Arabic Sentences Classification via Deep Learning

Dania Sagheer, PhD
student
Dept. Artificial Intelligence and Natural Languages
Faculty of Informatics Engineering, Aleppo university

Fadel Sukkar
Professor
Dept. Artificial Intelligence and Natural Languages
Faculty of Informatics Engineering, Aleppo university

ABSTRACT
This paper presents a Convolutional Neural Network CNN Models to classify Arabic sentences into three topics. These sentences are derived from Essex Arabic Summaries Corpus (EASC) corpus, tokenized to words and transformed to sequences of word indices. All sequences are padded to be in the same length. The models of Convolution Neural Network are built on top of word embedding layer. The word embedding layer is either pre-trained or jointed into the model. Dropout and l2 weight regularization are used to overcome the overfitting during training. The CNN models achieve high performance in accuracy for Arabic sentences classification.

General Terms
Natural Language Processing, Deep Learning

Keywords
Classification, Convolutional neural network, Word Embedding

1. INTRODUCTION
Recently days the information on Internet is increased rapidly, so the need of automatic systems is increased, these automatic systems process the human language and try to understand it. These systems seek to be interactive systems as auto abstractive summarization [1] and the classification. In natural language processing NLP, text classification task is considered important step for text understanding [2]. Text classification needs text processing and analysis, because computer sees the text as group of symmetric characters without meaning differentiation between these characters.

Text classification is defined as assigning each text with predefined set of classes. Text classification is divided into two types: single-label classification and multi-label classification. In the single-label classification, each text belongs to only single class, whereas the multi-label classification assigns each text to more than one class. If the set of classes consist of two classes, then the text classification is called binary classification. However, if the set contains multi classes, text classification is called multi classification [3]. There are many applications of text classification such as: topic identification of document or sentence, detection of the book author, sentiment analysis, spam classification, and others [4].

Convolutional neural networks (CNN) are a category of the Deep neural networks that employs a mathematical operation called convolution. Convolution is a specified kind of linear operations. CNN are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers [5]. CNN has proven effectiveness in computer vision, classification and has begun applied in Natural language processing NLP [6]. In natural language processing the text is tokenization into words, characters, or bag of words, and the words, characters, or bag of words are called tokens. Although tokens are available, CNN models don't take it as input, they only work with numeric tensors. The numeric tensors are resulted by vectorization. Vectorization associates the generated tokens with the vectors. There are multiple ways of vectorization such as: one hot encoding and word embedding [7]. One hot encoding associates every token with a unique integer index and then turns this integer index into a binary vector of vocabulary size. The vector is zeros except at the index of the token will be one, so the vectors in the one hot encoding are sparse whereas word embedding are low dimensional of floating point vectors. Word embedding are learned from data. Word embedding can be either a part of the main task model or loaded as pre-trained into the model.

The researchers used the convolutional neural networks in text classification applications and Semantic clustering, so convolutional neural network is used to model of short texts [8]. Extracting lexical and sentence level features were implemented by a convolutional deep neural network, then these features are fed into a softmax classifier to predict the relationship between two marked nouns [9]. The Simple CNN with one layer of convolution on top of word vectors and little hyper parameters tuning can do sentence-level classification tasks [10]. The models for task of sentences classification requires practitioners to specify an exact model architecture and set accompanying hyper parameters, including the filter region size, regularization parameters, and so on [11]. To achieve text classification, we need to represent the semantic content of a sentence as it is presented in the research [12]. In the research [13] the method learns embedding of small text regions from unlabeled data for integration into a supervised CNN to achieve better results on text classification. The study in [14] presents empirical character-level convolutional networks for text classification. character-level CNN is an effective method, but it needs a very large size of data set. Also the research in [15] work in character-level, it increases the depth of CNN to increase the performance. The research [16] provides effective use for word order instead word embedding to feed into CNN for text categorization. In research [17] model progressively builds a document vector by aggregating important words into sentence vectors and then aggregating important sentences vectors to document vectors. The research in [18] builds an Arabic word embedding model and convolutional neural network (CNN) for sentiment classification.

In this paper, the model of convolutional neural network jointly on top word embedding layer and the two model of convolutional neural network on top pre-trained word embedding are suggested for Arabic sentences classification.

The paper is organized as follows: The Data set is briefly reviewed in Section 2. Section 3 introduce word embedding, Section 4 describes convolutional neural network, section 5 shows the methodology, and section 6 demonstrates the experimental results. Finally, conclusion is offered in Section 7.

2. Data Set

Essex Arabic Summaries Corpus (EASC), university of Essex, contains 153 Arabic Articles for summarization task [19], in this research, a set of 500 Arabic sentences derived from EASC is collected as the dataset. The dataset includes three topics: Music & art, environment, and finance. Each sentence belongs to only one topic. Each topic has nearly 160 Arabic sentences, so the dataset is considered balanced dataset. The dataset includes about 15000 words. Sentence length ranges between 30 to 47 words.

3. WORD EMBEDDING

There are many methods to get vectorization, which transforms each token to a vector. Previously the statistical methods are used to do the vectorization as TF-IDF which calculates term frequency across the one document and the all documents in the dataset. In vector space model one hot encoding method creates a vocabulary of most common words in the dataset and gives index for each word in the vocabulary, then builds vectors of zeros in vocabulary size and gets the value of one in the word index of the vector. One hot encoding presents vectors with high dimensional and all the values of the vector are zeroes except a single value, so it is considered sparse vectors. Word embedding can compress the vectors into low dimensional of float values, word embedding learns the values of the vectors from the data. The most common of word embedding methods is word2vec which has become important in 2013 when Mikilov et al from Google [20] suggested their method in word2vec. Word2vec model is based on tow strategies: skip- gram and continuous bag of words CBOW. The model trains on so large dataset and learns the vectors of the words from the context of the words in the dataset. In skip-gram strategy the model predicts the context word from the target word while CBOW predicts the target word from the context words. Word2vec shows the similarity between word vectors according to the semantic relationship among these words, and reflects some of the syntactic and morphology analyzes into mathematical relations between vectors.

Word embedding models can be used as pre-trained in top of other models for many tasks.

The research [21] presents AraVec Arabic word2vec models for the Arabic language using three different dataset resources: Wikipedia, Twitter and Common Crawl webpages crawl data, the models are built in the same strategies in Mikilov word2vec, skip-gram and CBOW. Gensim library is used to implement AraVec models. AraVec models are publicly available. AraVec models proven ability to capture similarity among word vectors, and are used as pre-trained in top of the models for Arabic NLP tasks.

4. CONVOLUTIONAL NEURAL NETWORK CNN

CNN is a category of Deep learning neural network. It achieves power effectives in computer vision as feature extraction from images and image classification, it is also successfully begun applied to NLP. The importance of CNN is shown in its automatic extraction of features without human supervision. The main block of CNN is convolutional layer that it uses the convolution operation in its computational. Convolution operation refers to mathematically operation that merges two sets of information. CNN consists of feature extraction part and classification or regression part. The feature extraction part detects the features from the input through the convolution layers, the classification part contains of the fully connected layers and the output layer. CNN model uses the convolution operation in the convolution layers to reduce the parameters in the traditional neural networks, because the neurons in the one layer do not connect to all the neurons in the next layer but only to a small region of it [22]. The convolution is performed on the input data with the use of a filter or kernel to produce a feature map. The filters represent the weights in the convolution layer [23]. Each filter indicates the features are searched about them in the input data. A convolution operation is applied by sliding the filter over the input data. At every location in the input region, an element-wise multiplication is performed and the results are summed onto the feature map. The region of the input where the convolution operation takes place is called receptive field. The size of the receptive field equals to the size of the convolution filter. It is recommended to use odd filter size. The convolution layers perform multiple convolution on an input data, each convolution uses a different filter and results a distinct feature map, all these feature maps are stacked together to form the output of the convolutional layer, so the filter count is the dimension of the convolutional layer output. The step for moving the filter over the input is called stride and the value of the stride by default is 1. When the stride size is increased, we get less overlap between the elements of the input data. The feature map size is smaller than the input size because the filter is contained in the input, if we want to retain the feature map size as the same size of the input data, the padding of zeros is used. Zero padding is expressed by the following equation:

$$\text{ZeroPadding} = \frac{\text{K} - 1}{2}$$

Where K is the filter size.

the following equation is used to calculate the feature map size:

$$\text{O} = \frac{(\text{W} - \text{K} + 2\text{P})}{\text{S}} + 1$$

Where O refers to the size of the output feature map, w is the size of the input, K is the filter size, P is the padding, and S is the stride. Lower feature maps detect simple features from the input data because they get less information, the deeper feature maps combine more information from previous feature maps so deeper feature maps can detect complex features. The output of the convolution will be passed through the nonlinear activation function. This could be the ReLu activation function. This means that the ReLu is applied to the feature maps. The function of ReLu is illustrated by the equation:

$$\text{F}(\text{x}) = \max(0, \text{x})$$

So the only negative values change to 0, other values retrain as these. ReLu function achieves better training of the model, because it increases the nonlinearity of the model. ReLu function helps to alleviate the vanishing gradient problem, which is the issue where the lower layers of the model train very slowly because the gradient decreases exponentially through the layers, so by applying ReLu function the performance of the training gets faster and better. Each convolution layer detects a level of the features. When we build the CNN models, we have to adjust a set of hyper

parameters. The hyper parameters of the CNN are convolutional layers count, filter size, filter count, stride and padding. The fully connected layers in the classification part expects a 1D vector so the output of the convolutional layers is flattened to a vector. this vector becomes the input of the fully connected layer. the final output will be reduced to a single vector of probability score by using softmax function on the output layer. CNN is trained by backpropagation with gradient descent or other learning methods.

5. METHODOLOGY

This research is fall in a single-label multi classification, that every sentence has to be belongs to only one class from three classes. Each class represent the topic of sentence.

5.1. Tokenization

The text is tokenized to words, according to a regular expression, regex, that is a sequence of characters. Regex in this research consists of the punctuation marks group, numbers group and Arabic alphabet group containing characters of vocalization. When the search algorithm finds sequence of the characters from regex group, it tokenizes it. All number tokens are referred to the same token "number". All the punctuation marks are referred to the same token "sign".

5.2. Word Indices

Each token that it is a word, is given an index, this index indicates the most common of this word in all text of the data set. the number of the most common words in the data set equals to 15000 words. These most common words are called vocabulary.

5.3. Pad Sequences

The sentences in the data set are transformed to the sequence of word indices. All sequences have the same length. The identical length is the number of the words in the longest sentence in the data set that equals to 47 words, so the sentences which their length less than 47 are padded by zeros.

5.4. Model

The dimension of the model input is a samples × sentence length. The samples represent the sentence numbers in the training set.

The model is a stack sequential of the layers that it consists of the embedding layer, three convolutional layers, flatten layer, one fully connected layer and the output classifier layer. The vocabulary of the data set is built in 1500 most common words. Each word gets the index, then these indices input to the embedding layer to transform each of them to a float point vector. The convolutional layers apply nonlinear ReLu activation function and have hyper parameters as filter size, filter count, stride and padding. The flatten layer is used to flatten the output of the convolutional layers to a 1D vector which inputs to the fully connected layer. The output layer contains of three neurons because the task is to classify sentences into three topics. The output layer applies the softmax activation function. Each output neuron gives the output a score of probability to be the one of the three topics. The labels of the topic classes are encoded as binary category with one hot encoding method.

The data set is split by random selection into training data and test data according to the training rate.

In this research, three models are built. In the first model, word embedding layer is trained jointly in the top of the model. It is trained to give word vectors which minimize the loss function by using the optimization method to achieve the main task of this model. The main task is the sentence classification. the input to this embedding layer is used to index a table with the embedding vectors The dimension of the word vectors is hyper parameter, and it is adjusted to be 256 dimension, the hyper parameters for convolutional layers are 512 for filter count, and 3 for filter size, so the convolution filter slide over the input and pick three words, each word is represented by 256 features. The stride equals to one and the padding is not used. The size of the output feature map from the first convolutional layer is:

$$O = \frac{W - K + 2P}{S} = \frac{47 - 3 + 0}{1} + 1 = 45$$

Where W is the sentence length represented the input, K is filter size, p is padding and s is stride. The dimension of this output feature map is 45× 512. This feature map is inputted to the second convolutional layer, and the size of the second output feature map is:

$$O = \frac{W - K + 2P}{S} = \frac{45 - 3 + 0}{1} + 1 = 43$$

The dimension of this output feature map is 43× 512.

The input of the third convolutional layer is the second output feature map, and the output feature map is:

$$O = \frac{W - K + 2P}{S} = \frac{43 - 3 + 0}{1} + 1 = 41$$

The dimension of the third feature map is 41×512.

The fully connected layer receives a 1D dimension vector, so the flatten layer flatten the dimension of the third convolutional layer output into a 1D vector 41*512=20992 then the output layer reduces the large numbers 20992 neurons to three neurons.

However, in the second model, AraVec Arabic word2vec is used as pre-trained word embedding layer so the weights of AraVec are loaded into the embedding layer, and the training parameter is initialized to a value false. Pre-trained AraVec Arabic word vectors have dimension of 300. They trained by the CBOW strategy on Wikipedia data resources. The convolutional layers' hyper parameters are adjusted. the filter size is selected to be three and the filter count is selected to be 500. The size of the feature maps is calculated Similar to the one in the first model. But the dimension is to be 500 instead of 512, so the fully connected input is 500*41=20500 neurons and also reduced to three output neurons.

Finally, the third model also uses pre-trained Aravec Arabic word2vec model, but it gives the words not present in the word2vec data set the mean of the other word vectors with a small random offset. Regularization is applied to these word vectors with 0.05 value to prevent the values of these vectors of memorizing from the other vectors. The convolutional layers and the fully connected layer are the same in the second model.

Pre-trained AraVec Arabic word2vec is proven effectiveness in capturing some semantic relations between words, so in the last two models it gives effectiveness in the sentences classification.

Generally, the filters in the convolutional layers in these models detect the features from the Arabic sentences through three levels. These features feed into the classification part to perform the classification task effectively.

The models are trained by using categorical cross entropy loss function to minimize the errors between the output class and the target class.

The filters represented weights are updated to learn through optimization function Adam. The learning rate and epochs number are adjusted in the models to give high accuracy in the classification results.

5.5. Overfitting

overfitting is happened during the training stage. Overfitting is when the model learns patterns that are specific to the training data but that are irrelevant when it comes to new data. so the training loss function curve keeps going down but the validation loss curve grows up.

to avoid this overfitting, dropout layers and weight regularization are used.

Dropout in the first and the second models is applied to the convolutional layers' neurons where some neurons in these layers randomly are disabled to reduce the dependences between the neurons. The rate of a dropping neurons is selected to 0.7.

weight regularization gets the performance of the model more generalization and resistance of overfitting.

Weight regularization put constraints on the complexity of a model by forcing the weights with large values to take only small values, that makes the distribution of weight values is more regular.

In the third model L2 regularization technique is used. L2 regularization technique adds cost which is proportional to the square of the value of the weight coefficients, so the weights are become regular. This cost value is called regularization factor and is assigned 0.01 of the all convolutional layers in these models.

6. EXPERIMENTAL RESULTS

These models are implemented in Python language by using libraries like Keras, numpy, re, Sklearn, Gensim and tensorflow in the back end.

The architecture of the models is built by Keras library. Numpy, panda, csv and re libraries are used for text preprocessing as text reading and text tokenization. The training and testing set are selected using Sklearn library to split the data set to training data and testing data randomly according to the specific training rate. Gensim library is used to load the pre-trained AraVec Arabic word2vec.

The implementation is achieved by GPU 940 MX personal computer.

In the first experiment, where the first model is implemented. embedding layer is jointly into the model, the embedding layer is not trained well, because the data set size is small and the main task of the training is text classification not word embedding. the model is trained on 14 epochs and The learning rate of the Adam function is 1e-4. The batch size of training is 30. In this experiment, the training set is 75 % of the data set. To avoid overfitting, dropout is used in embedding and convolutional layers. dropout layer selects randomly dropped neurons with dropping rate 0.7.

The implementation of this model takes 21.23 seconds.

As we notice in the (figure 1), the training loss decreases with every epoch until approaching zero value. But the validation loss remains without modification until the seventh epoch, then the validation loss starts with decreasing to reach a six value at the end of the training, that it is called overfitting, where a model that performs better on the training data isn't necessarily a model that will do better on data it has never seen before. Although dropout is added to this model to overcome the overfitting, the model still has overfitting.

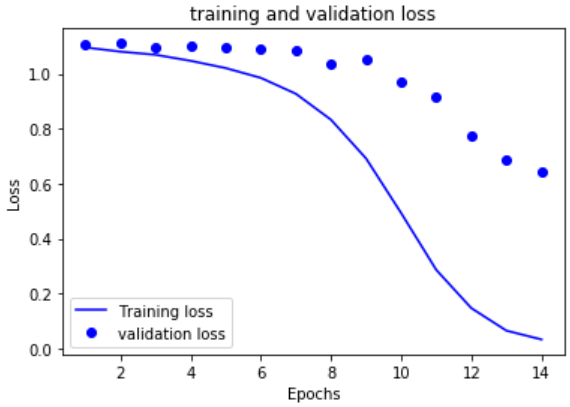

Fig 1: training and validation loss in the first model

The training accuracy is reached 99% as shown in (figure 2) but this accuracy doesn't be generalize to the new data in the validation set, so the validation accuracy reaches only to 76 % at epoch 14 in the end of the training. So the second model is suggested to perform better accuracy.

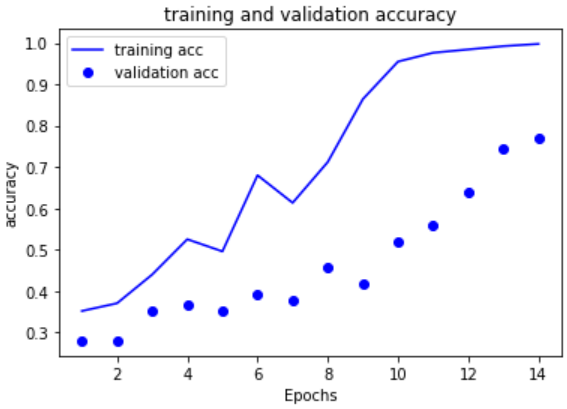

Fig 2: Training and validation accuracy in the first model

In the second experiment, where the second model is implemented.

AraVec pre-trained Arabic word2vec model is used as word embedding in this model to give word vectors. These word vectors capture the semantic relations between the words according to these contexts in the word2vec model data set. So the results of the sentence classification get better. Dropout also added to the model in the convolutional layers and with the same rate 0.7. the model is trained on 10 epochs and The learning rate of the Adam function is 1e-4. The batch size of the training is 30.

The experiment is achieved by using tow training rate, the first training rate is 75% of the data set, and the second one is 50% of the data set. for retesting the results, the seed of keras is constant and is 1337.

In the state of the training rate 75%, the implementation of this model takes 28 seconds. In the (figure 3) the training loss approaches zero value, and the validation loss decreases but it

can't reach less than 4.5, although the improvement is done to the model, overfitting is still shown in the model.

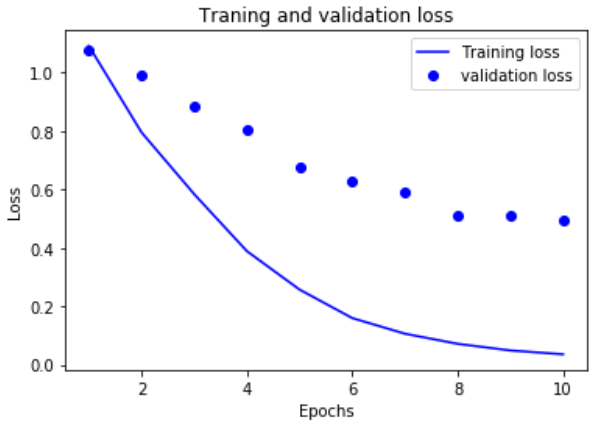

Fig 3: training and validation loss in the second model, training set is 75% of the data set

the performance accuracy of this model is rose from its in the previous model, the training accuracy approaches about 99 % from the epoch six, but the validation accuracy increases until reach more than 80% at epoch eight then decreases a little to stabilize to the 80%.

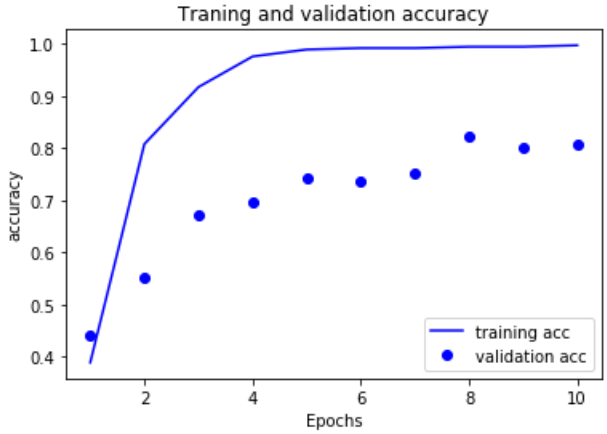

Fig 4: training and validation accuracy in the second model, training set is 75% of the data set

When the training rate is decreased, the training data will be reduced. So the model performance will decrease. The (figure 5) shows the training and validation loss by splitting the data set into 50% for the training data and 50% for the testing data. The training loss decreases gradually until be about 0.05 in the end of the training. Whereas the validation loss can't decrease less than 0.7. the performance accuracy of the training still reaches almost 99%, but the validation accuracy shows decrease in the performance. As illustrated in (figure 6) validation accuracy approaches to 71 % by training 50% of the data set. The time for implementing this model takes 23.75 seconds

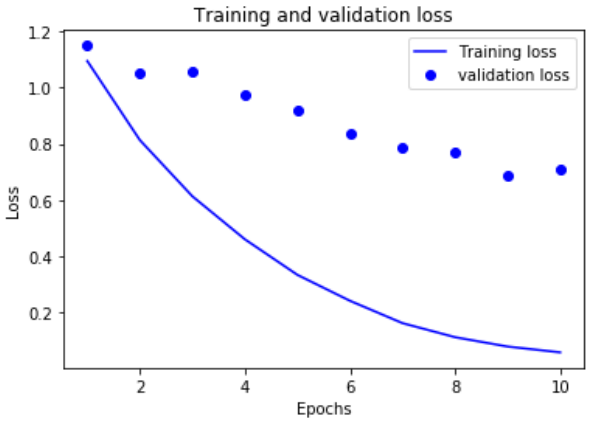

Fig 5: training and validation loss in the second model, training set is 50% of the data set

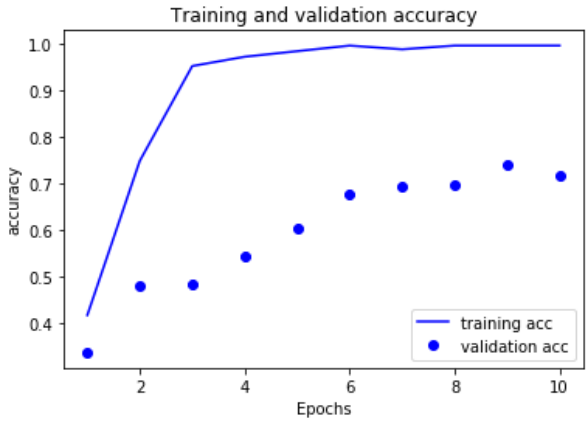

Fig 6: training and validation accuracy in the second model, training set is 50% of the data set

Finally, the third experiment is done by implement the third model. AraVec Arabic word2vec is used as word embedding as in the second model, but The words which are not present in the AraVec model data set, a vector of the mean of the other word vectors is suggested with random offset and regularization factor to represent these word vectors. Dropout layers are not used in this model whereas the l2 regularization with factor 0.01 is added to the convolutional layers to improve the performance and avoid overfitting.

The model is trained on 8 epochs and The learning rate of the Adam function is 1e-3. The batch size of training is 30. the seed of keras is constant and is 1337 for helping in retesting the results.

The experiment is implemented by using tow training rate, the data set is split by 75% rate and by 50% rate of the data set.

When the training set forms the 75% of the data set, the model takes 14.60 seconds for implementation. Training and validation loss function in this state converge as shown in (figure 7). That means the model can overcome the overfitting.

The validation accuracy increase gradually to approach 91 % at the epoch 8 in the end of the training, as illustrated in (figure 8).

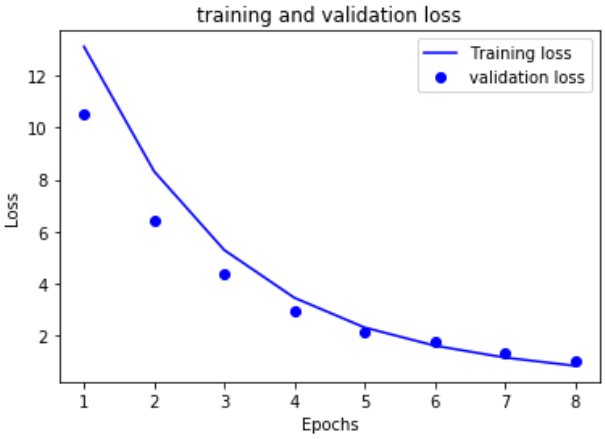

Fig 7: training and validation loss in the third model, training set is 75% of the data set

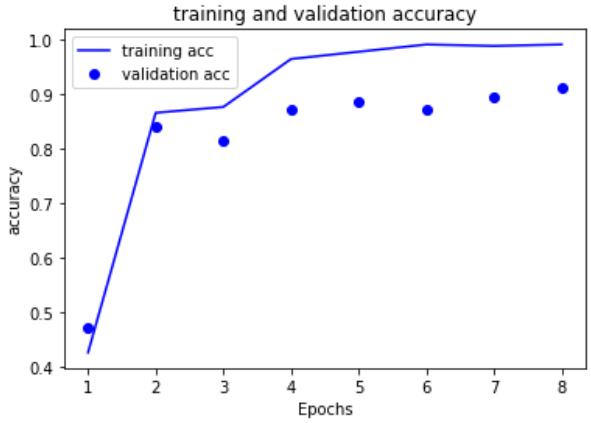

Fig 8: training and validation accuracy in the third model, training set is 75% of the data set

The performance is reduced a little, when the experiment is achieved by using only 50% of the data set for training. However, the training and validation loss functions converge to approach a value about 1.5 as shown in (figure 9).

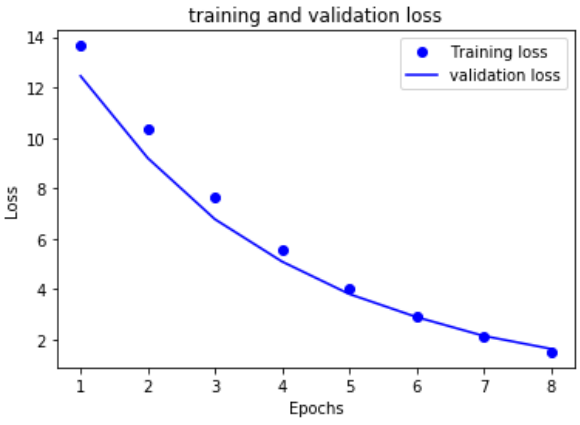

Fig 9: training and validation loss in the third model, training set is 50% of the data set

The accuracy performance decreases a little to reach about 87%. (Figure 10) shows the training and validation accuracy. The training accuracy peak to values more than 95% from the epoch four but the validation accuracy values increase to more than 75% from the three epoch to reach 87% at the eight epoch in the end of the training.

This experiment for implementing this model takes 15.4 seconds.

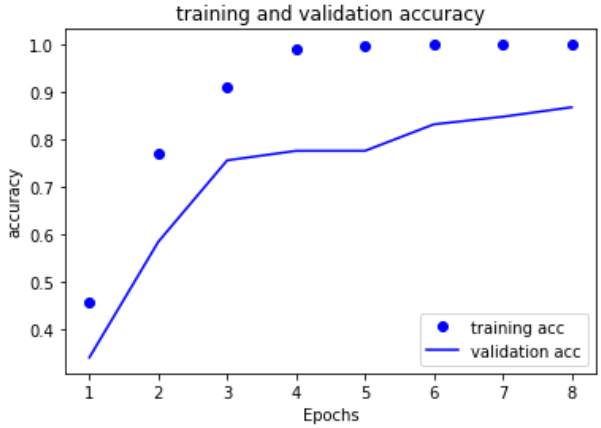

Fig 10: training and validation accuracy in the third model, training set is 50% of the data set

7. CONCLUSION

Three models of convolutional neural network models are implemented in top of embedding layer. The embedding layer either be jointly into the model, or be pre-trained AraVec Arabic word2vec model. The vectors for unknown words in AraVec dataset are created by the mean of the other vectors plus small random offset and regularization factor to get the distribution of these vectors are normal.

Overfitting is avoided by using two methods: dropout layers and l2 weight regularization.

Convolutional neural network models in top of word embedding layer proven powerful and achieve high performance accuracy in the NLP task, Arabic sentences classification.

8. REFERENCES

[1] Sagheer Dania, Sukkar Fadel, "A hybrid Intelligent System for Abstractive summarization", International Journal of computer Applications, vol (168),No (9), June, 2017.

[2] Xiang Zhang,Yann LeCun ,"Text understanding from scratch", arXiv:1502.01710v5 [cs.LG], Apr, 2016.

[3] Aris Kosmopoulos, "large scale hierarchical text classification", Ph.D. thesis department of informatics Athens university of economics and business, 2015.

[4] Mrs. Manisha Pravin Mali, Dr. Mohammad Atique, "Applications of Text Classification using Text Mining", International Journal of Engineering Trends and Technology (IJETT), Volume 13, Number 5, Jul 2014.

[5] Ian Goodfellow, Yoshua Bengio, Aaron Courville, "Deep Learning", An MIT Press book, 2016

[6] Marc Moreno, Lopez,Jugal Kalita "Deep Learning applied to NLP", arXiv:1703.03091v1 [cs.CL] 9 Mar 2017

[7] Francois chollet, "Deep learning with python", Manning publication Co, 2018

[8] PengWang, JiamingXu, BoXu,Cheng-LinLiu, HengZhang FangyuanWang, HongweiHao, "Semantic Clustering and Convolutional Neural Network for Short Text Categorization", Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers), pages 352–357, Beijing, China, July 26-31, 2015.

[9] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhouand JunZhao, "Relation Classification via Convolutional Deep Neural Network", Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2335–2344, Dublin, Ireland, 2014.

[10] Yoon Kim, "Convolutional Neural Networks for Sentence Classification", arXiv:1408.5882v2 [cs.CL], 2016

[11] Ye Zhang, Byron C. Wallace, "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification", arXiv:1510.03820v4 [cs.CL], 2016

[12] NalK alchbrenner, Edward Grefenstette, Phil Blunsom, "A Convolutional Neural Network for Modelling Sentences", arXiv:1404.2188v1, [cs.CL], 2014

[13] Rie Johnson, Tong Zhang, "Semi-supervised Convolutional Neural Networks for Text Categorization via Region Embedding", arXiv:1504.01255v3 [stat.ML], 2015

[14] Xiang Zhang Junbo Zhao Yann Le Cun, "Character-level Convolutional Networks for Text Classification", arXiv:1509.01626v3 [cs.LG] , 2016

[15] Alexis Conneau, Holger Schwenk, Yann Le Cun, Lo¨ıc Barrault , "Very Deep Convolutional Networks for Text Classification", arXiv:1606.01781 [cs.CL], 2017

[16] Rie Johnson, Tong Zhang Baidu, "Effective Use of Word Order for Text Categorization with Convolutional Neural Networks", Human Language Technologies: Annual Conference of the North American Chapter of the ACL, pages 103–112, Denver, Colorado, May 31 – June 5, 2015

[17] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, Eduard Hovy, "Hierarchical Attention Networks for Document Classification", Proceedings of NAACL-HLT 2016, pages 1480–1489, San Diego, California, June 12-17, 2016.

[18] Abdelghani Dahou,Shengwu Xiong, Junwei Zhou, Mohamed Houcine Haddoud and Pengfei Duan "Word Embeddings and Convolutional Neural Network for Arabic Sentiment Classification", Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 2418–2427, Osaka, Japan, 2016.

[19] Arabic Corpora. Essex Arabic Summaries Corpus: https://www.essex.ac.uk/linguistics/research/arabic/arabiccorpora/easc.aspx

[20] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffry Dean, "Efficient Estimation of Word Representations in Vector Space", arXiv:1301.3781v3 [cs.CL] 7 Sep 2013.

[21] Abu Bakr Soliman, Kareem Eissa, Samhaa R. El-Beltagy "AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP", 3rd International Conference on Arabic Computational Linguistics, ACLing 2017, 5-6 November 2017, Dubai, United Arab Emirates

[22] Josh Patterson and Gibson, "Deep learning, A practitioner's approach", O'Reilly, Media, Inc., 2017

[23] Hamed Habibi Aghdam, Elnaz Jahani Heravi, "Guide to Convolutional Neural Networks. A Practical Application to Traffic-Sign Detection and Classification", Springer International Publishing AG 2017.