IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

FEW-SHOT TRANSFER LEARNING BASED ON CNN FOR TEXT CLASSIFICATION

1 CHINTALAPUDI SOWNDARYA LAHARI
1M.tech, Department of Computer Science,
Andhra University,Visakhapatnam, AP, India.

ABSTRACT: Most of the Deep Neural Network models like RNN and CNN are data hungry models. We may or may not have the huge dataset available to train those models. Thus, we require a model or an architecture which can use the pre-trained model to train the new model, which comparatively reduces the training time and gives good accuracy. The proposed model is a Convolutional Neural Networks (CNN) based model that is obtained by using few-shot transfer learning with a simple word embedding model to train the given DBPEDIA dataset. With a large DBPEDIA dataset available, the proposed CNN based model aims to extract the local and position-invariant features comparatively faster than the RNN based model. The few shot transfer learning aims to recognize concepts from the pre-trained word embedded models and extracts some common knowledge from the sequence of input data of training dataset. The few-shot transfer learning mechanism leverage the knowledge from the common source domains to special target domain with few data supporting the CNN model to extract the relevant feature efficiently. Thus, the proposed model, Few-Shot Transfer Learning based on CNN model, exhibits significant classification performance compared to other alternative models.

Key-words: CNN model, transfer learning, pooling strategy, deep learning model, word embedding model, text classification, semantic analysis

I. INTRODUCTION

Machine-learning technology aims to provide the system to learn automatically from the experience without using explicit programs. It mainly focuses on developing the computer programs that can access the data and learn by itself. Machine learning is the part of Artificial Intelligence. For real-time Projects Machine learning provides accurate results by developing computer algorithms which can perform on large datasets.

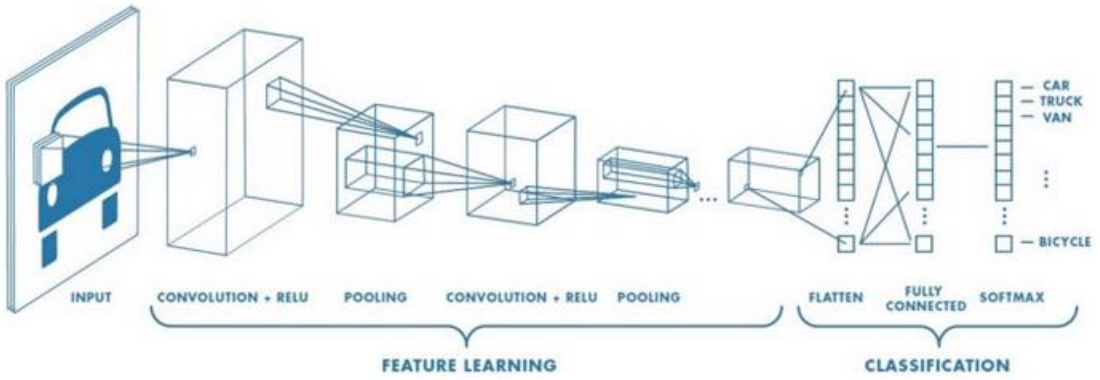
DEEP LEARING: Deep learning allows the computer to solve complex problems, by using artificial neural network nodes which intimates how humans think and learn. It is subset of Machine learning. Deep learning algorithms learn from large amounts of data even if they unstructured, diverse .In this field ,the computer algorithms are effectively used to learn and examine the data It uses the hierarchical level of neural networks to process the data where the nodes are connected like a web.[7]
In general, the programs build analysis in linear way to process the data, whereas deep learning uses hierarchical function to process the data in Non-Linear approach.

2.Related work:

In this paper, we will discuss about CNN which uses transfer learning for text classification.
Here it aims how to extract the different patterns or classification from sequence of input data on huge datasets like DBPEDIA.

2.a CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (CNN) model is a special kind of neural network for data processing. These models to train and test, each input sequence will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply Softmax function is used to categorize the patterns of text with probability values between 0 and 1.



Convolutional networks layers use convolution operation in place of general multiplication of kernel in at least one the CNN layers. Convolutional neural networks (CNN) trained on top of pre-trained word vectors for sentence-level classification tasks. The CNN models include sentiment analysis and question classification.[7]

2.b) Transfer Learning: In simple words, the Transfer learning in CNN means it takes the large dataset on which the model is trained and converts or leverage the knowledge to small datasets. The transfer learning in CNN is a successful attempt over the RNN ,because the model trained is faster on the GPUs.

2.c) Text Classification: Text Classification is the process of categorizing or organizing the text into different patterns or groups where each group is given with some predefined and generic tags based on its content.[9]

In CNN model, the text classification is done by different parallel Convolutional neural networks layers which examine in the original document with different kernel /filter sizes for different multiple layers. This, in effect creates a multichannel CNN to read the text document with n-gram sizes i.e group of words.

2.d) Semantic Analysis: Semantic Analysis is the process of identifying the statements or sentences that are semantically correct based on some content which has similar meaning.

3)MODEL DESIGN: The core building block of neural network is the layers for data processing.

The Proposed model is a Sequential linear Architecture of (Conv1d,ReLu,Maxpool layer),Dropout layer,linear layer. Here the word embedding is performed on vocab-size of 404180,and the length of embedding is 300 as the dataset is English in language. The Pre-Trained word embeddings is embedded over the CNN model to train the model which uses GloVe word embedding.[2]

The Conv1d layer is the core layer in CNN which is exhibited with parameters like no.of input channels/input features and output features of 300 and 100 respectively. The Stride length is taken '1' as it is a text Classification and based on sequence of input data. The outcome data of this layer is passed to ReLu activation function.

The ReLu activation function is a linear function which outputs the input directly by resulting the values either '0' or greater than zero i.e positive no. to train the model and to get better performance.

Once the input feature is done from the previous layer it is passed through Max pool layer. Max Pool layer down samples the feature map by extracting the values which represent s the highest range in the given region or patch. The Max Pool in Text Classification is exhibited by taking the feature in each of the region which is more highlighted .

Here, in the proposed model the Max Pool is performed on maximum sentence length of 30, thus the kernel sizes are 3,4,5 respectively for three cycles of CNN layer. No padding is needed ,dilation is 1.

Then the data is passed to Dropout layer. In neural network Dropout is used or prevents the model from overfitting .The input learning parameter is used ,here learning rate of 0.8% is considered.

Then the Linear layer which is a default layer in CNN. The No.of input features is 300 and output features is 9 based on the label field in the dataset.

The Loss operation used in this model is Cross-Entropy Loss which is used to measure the performance of classification of text which find the probability distribution difference between two consecutive sentence and its loss during the model training.

The Optimizer SGD (Stochastic Gradient Descent) is used and its evaluated by learning rate schedular with step size of 2.

4)Methodology: The proposed model is completed in 5 steps:

4.a) Data Preprocessing: The Data preprocessing is done on DBPEDIA dataset from the Kaggle, collected from different stream pages. From the available 5 fields, we have selected some relevant fields to retrieve text articles as our input dataset. With over 2,40,942 records to work on, we have split the data into train and test in the ratio of 4:1 (train data: 80%, test data: 20%).

The data preprocessing is done by following steps

i) Data cleaning: In this, text and label are differentiated and selected, the null values are ignored. Then the parsed data is used with no missing values for vectorization process.

ii) Vectorization: The parsed input text data is converted into numeric tensors which will behave as inputs to the Deep Learning Model. After Vectorizing the input text the associated numeric vectors are embedded using predefined word embedding global vector for word representation.

4.b) Embedding Preprocessing: As the predefined embedding is an archive, to add it to model, we have followed two steps:

i. Parse the file into map index: Creating a key-value pair foe each of the indices present in the word embedding.
ii. Construct an embedding matrix to pass it to the layer: *embedding_matrix=[max,dim]*

This matrix has a shape of max_words (m) from the input dataset that we are using and embedding_dimension (d), that the key-value pair can acquire.

4.c) Building the network model: The proposed model follows Sequential network model architecture, where each node is connected to adjacent node in the layer. The activation function, also known as tensor operations, defines the behaviour of output from the node for the given input or set of inputs. When activated, it helps the neural network model to reduce the learning transformation of tensors by deciding which information from the input is relevant by transforming them non linearly.

The Activation function used in this model are

i) ReLU

ReLU or Rectified Linear Unit function, defined as:

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

where x is the input, is a non-linear function that back propagates the errors. Depending on the input behaviour, relevant or irrelevant, it activates the neurons making the network parse and easy to compute

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

ii) **Softmax function:**

This function is preferred when the model is applied to an n-dimensional input Tensor re-scaling them so that the elements of the n-dimensional output Tensor lie in the range [0,1] and sum to 1

Mathematically,

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

where x is a large set of output from the previous layer. We have used the **Softmax** function in our output layer of the network for the classification of the text.

4.d) Compiling the network model: The compilation for the proposed model is done in two steps:

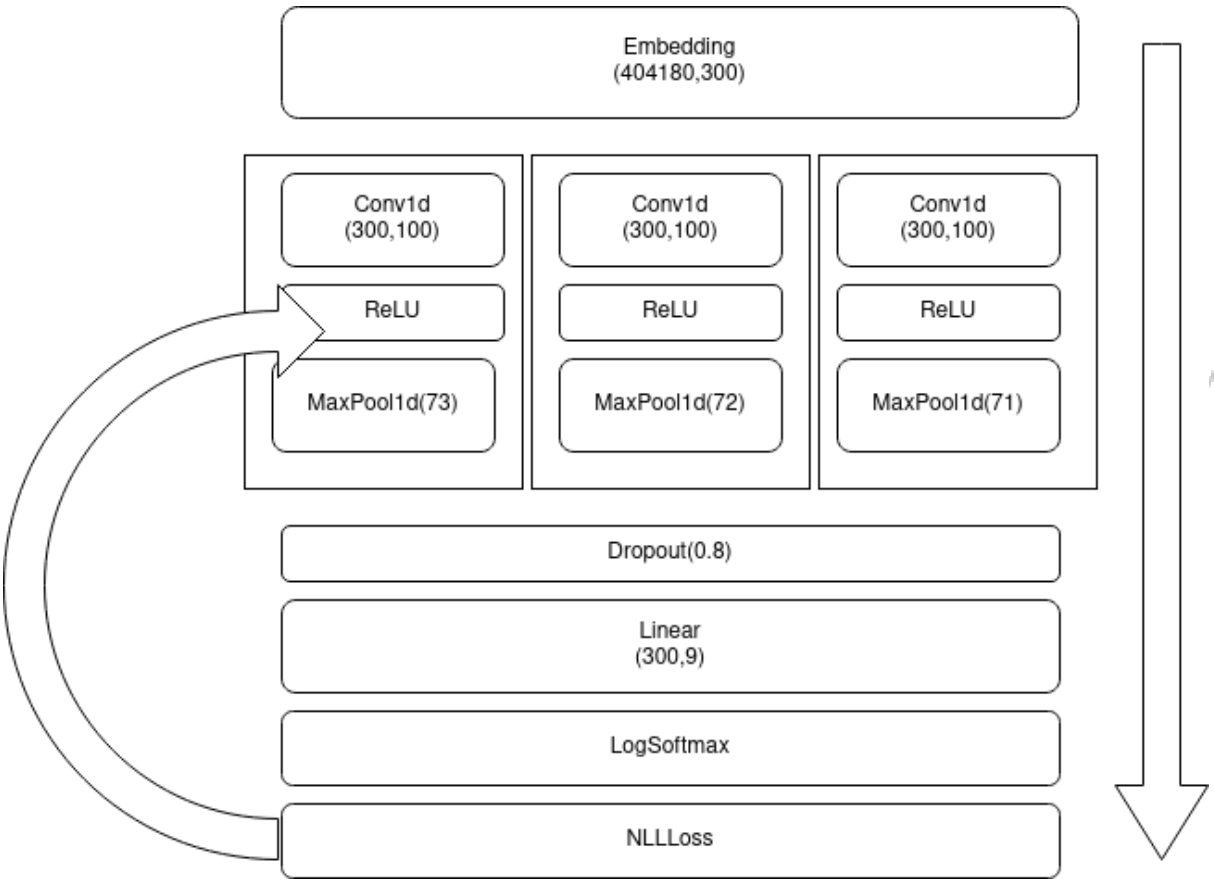
i) Loss function: This function is employed to minimizes the loss in quality of the selected set of parameters based on their induced score with ground truth labels, and for our data set, we have chosen binary cross-entropy to obtain the probabilistic output. The negative log likelihood loss(NLLLOSS) is useful to train a classification problem with C classes.

ii)Optimizer: The optimizer reduces the different cost function that determines the update of the network based on the loss function. SGD (Stochastic Gradient descent) optimizer is used here.

4.e) Classification of text: For the better distribution, we made random shuffle to the dataset, and after that, we divided them into three subsets: training dataset, validation dataset, test dataset. The neural network will then use the training dataset for training classifier model whereas it will use validation dataset for tuning some global parameters of the classifier. The model then uses the test dataset to estimate how well the model performs on new data.
At this final module, we are going to feed the test dataset to the trained model to analyze the prediction accuracy of the model. Based on obtained accuracy, we are going to evaluate different classification parameters and if possible, even enhancing the accuracy.

PROJECT LAYERED DESIGN

The below diagram gives a diagrammatical representation of the proposed layered model.
The proposed model is three Sequential Linear Architecture (Conv1d, ReLu, Max Pooling layer), Dropout layer Linear layer and a Embedded layer over the CNN layers.



I. Embedding layer

II. Convolutional layer(Conv1d+ReLu+MaxPool)

III. Dropout layer

IV. Linear layer

5)Experiments: The experiments are done on large DBPEDIA is an Ontology Dataset classification x with 3 label fields and one text field. Pytorch is used for the implementation where Google Colab is used to execute it which GPU based. By giving the input in a feed forward way into the SWEMs, with 300dimension for the word embedding layer. Then the pre-trained word embedding is used for the modelling the data after data preprocessed.

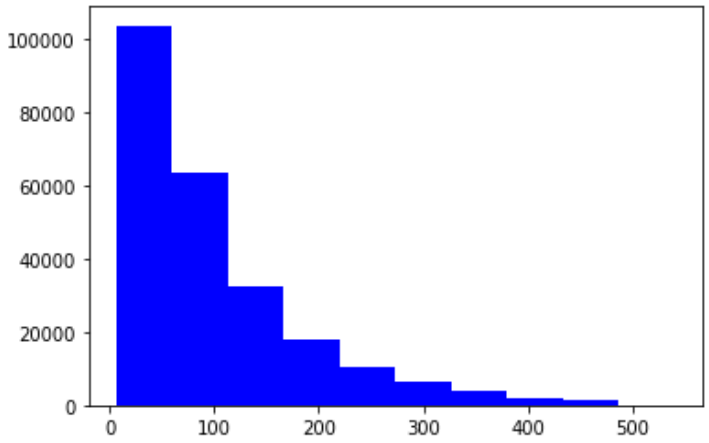
In the data preprocessing, All the url's and links are removed and then it is converted into lower case field values.

Now, the processed data is a .CSV files which are used for training and validation purpose.

After training, the length of each sequence is to be fixed to evaluate the local and position invariant features where the different parameters like kernel size, max length, etc are to be specified.

```
news_len=train_data['text'].str.split().map(lambda x: len(x))
hist(news_len,color='blue')
```

```
(array([103511.,  63151.,  32198.,  17869.,  10410.,   6554.,   3761.,
         2072.,   1268.,    148.]),
 array([  7. ,  60.2, 113.4, 166.6, 219.8, 273. , 326.2, 379.4, 432.6,
        485.8, 539. ]),
 <a list of 10 Patch objects>)
```



Thus, the following table specifies the size of training dataset, validation dataset number of iterations for training and validation dataset with batch size of

Size of training dataset	**240942**
Size of validation dataset	**36003**
Embedding size	**408041**
Training iterator size	**482**
Validation iterator size	**73**

The proposed model architecture consists of three sequential layers of Conv1d with ReLu and Maxpool, Dropout layer, Linear layer. The embedded layer is then added over CNN layer to achieve the feature of transfer learning.

6)RESULTS: Then the layers are passed through several iterations in the feed forward way and its accuracy at each end of the iteration is taken. No.of epochs are taken according to the iterations so that the model is not overfitted. The final output of the model is with the accuracy of 98.4% and 97.82% for training and validation respectively. The below one shows the accuracies for the each epoch.

```
Iter: 301
        Average training loss: 0.05300
        Val Accuracy: 0.9777
Iter: 401
        Average training loss: 0.05442
        Val Accuracy: 0.9785
Epoch: 14
Iter: 1
        Average training loss: 0.03636
        Val Accuracy: 0.9782
Iter: 101
        Average training loss: 0.05510
        Val Accuracy: 0.9785
Iter: 201
        Average training loss: 0.05350
        Val Accuracy: 0.9775
Iter: 301
        Average training loss: 0.05293
        Val Accuracy: 0.9781
Iter: 401
        Average training loss: 0.05183
        Val Accuracy: 0.9785
Final Training Accuracy: 0.9840
Final Validation Accuracy: 0.9782
```

7)CONCLUSION: In this paper, the online news posts from the different stream are downloaded which is in CSV format. It is then converted into python data frame which is further used in training to the model. The word embedding from GloVe is downloaded which is further used to map the numeric tensor token during the process of vectorization. These files serve as training data. A model based upon Deep Neural Network with CNN is used to fit the training data. After multiple epochs, the model became able to take comments from the user and provide relevant responses. The accuracy of the model is tested using various input comments. The responses from the model is also analyzed. The model became fairly decent on providing relevant output to the input comments. However, it also suffered from the drawbacks of consuming more time for training. However, the accuracy can be easily improved by using regional embedding data rather than simple word embedding models.

REFERENCES:

Jason Brownlee (Feb 2018) *How to Use Word Embedding Layers for Deep Learning with Keras,* Available at: *https://machinelearningmastery.com/use-word-embedding- layers-deep-learning-keras/* (Accessed: 18th November 2018).

Jeffrey Pennington, Richard Socher, Christopher D. Manning (2014) *GloVe: Global Vectors for Word Representation,* Available at: *https://nlp.stanford.edu/projects/glove/* (Accessed: Nov 2018).

Felix A. Gers, Jiirgen Schmidhuber,Fred Cummins (1999) 'Learning t o Forget: Continual Prediction with LSTM', *Artificial Neural Networks, 7 10 September 1999, Conference Publication No. 470 IEE 1999,* (), pp. 850-855.

FRANÇOIS CHOLLET (2018) *Deep Learning with Python*, ISBN 9781617294433 edn., 20 Baldwin Road PO Box 761 Shelter Island, NY 11964: Manning Publications Co..

G. Shanmugasundaram (2017) 'Investigation on Social Media Spam Detection', *International Conference on Innovations in information Embedded and Communication Systems (ICIIECS),* 17(978-1-5090-3294-5), pp. .

Julien Fontanarava, Gabriella Pasi,Marco Viviani (2017) 'Feature Analysis for Fake Review Detection through Supervised Classification', *International Conference on Data Science and Advanced Analytics,* 17(978-1-5090-5004-8), pp. 658-666.

Li Deng and Dong Yu (2014) *Deep Learning: Methods and Applications*, Volume 7 Issues 3-4, ISSN: 1932-8346 edn., Tokyo: Foundations and Trends in Signal Processing. Megan Risdal

(2016) *Getting Real about Fake News,* Availableat: https://www.kaggle.com/mrisdal/fake-news/home (Accessed: Nov 2018).

Mykhailo Granik, Volodymyr Mesyura (2017) 'Fake News Detection Using Naive Bayes Classifier', *First Ukraine Conference on Electrical and Computer Engineering,* 17(978-1-5090-3006-4), pp. 900-903.

Suman Ravuri,Andreas Stolcke (2015) 'A COMPARATIVE STUDY OF NEURAL NETWORK MODELS FOR LEXICAL INTENT CLASSIFICATION', *ASRU,* 15(978-1- 4799-7291-3), pp. 368-374.

Sydney A. Barnard, Soon M. Chung, Vincent A. Schmidt (2017) 'Content-based Clustering and Visualization of Social Media Text Messages', *International Conference on Data and Software Engineering (ICoDSE),* 17(978-1-5386-1449-5), pp. .

Vivek Yadav (Jun 23, 2017) *Deep learning setup for Ubuntu 16.04: Tensorflow 1.2, keras, opencv3, python3, cuda8 and cudnn5.1,* Available at: https://medium.com/@vivek.yadav/deep-learning-setup-for-ubuntu-16-04-tensorflow-1-2- keras-opencv3-python3-cuda8-and-cudnn5-1-324438dd46f0 (Accessed: 18th November 2018).