia, Spain, pp. 427–431 (2017)
6. Wang, J., Yu, L.-C., Lai, K.R., Zhang, X.: Dimensional sentiment analysis using a regional CNN-LSTM model. In: The 54th Annual Meeting of the Association for Computational Linguistics, vol. 225 (2016)
7. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short term memory networks. In: Proceedings of ACL (2015)
8. Johnson, R., Zhang, T.: Convolutional neural networks for text categorization: shallow word-level vs. deep character-level. arXiv:1609.00718 (2016)
9. Socher, R., et al.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of EMNLP (2013)
10. Hochreiter, S., Schmidhuber, J.: Long short term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of International Conference on Learning Representations (ICLR 2013), Workshop Track (2013)
12. Zhang, X., Zhao, J.J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems, Montreal, Quebec, Canada, pp. 649–657 (2015)
13. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, 25–29 October Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 1746–1751 (2014)
14. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A.J., Hovy, E.H.: Hierarchical attention networks for document classification. In: NAACL HLT, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, pp. 1480–1489 (2016)