Ingénierie des Systèmes d'Information
Vol. 24, No. 4, August, 2019, pp. 425-431
Journal homepage: http://iieta.org/journals/isi

An Evolving Hybrid Deep Learning Framework for Legal Document Classification

Neha Bansal*, Arun Sharma, R.K. Singh

Indira Gandhi Delhi Technical University for Women, Delhi, India

Corresponding Author Email: nehabansal33@gmail.com

https://doi.org/10.18280/isi.240410

Received: 10 April 2019
Accepted: 18 July 2019

Keywords:
convolution neural network (CNN), bidirectional long short-term memory (BiLSTM), neuroevolution, hyper-parameters, optimization

ABSTRACT

In the digital age, many countries have digitalized their judgement documents and uploaded them online. The automated classification of these documents could save lots of manpower and time. In this paper, an automatic classification method is developed for online judgement documents from Washington University School of Law Supreme Court Database (SCDB). Our approach was designed under a hybrid framework of deep learning, which couples convolution neural network (CNN) and with a recurrent neural network called bidirectional long short-term memory (BiLSTM). Firstly, the CNN architecture was optimized by genetic algorithm to generate the optimal word vector. Then, the vector was used to train the bi-LSTM for Softmax classification. The experimental results show that our model far exceeded the existing models in classification accuracy, indicating the effectiveness of integrating the genetically-modified CNN with the bi-LSTM.

1. INTRODUCTION

Document classification is a supervised machine learning algorithm task used in different business problems. The goal of document classification is to categorize the given document into one or more predefined categories. When the documents to be classified are large and have many categories, the classification models could be improved by incorporating more information about the documents' overall structure and adding domain specific knowledge to the model. Lately, several deep learning algorithms have attempted to preserve even more local structure by learning high-dimensional representations using recursive neural network architectures.

A Legal judgment is a decision by a court or other tribunal that resolves a controversy and adjudicates the rights and liabilities of the parties. The employment of various automatic analysis techniques in the legal domain has been a focus of researchers for several decades now. In early works, researchers used mathematical and statistical analysis for performing the legal document analysis [1-3]. With the advent of machine learning and data mining, researchers focused on employing these algorithms to the task of legal document analysis. One of the major steps in the application of such techniques is the extraction and selection of important features from the text content [4-6] and some focused on identifying the overall structure of the document for performing the analysis [7-9]. However, these methods were able to identify only the shallow features and some manually designed factors, and involved massive human efforts. The features remain biased to the problem at hand and thus, cannot be generalized to other tasks. With the successful implementation of neural networks in Kim [10] and Baharudin et al. [11] works on various NLP tasks, researchers started exploring the same in handling the legal document analysis. Recently, as per the survey done by Bansal et al. [12], deep learning is being used in handling various complex legal tasks. The major benefit of employing deep learning models is to reduce the large human effort in pre-processing of documents which is required to extract the features from the text.

We implement a novel approach based on a hybrid model of convolutional neural networks and bi-LSTM network, a variant of recurrent neural network to automatically classify judgment documents. We employ genetic algorithm to tune the hyperparameters values for the CNN network architecture. Different from traditional machine learning methods, the work of feature detection is done by the model in case of deep learning models. We summarize the main contributions of the proposed approach as follows: i) a CNN model which is used to capture the word vectors; ii) application of genetic algorithm to obtain hyperparameters values for the CNN network; iii) Application of bi-LSTM model for classification at the final step; and iv) generic approach, applicable to different languages and domains. We evaluate the approach on the datasets of judgment documents from Washington University School of Law Supreme Court Database (SCDB) [13]. We propose that the hybrid model will achieve a high accuracy when compared to other state-of-the-art works as it provides a genetically evolving CNN which will generate an improved feature vector. We also propose that the efficiency will be high as the optimized feature set will result in dimensionality reduction for the learning model. The rest of paper is organized as: Section 2 give brief discussion on related works with respect to document classification using RNN and CNN models. Section 3 describes the proposed methodology in detail. Section 4 presents the dataset and experimental results, while section 5 gives the conclusion and future work.

2. RELATED WORK

This section discusses the application of convolution

neural network and recurrent neural network in natural language field. The motivation for using CNN and RNN (bi-LSTM) architecture is two-fold: (i) The model can learn hidden semantics by extracting n-gram features from sentence though convolutional filters, and (ii) Using bi-LSTM the context of the sentence can be preserved in both the directions as bi-LSTM has the ability to remember. We also review some most recent works that have implemented genetic algorithms to optimize parameters for deep neural networks.

2.1 Convolutional neural network in NLP

Convolutional neural networks are special neural networks in which hidden layers of neurons are replaced by one or more convolutional and polling layers, followed by fully-connected layers. Convolution neural network were initially introduced for solving the complex tasks in computer vision. However in the recent past, CNNs have been successfully applied to text classification task. Kim et al. [10] proposed a basic CNN network for sentiment analysis task using word2vec embeddings. Author concluded that a simple CNN network with just one convolutional layer and little hyperparameter tuning outperformed many state-of-the-art classifiers. Kotzias et al. [14] proposed a CNN based framework for polarity prediction of three review datasets. Authors concluded that the accuracy obtained was comparable and the approach was scalable. A very deep CNN model for text classification by having a depth up to 29 convolutional layers was given by Conneau et al. [15]. Authors conducted experiment on eight large text datasets available freely, which included sentiment classification, topic categorization, and news categorization tasks. Dahou et al. [16] used CNN with differential evolution algorithm to perform Arabic sentiment classification. Experimental results were evaluated on five different Arabic sentiment datasets and the accuracy was found to be better than other state-of-the-art algorithms. Character level convolution neural network was used by Zhang et al. [17] for text classification. Long length document classification using local convolution feature aggregation was proposed by Liu et al. [18]. Authors developed three models and conducted experiments on four-class arXiv paper dataset and concluded that good accuracy was achieved. Besides classification, CNN have been adapted for solving other text analytics tasks such as question answering [19-21] and language modelling [22, 23]. CNNs are giving state-of-the art results in solving various problems.

2.2 Recurrent neural network in NLP

The number of pages for the manuscript must be no more than ten, including all the sections. Please make sure that the whole text ends on an even page. Please do not insert page numbers. Please do not use the Headers or the Footers because they are reserved for the technical editing by editors.

2.3 Neuroevolution in improving deep learning models

Neuroevolution (NE) is the process of improving neural network parameters and architecture using evolutionary algorithms. A large number of researchers are working on the task of evolving deep neural networks using Neuroevolution. Genetic algorithms are metaheuristic and imbibe the natural selection process. The core idea of genetic algorithms is to evolve individuals by applying standard operators such as selection, mutation and crossover. A multimode evolutionary neural network was proposed by Young et al. [29] to learn optimised CNN parameters via a genetic algorithm. Suganuma et al. [30] used cartesian approach based genetic programming to build optimized CNN architecture for image classification task. Hutter and Loshchilov [31] applied Covariance Matrix Evolution Scheme for tuning parameters of convolution and fully-connected layers. Dahou et al. [32] implemented a Differential Evolution algorithm to evolve the features for sentiment classification for Arabic language using the CNN model. Xie and Yullie [33] also worked on evolving CNN architecture by representing GA individuals using a binary encoded scheme.

3. PROPOSED METHODOLOGY

The proposed framework is used to develop a classification architecture that is configured automatically using genetic algorithm. Also, instead of using LSTM, we propose to make use of a bidirectional LSTM network which helps in capturing the sequential context from both left and right. The proposed framework will consist of three stages: pre-processing using embeddings, optimizing CNN hyper-parameters using genetic algorithm, and bi-LSTM network layer at the end. The proposed classification architecture is shown in Figure 1.

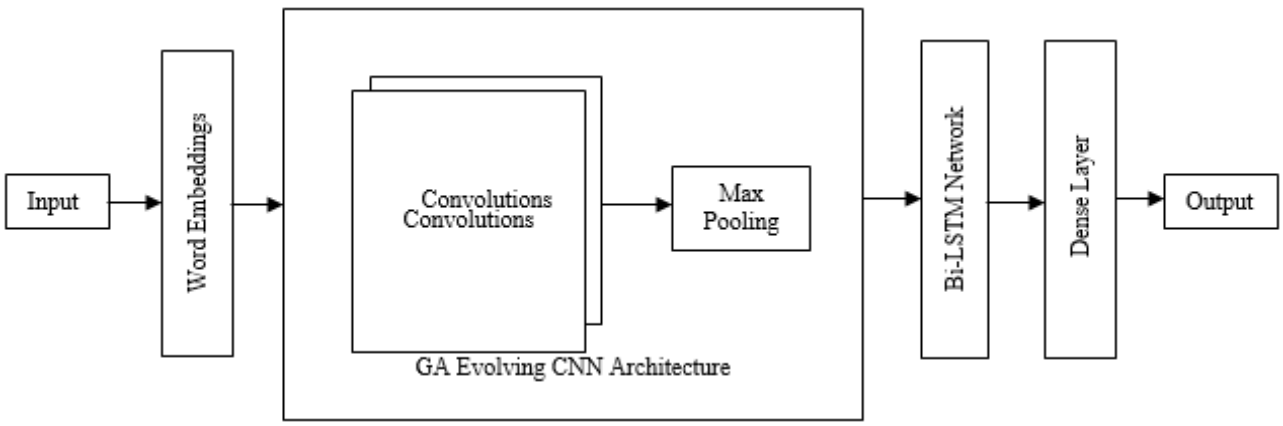


Figure 1. Proposed classification architecture

3.1 Pre-processing

The first layer performs pre-processing on the input. Input document is converted in the form of a vector representation corresponding to each word or sentence. The use of word embeddings for representing text in the form of a vector has become a standard approach. This can be done by building an embedding from the text corpus or by using a pre-trained embedding.

GloVe [34] and word2vec [35] are two successful pre-trained word embeddings that are being used extensively as they have proven to attain high accuracy results. Both the embeddings represent text as continuous word vectors in a low dimensional space and are very useful in various text processing tasks. Glove embeddings are used in this work. We also constructed one embedding from the corpus and tested results using the same. A maximum length was set for sentence to make it uniform throughout the set.

3.2 Evolving CNN network using genetic algorithm

CNNs were first introduced in the paper of Yann Lecun et al. [36]. Earlier being used for computer vision tasks, over the time they have been evolved to solve challenging text data. The text input to the CNN has two dimensions, among which one dimension denotes the d-dimensional vector of each word, while the other captures the number of words. So, if we assume a total of 30 words wherein each word is represented with a 100 dimensional vector, then the input size to CNN will be 1* 30*100. The process flow for genetic algorithm in designing CNN network is shown in Algorithm 1. The two prerequisites for genetic algorithm are, a genetic representation of the solution domain and second, a fitness function to evaluate each individual. The algorithm works by going through the standard bio-inspired operations and discovers the best hyperparameters for the CNN network. The evolving genetic framework consists of three stages: initialization, evaluation and update. Each stage of the proposed framework is discussed in detail in the following sections.

3.2.1 Initialization stage

In the initialization stage, the process starts by providing the algorithm with the input document dataset, the population size and the generation number of the genetic algorithm and specifying a predefined set of building blocks for CNN network. The population $\boldsymbol{X_{NXN_{par}}}$, consists of N solutions andN_{par}, which gives the number of parameters to be optimized.

The hyperparameters used to control the CNN network configuration in proposed model are number of filters, kernel size, Initialization mode and drop-out rate. The set of values for each parameter is generated and genetic parameters crossover and mutation are set. A random integer population of size N is generated using the following equation:

$$x_{ij} = l_j + rand * (u_j - l_j), where$$

j varies from 1 to N_{par} and i varies from 1 to N, and l_j and u_j represents the lower and upper values for the jth parameter. The sample values for one solution x_i are shown in Table 1. The values show a configuration to build a CNN model. These values are now provided to the evaluation stage.

Table 1. Sample values for solution

Parameters	Example Value
Number of filters	5
Filtersize	[3, 4]
Initialization mode	Uniform
Dropout rate	0.6

Algorithm 1: Genetic Algorithm to optimize hyperparameters for CNN Network

1: Input: The Judgment dataset *D*, Initial population *X*, Termination Criteria (max accuracy with min features), crossover and mutation probabilities.

2: Initialization: Splitting the dataset into training and testing using 80:20 for building and evaluating CNN network using initial population $X_{NXN_{par}}$, where *N* is the number of solutions and $N_{par}$is the number of parameters to be used for configuring CNN.

3: *for i = 1,2,…,I* do

Evaluation:

4: *CNN_net*← building CNN architecture using training dataset and initial population for $\forall x_i$ in X;

5: *fit_val(CNN_net)*← compute fitness of each CNN_net using the defined fitness function;

Updation:

6: *(CNN_net)*x_b← select CNN_net with best solution x_b

7: *Crossover:* for any two solutions, a random split position is chosen. Two new solutions are created by swapping the information at the split point.

8: *Mutation:* for each non-crossover solution, mutation operation is applied

9: *Selection:* producing a new population

10: *end for*

11: Output: a set of individuals in the final population with maximum accuracy and minimum number of features.

3.2.2 Evaluation stage

In this stage the CNN network is build using the parameters finalized in the initialization stage. The 80:20 split method is used to divide the dataset randomly into training and testing sets. The training dataset is used to build the network and testing set is used to evaluate the model on unseen data by using the fitness function.

3.2.3 Update stage

The best solution x_b with the highest value for the fitness function is chosen. After this, each solution in the current population is updated using the three operators of the genetic algorithms: mutation, crossover and selection. The evaluation and update stage are repeated until the termination criterion is reached.

3.3 Bi-directional LSTM layer for classification

Bidirectional LSTM is a variant of LSTM recurrent neural networks. Recurrent Neural Networks have proven to be very effective in the tasks of text processing. An RNN is capable of utilizing word order information and use this to construct a more precise semantic expression for each word. However, a regular LSTM is not able to capture the future contextual information while processing sequences, whereas a bi-LSTM is able to use both past and future contexts by processing the text from both directions. Based on the work of [13], this work further optimizes the model to change the unidirectional LSTM layer to a bidirectional LSTM layer.

A bi-LSTM network involves duplicating the first recurrent layer in the network so that there are now two

layers side-by-side, which provide the input sequence as-is as input to the first layer and provide a reversed copy of the input sequence to the second. Thus, the word vector for the input has an expression obtained by LSTM in two directions. A bidirectional LSTM network is shown in Figure 2. At the end, a dense layer is used is used with size equal to number of classes and SoftMax function to convert number into probability.

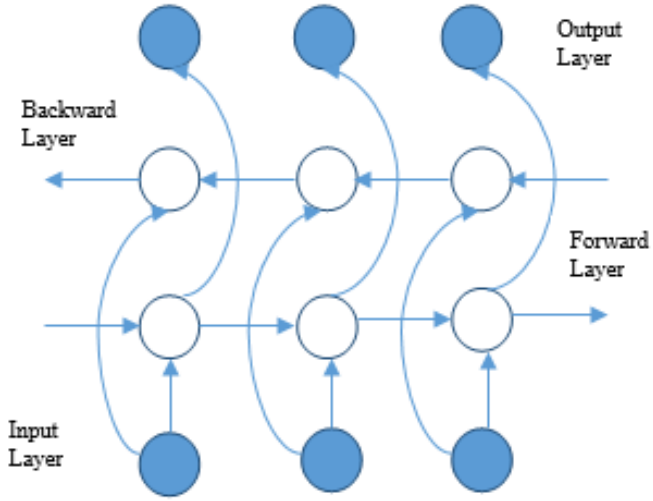
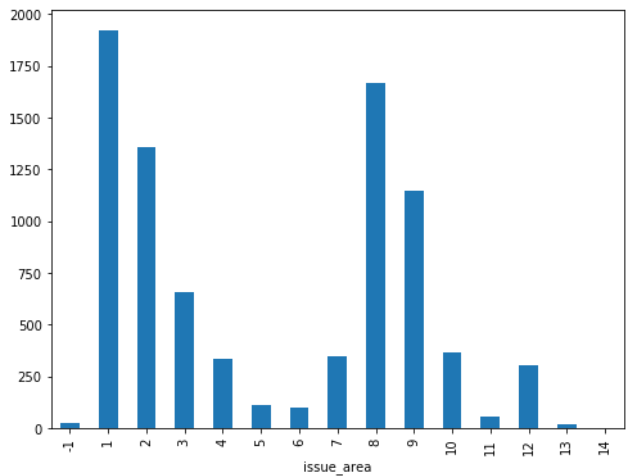


Figure 2. Bi-directional LSTM network

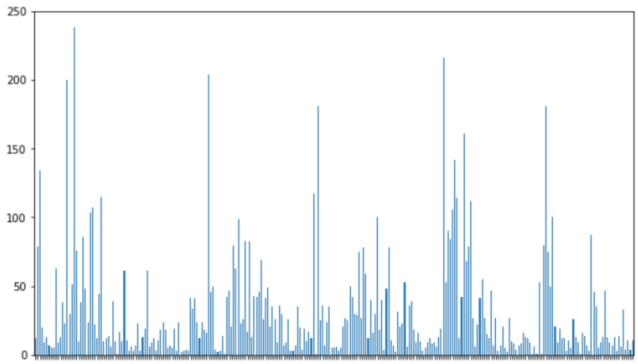
4. DATASET AND EXPERIMENTAL ANALYSIS

4.1 Dataset

The work uses manually categorized Supreme Court Database (SCDB) corpus, from Washington University School of Law [20]. The dataset consists of 8419 SCOTUS legal opinions, classified into 15 legal categories, which are further arranged into 279 sub-categories. Categories are shown on the x-axis and number of documents in the y-axis (Figure 3(a)). The dataset is available in python textacy package. The dataset in textacy package has 11 attributes. Category and sub-category are identified with the name 'issue_area' and 'issue' respectively (Figure 3(b)) in the attributes. For our experiment, the dataset is randomly split into training and testing for building and evaluating the model. A random 80:20 split was chosen for the dataset.



(a)



(b)

Figure 3. a) Legal dataset – categories; b) Legal dataset – sub-categories

4.2 Experimental setup and parameter optimization

Python 3.6 is used to carry out deep learning implementation. The text dataset is vectorized using both supervised as well as unsupervised approach. Embeddings are build using the corpus as well as using publicly available standard model glove. Glove is pre-trained word vectors trained on part of Google news dataset of about 100 billion words. The embeddings are given as input to an evolving parallel CNN architecture. CNN architecture is configured using the optimized hyperparameters obtained through genetic algorithm. At each evaluation, best individual is chosen as per the results of fitness function and is used to update the population. The fitness function consisted of two objectives: accuracy (maximize) and number of features (minimum). The process stops when the termination criterion is met.

4.3 Experimental results

In order to compare the performance of our proposed approach, the following baseline models were defined:

Base (RFC): A random forest classifier model was built using all the features in the form of word vector obtained using count vectorizer from keras pre-processing library. We used random forest as a baseline classifier as it is one of the best classifiers in the traditional machine learning models. A low accuracy score of 0.56 was obtained with RFC.

Base (CNN): A simple CNN model was built using both glove embedding as well as embeddings created from corpus. The hyperparameters values used for building the CNN network are: a convolutional layer with 128 filters with a filter size of 5 and 'relu' activation function, a global max pooling layer and a dense layer with 'softmax' activation function at the end with size equal to the number of labels in the dataset.

Base (LSTM): A unidirectional LSTM model was built using glove embedding as well as embeddings from corpus. LSTM model is built using 64 units, and a dropout of 0.5 issued to avoid overfitting. A dense layer with size equal to that of labels is placed at the end.

For a comparative study a number of different hybrid architectures composed of CNN and LSTM are also evaluated:

CNN + LSTM: In this model, a hybrid of CNN and LSTM is worked upon. The embedding vector was given to the CNN network which learned feature vector. This feature

vector is then passed to the LSTM layer with 64 units. A dense layer with 'softmax' activation function is used at the end.

CNN + Bi-LSTM: A hybrid network consisting of CNN trained with embedding vector and 'relu' activation function and a bi-directional LSTM recurrent neural network followed by a dense layer at the end is also evaluated. A bi-directional LSTM network captures the word context from both left and right side.

(E-CNN) + Bi-LSTM: This represents our proposed model. A CNN architecture whose parameters are optimized using genetic algorithm is hybrid with a bi-LSTM network at the end. The model achieves best accuracy for testing dataset. We present the training and testing dataset evaluation results in Table 2.

In this work a GA optimized CNN is used to get the best feature vector, which is further provided to bi-LSTM network for the classification of legal documents. The performance of a neural network is greatly determined by the settings of its hyperparameters such as the number of convolutional layers, filter sizes, number of neurons, dropout rate and learning rate. In our problem, we have developed a genetically optimized CNN network for having an optimized feature set which provided maximum classification accuracy. As the number of configurations for the CNN hyperparameters is very large, trying to have a configuration manually for the CNN is a cumbersome task. The manually selected settings for our example dataset resulted in an overall accuracy of 78 %. The GA optimised CNN with bi-LSTM outperformed the manually parameterized CNN and gave an accuracy close to 82 %.

It is also clear that in base models, SVM, a simple machine leaning model achieves very low accuracy of 56 %. However, CNN a simple convolutional network achieves an accuracy of around 72 %. LSTM, recurrent neural network achieves very low accuracy of around 23 % for the chosen dataset.

5. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a novel approach for document classification using a hybrid deep learning model. Specifically, we tried to learn a genetically evolving CNN architecture and the effect of using a bi-LSTM network. The proposed model is evaluated by automatically classifying judgment documents from Washington University School of Law Supreme Court Database (SCDB). The dataset is labelled and has 15 categories and 279 finer-grained categories.

As the tuning of hyper-parameters for a CNN is a tedious task, we employed GA method, to search for the best CNN structure. The GA process starts by building an initial population of potential CNN networks and evaluates each network by using the fitness function.

Table 2. Classification accuracy for training and testing dataset for 15 categories

Model	15 Categories	
	Training	Testing
Base Models		
RFC	0.99	0.56
CNN + gloveembedding	0.87	0.67
CNN + embeddings from text corpus	0.99	0.76
LSTM + gloveembedding	0.34	0.37
LSTM + embeddings from text corpus	0.23	0.23
Hybrid Models		
CNN + LSTM + gloveembedding	0.89	0.72
CNN + LSTM + embeddings from the text corpus	0.88	0.74
CNN + Bi-LSTM + gloveembeddings	0.88	0.77
CNN + Bi-LSTM + embeddings from the text corpus	0.86	0.78
E-CNN + Bi-LSTM + gloveembedding	0.89	0.82
E-CNN + Bi-LSTM + embeddings from the text corpus	0.86	0.82

The multiple GA iterations resulted in finding the best model that resulted in maximum accuracy and minimum features. The feature vector obtained from E-CNN was used as input for the Bi-LSTM network and classification is done using the softmax activation function at the end. Our work will allow other researchers in this area to design such hybrid networks for different fields like medicine, biology geology etc. for their particular classification problem.

REFERENCES

[1] Keown, R. (1980). Mathematical models for legal prediction, 2 computer LJ 829. The John Marshall Journal of Information Technology & Privacy Law, 2(1): 29.

[2] Segal, J.A. (1984). Predicting supreme court cases probabilistically: The search and seizure cases, 1962-1981. American Political Science Review, 78(4): 891-900.

[3] Lauderdale, B.E., Clark, T.S. (2012). The supreme court's many median justices. American Political Science Review, 106(4): 847-866.

[4] Lin, W.C., Kuo, T.T., Chang, T.J., Yen, C.A., Chen, C.J., Lin, S.D. (2012). Exploiting machine learning models for Chinese legal documents labeling, case classification, and sentencing prediction. Proceedings of ROCLING, 140.

[5] Aletras, N., Tsarapatsanis, D., Preoţiuc-Pietro, D., Lampos, V. (2016). Predicting judicial decisions of the European court of human rights: A natural language processing perspective. Peer J. Computer Science, 2: e93. http://dx.doi.org/10.7717/peerj-cs.93

[6] Sulea, O.M., Zampieri, M., Malmasi, S., Vela, M., Dinu, L.P., van Genabith, J. (2017). Exploring the use of text classification in the legal domain. arXiv preprint arXiv:1710.09306.

[7] Saravanan, M., Ravindran, B. (2010). Identification of

rhetorical roles for segmentation and summarization of a legal judgment. Artificial Intelligence and Law, 18(1): 45-76. https://doi.org/10.1007/s10506-010-9087-7

[8] Mezghanni, I.B., Gargouri, F. (2016). Detecting hidden structures from Arabic electronic documents: Application to the legal field. In: 14th International Conference on Software Engineering Research, Management and Applications (SERA), IEEE, Towson USA, pp. 75-81. http://dx.doi.org/10.1109/SERA.2016.7516131

[9] Farzindar, A., Guy, L. (2004). Letsum, an automatic legal text summarizing system. Legal knowledge and Information Systems, JURIX, 11-18.

[10] Kim, Y. (2014). Convolutional neural networks for sentence classification. In Proceedings of EMNLP. http://dx.doi.org/10.3115/v1/D14-1181

[11] Khan, A., Baharudin, B., Lee, L.H., Khan, K. (2010). A review of machine learning algorithms for text-documents classification. Journal of Advances in Information Technology, 1(1): 4-20. http://dx.doi.org/10.4304/jait.1.1.4-20

[12] Bansal, N., Sharma, A., Singh, R.K. (2019, May). A Review on the application of deep learning in legal domain. In IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, Cham, pp. 374-381. http://dx.doi.org/10.1007/978-3-030-19823-7_31

[13] Spaeth, H.J., Epstein, L., Martin, A.D., Segal, J.A., Ruger, T.J., Benesh, S.C. (2017). 2017 supreme court database, version 2017 release 01. 2017.

[14] Kotzias, D., Denil, M., Freitas, N., Smyth, P. (2015). From group to individual labels using deep features. Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 597-606. http://dx.doi.org/10.1145/2783258.2783380

[15] Conneau, A., Holger, S., Barrault, L., Lecun, Y. (2017). Very deep convolutional networks for text classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, 1: 1107-1116. http://dx.doi.org/10.18653/v1/E17-1104

[16] Dahou, A., Elaziz, M.A., Zhou, J., Xiong, S. (2019). Arabic sentiment classification using convolutional neural network and differential evolution algorithm. Computational Intelligence and Neuroscience, 2019: 16 pages. http://dx.doi.org/10.1155/2019/2537689

[17] Zhang, X., Zhao, J., LeCun, Y. (2015). Character-level convolutional networks for text classification. In Advances in Neural Information Processing Systems, pp. 649-657.

[18] Liu, L., Liu, K., Cong, Z., Zhao, J., Ji, Y., He, J. (2018). Long length document classification by local convolutional feature aggregation. Algorithms, 11(8): 109. http://dx.doi.org/10.3390/a11080109

[19] Kalchbrenner, N., Grefenstette, E., Phil, B. (2014). A convolutional neural network for modelling sentences. Proceedings of the Conference 52nd Annual Meeting of the Association for Computational Linguistics, ACL, 1: 655-665. http://dx.doi.org/10.3115/v1/P14-1062

[20] Dauphin, Y.N., Fan, A., Auli, M., Grangier, D. (2017). Language modeling with gated convolutional networks. Proceedings of the International Conference on Machine Learning (ICML), pp. 933-941.

[21] Qiu, X., Huang, X. (2015). Convolutional neural tensor network architecture for community-based question answering. In Proceedings of the 24th International Conference on Artificial Intelligence, AAAI Press: Buenos Aires, Argentina, pp. 1305-1311.

[22] Dong, L., Wei, F., Zhou, M., Xu, K. (2015). Question answering over freebase with multi-column convolutional neural networks. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, 2015, pp. 260- 269. http://dx.doi.org/10.3115/v1/P15-1026

[23] Yin, W., Yu, M., Xiang, B., Zhou, B., Schutze, H. (2016). Simple question answering by attentive convolutional neural network. Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers 2016, pp. 1746-1756.

[24] Weng, W.H., Wagholikar, K.B., McCray, A.T., Szolovits, P., Chueh, H.C. (2017). Medical subdomain classification of clinical notes using a machine learning-based natural language processing approach. BMC Medical Informatics and Decision Making, 17(1): 155. http://dx.doi.org/10.1186/s12911-017-0556-8

[25] Tai, K.S., Socher, R., Manning, C.D. (2015). Improved semantic representations from tree-structured long short-term memory networks. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. http://dx.doi.org/10.3115/v1/P15-1150

[26] Wang, J.H., Liu, T.W., Luo, X., Wang, L. (2018). An LSTM approach to short text sentiment classification with word embeddings. In Proceedings of the 30th Conference on Computational Linguistics and Speech Processing (ROCLING 2018), pp. 214-223.

[27] Lai, S., Xu, L., Liu, K., Zhao, J. (2015). Recurrent convolutional neural networks for text classification. In Twenty-ninth AAAI Conference on Artificial Intelligence.

[28] Liu, P., Qiu, X., Huang, X. (2016). Recurrent neural network for text classification with multi-task learning. arXivpreprint arXiv:1605.05101.

[29] Young, S.R., Rose, D.C., Karnowski, T.P., Lim, S.H., Patton, R.M. (2015). Optimizing deep learning hyper-parameters through an evolutionary algorithm. in Workshop on Machine Learning in High-Performance Computing Environments. ACM, 2015, pp. 4:1-4:5. http://dx.doi.org/10.1145/2834892.2834896

[30] Suganuma, M., Shirakawa, S., Nagao, T. (2017). A genetic programming approach to designing convolutional neural network architectures. In Proceedings of the 2017 Genetic and Evolutionary Computation Conference. Berlin, Germany: ACM, 2017, pp. 497-504. http://dx.doi.org/10.1145/3071178.3071229

[31] Loshchilov, I., Hutter, F. (2016). CMA-ES for hyperparamete roptimization of deep neural networks. arXivpreprint arXiv:1604.07269.

[32] Dahou, A., Elaziz, M.A., Zhou, J., Xiong, S. (2019). Arabic sentiment classification using convolutional

neural network and differential evolution algorithm. Computational Intelligence and Neuroscience, 2019. http://dx.doi.org/10.1155/2019/2537689

[33] Xie, L.X., Yuille, A. (2017). Genetic CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, pp. 1379-1388. http://dx.doi.org/10.1109/ICCV.2017.154

[34] Socher, P.R., Manning, C.D. (2014). Glove: Global vectors for word representation. In EMNLP, 2014. http://dx.doi.org/10.3115/v1/D14-1162

[35] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J. (2013). Distributed representations of words and phrases and their compositionality. CoRR, vol. abs/1310.4546, 2013. [Online]. Available: http://arxiv.org/abs/1310. 4546

[36] LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. Nature, 521(7553): 436. http://dx.doi.org/10.1038/nature14539