CINTI-MACRo 2019 • IEEE Joint 19th International Symposium on Computational Intelligence and Informatics and 7th IEEE International Conference on Recent Achievements in Mechatronics, Automation, Computer Sciences and Robotics • November 14-16, 2019 • Szeged, Hungary

Deep learning methods for Fake News detection

1st Viera Maslej Krešňáková
Dept. of Cybernetics and Artificial Intelligence
FEI, Technical university of Košice
Košice, Slovakia
viera.maslej.kresnakova@tuke.sk

2nd Martin Sarnovský
Dept. of Cybernetics and Artificial Intelligence
FEI, Technical university of Košice
Košice, Slovakia
martin.sarnovsky@tuke.sk

3rd Peter Butka
Dept. of Cybernetics and Artificial Intelligence
FEI, Technical university of Košice
Košice, Slovakia
peter.butka@tuke.sk

***Abstract*—Spreading of misinformation on the web nowadays represents a serious issue, as their influence on peoples opinions may be significant. Fake news represents a specific type of misinformation. While its detection was mostly being performed manually in the past, automated methods using machine learning and related fields became more critical. On the other hand, deep learning methods became very popular and frequently used methods in the field of data analysis in recent years. The study presented in this paper deals with the detection of fake news from the textual data using deep learning techniques. Our main idea was to train different types of neural network models using both entire texts from the articles and to use just the title text. The models were trained and evaluated on the Fake News dataset obtained from the Kaggle competition.**

***Index Terms*—neural networks, convolution, LSTM, text mining, fake news**

I. Introduction

Nowadays, the World Wide Web can be considered as an environment, in which people can create and share the information freely and almost without any restrictions. Most of the users act responsibly on the web and try to keep it safe and effective. But a group of web users acts in a way, that could be described as anti-social behavior. There are several definitions of the anti-social behavior available, but in general, it could be realized using two primary forms [1]. The first one is spreading of the misinformation, which could gain numerous forms hoaxes, spreading of fake news on the web. The other group is represented by the reactions of particular users, which could include discussion manipulation, cyberbullying, or similar forms. Both forms of anti-social behavior present a serious issue, as their consequences can be significant in the real world. Fake news and hoaxes can influence personal opinions of the web users and can have a substantial impact on their decisions during the elections. As a typical example, during the 2016 US presidential elections, the amount of fake news was higher than the number of real ones on Facebook [2]. This became even more significant, as the majority (61%) of targeted users considered Facebook as a primary source of the news [3].

There is a strong need to detect and eliminate the various forms of anti-social behavior effectively. Currently still frequently used are manual techniques, using human moderators, who are responsible for finding and revealing the anti-social behavior in on-line communities. However, manual detection is not only very time and resource consuming, but it also could be unreliable (e.g., because of subjective judgment, etc.) [4]. Therefore, advanced methods of data analytics and machine learning are explored to automate the process of detection of such behavior, as well as to improve the efficiency of the detection.

This paper presents the use of various deep-learning neural network models used to detect fake news from the text. The primary motivation is to use just the textual content of the document, not considering the authors characteristics, or other attributes. We also focused wanted to explore if the title of the news article could be sufficient to decide if the article present fake or real news. The paper is organized as follows: Section 2 provides state of the art in the domain of fake news detection. Theoretical overview of deep-learning models is provided in section 3. Section 4 describes the dataset used in the experiments, and its preprocessing, section 5 is dedicated to the description of the developed models, and the following section summarizes the results.

II. Fake news detection

The detection of fake news can be considered as a similar problem (from the data analysis perspective) as spam detection. In both cases, the main objective is to process the data in the form of text documents and deciding, whether they can be considered as fake news or not. Fake news detection then could be considered as a binary text classification problem, but due to nature of the data, usually techniques of natural language processing, data science and machine learning. In text-mining field, numerous machine learning approaches were already presented and studied, including scalable, distributed ones [5]–[8]. Usually, classifiers are trained using multiple features. In this case, we can consider two major types of features. Text content features represent the information extracted from the

978-1-7281-5625-5/19/$31.00 ©2019 IEEE

actual content of the article, which mostly includes the actual text of the article and its title [9]. However, there are multiple studies that also focus on covering the multimedia content included in the article, e.g., images [10], [11]. On the other hand, source and context features provide additional information about the author of the article, media which publishes the article, the source, or users who interact with it, etc. [12], [13]. Various different supervised learning models were used in the classification task in fake news detection domain [14]–[16]. Recently, also neural networks and deep-learning models were used to tackle this problem [2], [17]–[19]. Important aspect in fake news detection could be topic modelling [20], as the detection methods could perform differently in the news from different domains.

III. Deep learning

Deep forward neural networks known as *feedforward neural networks* or *multilayer perceptrons* are basic models of deep learning. The goal of forward neural networks is to approximate the function f^*. For example, $y = f^*(x)$ maps the x input to y. The forward neural network defines the mapping $y = f(x;\theta)$ and finds the value of the parameters θ, which leads to the best approximation of the function [21].

The basic model of a neuron is called perceptron. Perceptron receives input signals $\bar{x} = (x_1, x_2, \ldots, x_{n+1})$ via synaptic scales that form the vector $\bar{w} = (w_1, w_2, \ldots, w_{n+1})$. Perceptron output is given as a scalar product of the weight and vector vector, transformed by the activation function

$$output = f(\bar{w} \cdot \bar{x}) = f\left(\sum_{i=1}^{n+1} w_i x_i\right). \quad (1)$$

The basic model of perceptron is shown in the picture 1.

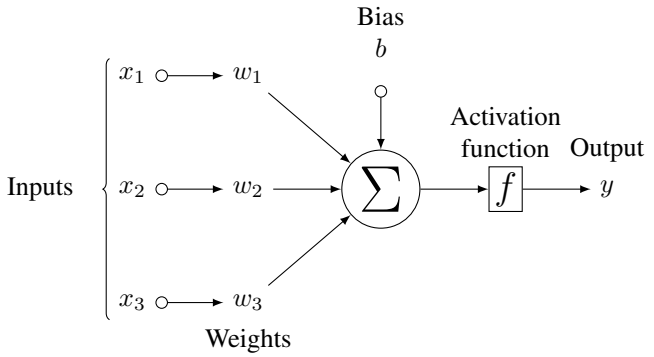

Fig. 1. Perceptron

A. Convolutional neural network

Convolutional neural networks (CNN) are special neural networks that are used for data processing. These data are interpreted as matrices. The data belonging to the time series data in the form of a 1-D matrix (values of regular time intervals) or data can be in a 2-D matrix (pixel images). CNN's are very successful in practice. The name of these networks suggests that there is a mathematical operation called convolution. The convolutional networks are in simple neural networks that use a convolution of possible multiple numerical data in one of their layers.

The notion of convolution in mathematics is defined as:

$$S(i,j) = (K*I)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \quad (2)$$

where the local operator (kernel, filter) $\mathbf{K}$ is applied, for example, to a digital image $\mathbf{I}$ [21].

B. Long-short term memory

Recurrent neural networks (RNNs) are a type of neural network for processing sequential values of $x^{(1)}, \ldots, x^{(\tau)}$. RNNs still process one element of the sequence at a time, storing a state vector in a hidden layer that contains information about the use of all elements of the sequence in the past.

Long short-term memory (LSTM) is a recurrent neural network. LSTMs use individual hidden drives whose natural behavior is to remember the inputs for a long time. A particular unit, also called a memory cell, acts as an accumulator or a gated leaky neuron that has a connection in the next steps with its weight of 1. That is, copies of its real state, adds an external signal, but this connection is multiply coded another unit that decides when to erase content from memory [22].

IV. Fake news detection

A. Dataset characteristics.

In the experiments, we used a dataset of annotated news from the Kaggle competition[1]. Overall, the dataset consisted of 20386 articles from the political news area. Each record was described using following attributes:

- `id:` unique id for a news article
- `title:` the title of a news article
- `author:` author of the news article
- `text:` the text of the article; could be incomplete
- `label:` a label that marks the article as potentially unreliable
  - `1:` unreliable
  - `0:` reliable

In the experiments, we used the attributes text, title and label, as our intention was to build the models able to decide the target attribute solely based on the textual characteristics.

B. Preprocessing

Word Embedding is a representation of text where words that have the same meaning have a similar representation. In the deep learning frameworks such as TensorFlow, Keras, this part is usually handled by an embedding layer, which stores a lookup table to map the words represented by numeric indexes to their dense vector representations.

We used the Gensim implementation of Word2Vec. The first step is to prepare the text corpus for learning the embedding by:

[1]Dataset available on https://www.kaggle.com/c/fake-news/data

0	str	1	House Dem Aide: We Didn’t Even See Comey’s Letter Until Jason Chaffetz ...
1	str	1	FLYNN: Hillary Clinton, Big Woman on Campus - Breitbart
2	str	1	Why the Truth Might Get You Fired

Fig. 2. Sample of original data

0	list	10	['house', 'dem', 'aide', 'even', 'see', 'comey', 'letter', 'jason', 'c ...
1	list	7	['flynn', 'hillary', 'clinton', 'big', 'woman', 'campus', 'breitbart']
2	list	4	['truth', 'might', 'get', 'fired']

Fig. 3. Data sample after preprocessing

0	list	10	[25, 514, 888, 218, 187, 61, 483, 1423, 2938, 8108]
1	list	7	[760, 6, 7, 71, 85, 1077, 5]
2	list	4	[287, 740, 49, 674]

Fig. 4. Text samples as vector

- creating word tokens,
- concerting to lower case,
- removing punctuation,
- removing tokens that are not alphabetic,
- removing stop words.

The Word2Vec algorithm processes documents sentence by sentence.

The next step was to convert the word embedding into a tokenized vector. Review documents are integer encoded before passing them to the Embedding layer. The integer maps to the index of a specific vector in the embedding layer. Therefore, it is essential that we lay the vectors out in the Embedding layer such that the encoded words map to the correct vector.

After we mapped embeddings from the Word2Vec model for each word to the vocabulary, we created a embeddings matrix with of word vectors. Now we are done with the data preparation. You can see a small preview of the gradual data transformation in the Fig. 3- 6.

C. Modelling

In the modeling phase, we trained various models on the training data. To train the classification models, we used 80/20 split to training and validation sets. We created four different models for both classification tasks – classification of short titles and full texts:

1) Feedforward neural network
2) CNN with one convolutional layer
3) CNN with more convolutional layer
4) LSTM

During the training phase, we evaluated various algorithm parameters to obtain the best results. Models were then validated during the training phase, and validation loss and accuracy ratios on the training and validation set were examined to avoid the models over-fitting. We have fixed the parameters and run the model multiple times to guarantee the learning variation.

The CNN sample for text classification is in Fig. 5, where the input embeddings are connected to 1D convolutional layers with multiple filter widths and feature maps are connected to max-over-time pooling layer, which is connected to fully connected layer with dropout and sigmoid function.

Summary tables for full text classification you can see in the Fig. 9- 11 in the attachment. The LSTM model includes a layer of embeddings, Dropout(0.2), LSTM(4), Dropout(0.3), fully connected layer with 100 units and activation function ReLu, Dropout(0.4) and fully connected layer with activation function sigmoid. LSTM network was trained on full-text data using the first 1000 words of the articles (because of memory limitations).

Summary tables for feedforward and convolutional neural networks for title classification you can see in Fig. 6-8 in the attachment. The LSTM model includes a layer of embeddings, Dropout(0.2), LSTM(100), Dropout(0.5), fully connected layer with 100 units and activation function ReLu, Dropout(0.2) and fully connected layer with activation function sigmoid.

The LSTM model performed very well, but only for the classification of short titles. For the full texts, we were not able to correctly perform the training due to not sufficient GPU memory in the testbed used during the experiments.

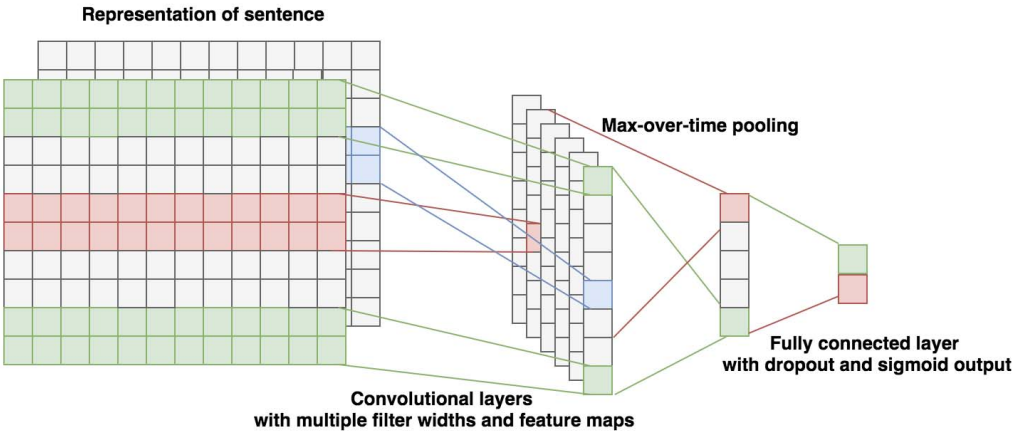

Fig. 5. Convolutional neural network for text classification

V. Experiments and evaluation

The main objective of the experiments was to evaluate the performance of described deep-learning models on the fake-news dataset. We considered two approaches, in the first one, we decided to train the models on the entire text from the article, and then we evaluated, how the networks perform when trained just on the short title text. The main reason behind this was to assess if the models could detect fake news articles only using their titles. The experiments were conducted on a PC equipped with a 4-core Intel Xeon processor clocked at 4GHz and NVIDIA Quatro P4000 GPU with 16GB memory.

To evaluate the performance of the models, we used commonly used metrics used in classification tasks:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall}, \tag{6}$$

where:

- *TP (True Positive):* when predicted fake news pieces are actually annotated as fake news,
- *TN (True Negative):* when predicted true news pieces are actually annotated as true news,
- *FN (False Negative):* when predicted true news pieces are actually annotated as fake news,
- *FP (False Positive):* when predicted fake news pieces are actually annotated as true news.

From the perspective of the fake news detection, we decided to focus on the recall of the models, as the correct detection of most of the positive examples is crucial. However, as it is also essential to minimize the FN ratio, the F1 metric could be useful, as it combines both precision and recall metrics. Besides the performance evaluation, we also measured the efficiency of model training. We measured the time required to train the model as well as the average time to prepare one epoch during the training process. Table I summarizes the results obtained on the data containing the news body. However, when comparing the model results on the title texts, we estimated, that the model would perform similarly.

TABLE I
Results on the full text data

	Accuracy	Precision	Recall	F1	Time
FF	0.898121	0.909406	0.884449	0.896754	26 s
CNN1	0.961705	0.948131	0.976890	0.962295	34 s
CNN2	0.975193	0.969480	0.981178	0.975294	82 s
LSTM	0.918593	0.908197	0.931764	0.919829	906 s

As we can see from the results, the CNN2 network achieved the best results when comparing the performance metrics, while still managed to train in a reasonable time. Table II shows the confusion matrix on the testing set.

TABLE II
Confusion matrix of the best model on the full text data

		Actual	
		Yes	No
Predicted	Yes	2016	64
	No	39	2033

A similar set of experiments was performed on the title texts. Table III summarizes the performance of the models.

TABLE III
RESULTS ON THE TITLE TEXTS

	Accuracy	Precision	Recall	F1	Time
FF	0.912796	0.854348	0.990923	0.917581	8 s
CNN1	0.911808	0.851036	0.993949	0.916957	4 s
CNN2	0.933547	0.909135	0.958716	0.933267	3 s
LSTM	0.912549	0.846620	1.000000	0.916940	25 s

As we can see from the results, all models achieved very good performance. LSTM model managed to detect all fake news articles in the testing set, while still kept the FP ratio at a reasonable level. Table IV depicts in more detail the performance of the model using a confusion matrix. It shows, that all positive examples (fake news articles) were correctly classified by the model, while 354 of true news were predicted as fake ones.

TABLE IV
CONFUSION MATRIX OF THE BEST MODEL ON TITLE TEXTS

		Actual	
		Yes	No
Predicted	Yes	1740	354
	No	0	1954

VI. CONCLUSION

The work presented in this paper is aimed to use deep learning techniques to tackle the problem of the detection of fake news from the text. We trained different neural network models (feedforward, convolutional, and LSTM) on data containing the full text of the analyzed articles as well as only title texts. The models were trained using a labeled dataset of fake and real news, and such models proved to be effective in this task. When comparing the evaluation metrics, most of the models gained consistent performance; however, convolutional and LSTM models proved to be the most effective. When comparing the evaluation metrics on the full-text data to only title texts, the models still managed to perform on a similar level. On the other hand, the effect of using just the title texts for training proved to be effective during the training phase. This could be significant in real-world tasks, when using much larger training data or when the deployed models have to be updated frequently to adjust to incoming data.

ACKNOWLEDGMENT

This work was supported by Slovak APVV research grant under the contract No. APVV-16-0213 and APVV research grant under the contract No. APVV-17-0267.

REFERENCES

[1] S. Kumar, J. Cheng, and J. Leskovec, "Antisocial Behavior on the Web: Characterization and Detection," *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 947–950, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3041021.3051106

[2] C. Budak, "What happened? The Spread of Fake News Publisher Content During the 2016 U.S. Presidential Election," 2019, pp. 139–150.

[3] A. Mitchell, J. Gottfried, and K. E. Matsa, "Facebook Top Source for Political News Among Millennials — Pew Research Center," 2015. [Online]. Available: http://www.journalism.org/2015/06/01/facebook-top-source-for-political-news-among-millennials/

[4] I. Kayes, N. Kourtellis, D. Quercia, A. Iamnitchi, and F. Bonchi, "The Social World of Content Abusers in Community Question Answering," 2016, pp. 570–580.

[5] M. Sarnovský, P. Btka, and J. Paralič, "Grid-based support for different text mining tasks," *Acta Polytechnica Hungarica*, 2009.

[6] M. Sarnovský, P. Butka, P. Bednár, F. Babič, and J. Paralič, "Analytical platform based on Jbowl library providing text-mining services in distributed environment," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015.

[7] M. Sarnovsky and N. Carnoka, "Distributed algorithm for text documents clustering based on k-Means approach," in *Advances in Intelligent Systems and Computing*, 2016.

[8] P. Butka, J. Pócs, J. Pócsová, and M. Sarnovský, "Multiple data tables processing via one-sided concept lattices," *Advances in Intelligent Systems and Computing*, 2013.

[9] H. Ahmed, I. Traore, and S. Saad, "Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10618 LNCS, 2017, pp. 127–138.

[10] Z. Jin, J. Cao, Y. Zhang, J. Zhou, and Q. Tian, "Novel Visual and Statistical Image Features for Microblogs News Verification," *IEEE Transactions on Multimedia*, vol. 19, no. 3, pp. 598–608, 2017.

[11] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, and J. Gao, "EANN," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18*, 2018, pp. 849–857. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3219819.3219903

[12] K. Wu, S. Yang, and K. Q. Zhu, "False rumors detection on Sina Weibo by propagation structures," in *Proceedings - International Conference on Data Engineering*, vol. 2015-May, 2015, pp. 651–662.

[13] L. Wu and H. Liu, "Tracing Fake-News Footprints: Characterizing Social Media Messages by How They Propagate," *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pp. 637–645, 2018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3159652.3159677

[14] X. Zhang and A. A. Ghorbani, "An overview of online fake news: Characterization, detection, and discussion," *Information Processing and Management*, 2019.

[15] T. Rasool, W. H. Butt, A. Shaukat, and M. U. Akram, "Multi-Label Fake News Detection using Multi-layered Supervised Learning," in *Proceedings of the 2019 11th International Conference on Computer and Automation Engineering - ICCAE 2019*, 2019, pp. 73–77. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3313991.3314008

[16] S. Helmstetter and H. Paulheim, "Weakly supervised learning for fake news detection on Twitter," in *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018*, 2018, pp. 274–277.

[17] S. Girgis, E. Amer, and M. Gadallah, "Deep Learning Algorithms for Detecting Fake News in Online Text," in *Proceedings - 2018 13th International Conference on Computer Engineering and Systems, ICCES 2018*, 2019, pp. 93–97.

[18] T. Saikh, A. Anand, A. Ekbal, and P. Bhattacharyya, "A Novel Approach Towards Fake News Detection: Deep Learning Augmented with Textual Entailment Features," 2019, pp. 345–358.

[19] J. C. Reis, A. Correia, F. Murai, A. Veloso, F. Benevenuto, and E. Cambria, "Supervised Learning for Fake News Detection," *IEEE Intelligent Systems*, vol. 34, no. 2, pp. 76–81, 2019.

[20] M. Smatana and P. Butka, "TopicAE: A Topic Modeling Autoencoder," *Acta Polytechnica Hungarica*, vol. 16, no. 4, jul 2019. [Online]. Available: http://uni-obuda.hu/journal/Smatana_Butka_91.pdf

[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, p. 436, may 2015. [Online]. Available: https://doi.org/10.1038/nature14539 http://10.0.4.14/nature14539

SUMMARY TABLES OF MODELS FOR TITLE CLASSIFICATION

Layer (type)	Output Shape	Param #
embedding_6 (Embedding)	(None, 72, 100)	2308900
flatten_3 (Flatten)	(None, 7200)	0
dense_20 (Dense)	(None, 256)	1843456
dense_21 (Dense)	(None, 100)	25700
dense_22 (Dense)	(None, 80)	8080
dense_23 (Dense)	(None, 20)	1620
dense_24 (Dense)	(None, 1)	21

Total params: 4,187,777
Trainable params: 1,878,877
Non-trainable params: 2,308,900

Fig. 6. Summary table of feedforward neural network for title classification

Layer (type)	Output Shape	Param #
embedding_7 (Embedding)	(None, 72, 100)	2308900
conv1d_7 (Conv1D)	(None, 72, 200)	20200
global_max_pooling1d_7 (Glob	(None, 200)	0
dense_25 (Dense)	(None, 256)	51456
dense_26 (Dense)	(None, 100)	25700
dense_27 (Dense)	(None, 80)	8080
dense_28 (Dense)	(None, 1)	81

Total params: 2,414,417
Trainable params: 105,517
Non-trainable params: 2,308,900

Fig. 7. Summary table of convolutional neural network for title classification

Layer (type)	Output Shape	Param #
input_3 (InputLayer)	(None, 72)	0
embedding_8 (Embedding)	(None, 72, 100)	2308900
conv1d_8 (Conv1D)	(None, 71, 10)	2010
conv1d_9 (Conv1D)	(None, 70, 10)	3010
global_max_pooling1d_8 (GlobalM	(None, 10)	0
global_max_pooling1d_9 (GlobalM	(None, 10)	0
concatenate_3 (Concatenate)	(None, 20)	0
dense_29 (Dense)	(None, 20)	420
dropout_3 (Dropout)	(None, 20)	0
dense_30 (Dense)	(None, 1)	21
activation_3 (Activation)	(None, 1)	0

Total params: 2,314,361
Trainable params: 2,314,361
Non-trainable params: 0

Fig. 8. Summary table of convolutional neural network with 2 convolutional layer for title classification

SUMMARY TABLES OF MODELS FOR FULL TEXT CLASSIFICATION

Layer (type)	Output Shape	Param #
embedding_2 (Embedding)	(None, 24234, 100)	18850100
flatten_2 (Flatten)	(None, 2423400)	0
dense_3 (Dense)	(None, 10)	24234010
dense_4 (Dense)	(None, 1)	11

Total params: 43,084,121
Trainable params: 24,234,021
Non-trainable params: 18,850,100

Fig. 9. Summary table of feedforward neural network for text classification

Layer (type)	Output Shape	Param #
embedding_4 (Embedding)	(None, 24234, 100)	18850100
conv1d_2 (Conv1D)	(None, 24233, 64)	12864
global_max_pooling1d_2 (Glob	(None, 64)	0
dense_7 (Dense)	(None, 100)	6500
dense_8 (Dense)	(None, 1)	101

Total params: 18,869,565
Trainable params: 19,465
Non-trainable params: 18,850,100

Fig. 10. Summary table of convolutional neural network for text classification

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 24234)	0
embedding_5 (Embedding)	(None, 24234, 100)	18850100
conv1d_3 (Conv1D)	(None, 24233, 10)	2010
conv1d_4 (Conv1D)	(None, 24232, 10)	3010
conv1d_5 (Conv1D)	(None, 24231, 10)	4010
global_max_pooling1d_3 (GlobalM	(None, 10)	0
global_max_pooling1d_4 (GlobalM	(None, 10)	0
global_max_pooling1d_5 (GlobalM	(None, 10)	0
concatenate_1 (Concatenate)	(None, 30)	0
dense_9 (Dense)	(None, 100)	3100
dropout_1 (Dropout)	(None, 100)	0
dense_10 (Dense)	(None, 1)	101
activation_1 (Activation)	(None, 1)	0

Total params: 18,862,331
Trainable params: 18,862,331
Non-trainable params: 0

Fig. 11. Summary table of convolutional neural network with 3 convolutional layer for text classification