Journal of Intelligence Studies in Business

ISSN: 2001-015X

Journal of Intelligence Studies in Business

Vol. 9, No. 2 2019

Journal of Intelligence Studies in Business

Vol 9, No 2, 2019

Included in this printed copy:

Editor-in-chief:
Klaus Solberg Søilen

A new corpus-based convolutional neural network for big data text analytics

Wedjdane Nahili[a]*, Kahled Rezeg[a], Okba Kazar[a]

[a]LINFI Laboratory, Computer Science, Biskra University, Algeria

*w.nahili@univ-biskra.dz

To cite this article: Nahili, W., Rezeg, K. & Kazar, O. (2019) A new corpus-based convolutional neural network for big data text analytics. *Journal of Intelligence Studies in Business*. 9 (2) 59-71.

Article URL: https://ojs.hh.se/index.php/JISIB/article/view/409

Journal of Intelligence Studies in Business
Vol. 9, No. 2 (2019) pp. 59-71
Open Access: Freely available at: https://ojs.hh.se/

Stevan Dedijer
1911-2004

A new corpus-based convolutional neural network for big data text analytics

Wedjdane Nahili[a]*, Kahled Rezeg[a] and Okba Kazar[a]

[a]*LINFI Laboratory, Computer Science, Biskra University, Algeria*

Corresponding author (*): w.nahili@univ-biskra.dz

Received 3 September 2019 Accepted 27 October 2019

ABSTRACT Companies market their services and products on social media platforms with today's easy access to the internet. As result, they receive feedback and reviews from their users directly on their social media sites. Reading every text is time-consuming and resource-demanding. With access to technology-based solutions, analyzing the sentiment of all these texts gives companies an overview of how positive or negative users are on specific subjects will minimize losses. In this paper, we propose a deep learning approach to perform sentiment analysis on reviews using a convolutional neural network model, because that they have proven remarkable results for text classification. We validate our convolutional neural network model using large-scale data sets: IMDB movie reviews and Reuters data sets with a final accuracy score of ~86% for both data sets.

KEYWORDS Convolutional neural networks, deep learning, natural language processing, NLP, user reviews, sentiment analysis, text classification

1. INTRODUCTION

The main purpose of sentiment analysis is analyzing and understanding expressed human emotion in text data. People are sharing daily thoughts and opinions about everything, and as a result, social media platforms have become the source of varied data, such as reviews of products, movies, and services. With the availability of this content a new type of information is harvested. Understanding 'what people think' and the real meaning of this user-generated data is crucial. Movie review sites such as IMDB, Rotten Tomatoes and Netflix represent an important source of information for researchers. The main reason behind this attention is the fact that valuable knowledge is often hidden behind this content and cannot be easily processed, which has gained increasing popularity among natural language processing (NLP) researchers. Deep learning algorithms are useful when it comes to solving natural language processing problems, and the reason resides in the combination of a large sample of data and a general learning algorithm (Collobert et al., 2011). Several methods can do this with traditional algorithms such as Naive Bayes and Support Vector Machine (SVM). Most of these methods consider the text word by word and classify a sentence as positive or negative by analyzing the words in the text. Sometimes information can be lost by extracting a keyword without another word (Shen et al., 2014). Recently, sentiment analysis research successfully used deep learning. Convolutional neural networks is one of the machine learning models that has archived remarkable results in image recognition and in natural language processing (Collobert et al., 2011).

In order to propose a text classification approach using deep learning, this work introduces a new convolutional neural network architecture for text classification, solving different natural language processing tasks, specifically sentiment analysis. Our model's strengths are its training time and accuracy. In our sentiment analysis model, we utilize convolutional neural networks because they have impressive results in image analysis and classification fields. With their convolution operation they can extract an area of features from global information, and are able to consider the relationship among these features (Y. Kim, 2014). For computer vision, such as image analysis, convolutional neural networks are able to extract pixel data information. This means they can not only extract the pixels one by one, but also the feature information can be extracted piece by piece, where the piece contains multi-pixel data information. Thus, according to (Krizhevsky et al., 2012) when text is transferred into a matrix, it can also be considered to be the same as an image-pixels matrix. As a result, we can do the same operation to the text data to make the input features to the model that can be trained in another effective way (Yoon Kim, 2014).

In this paper, we propose a convolutional neural network (CNN) model to apply sentiment analysis on movie review data in order to predict sentiment orientation. Firstly, as an input to our network model, we use the word2vec proposed by Google to compute vector representations of words and reflect the distance between them. This step leads to initializing the parameters for our CNN model, therefore, efficiently improving the network performance in this particular problem. Secondly, we propose a CNN architecture with three convolution layers with padding, a flatten layer followed by two dense layers. To the best of our knowledge, using this layer architecture in a CNN model with an embedding layer (word2vec) to analyze movie reviews sentiment has not been addressed before in the literature. And finally, to improve the accuracy of our model, we use normalization and dropout layers.

The present work is organized as follows: Section 2 presents a brief literature background with some related concepts used in our approach. Section 3 outlines the related work on sentiment analysis and text classification, with an emphasis on deep learning methods. In Section 4, we present our approach and provide the description for the proposed architecture. In Section 5, the results and experimental setup are explained in detail along with the datasets used to train, test and validate our model and we present and elaborate on the performance using our model, and provide insight into the findings. Finally, we conclude our work and discuss future directions in Section 6.

2. BACKGROUND

2.1 Convolutional Neural Networks

Convolutional neural networks, also known as ConvNets, are a deep learning tool that has gained traction in computer vision applications (S. Srinivas et al., 2016). They were first introduced in Y. LeCun et al., (1989) to recognize handwritten ZIP code in 1989. They were later extended to recognize and classify various objects such as hand-written digits (MNIST), house numbers (P. Sermanet et al., 2012), Caltech-101 (L. Fei-Fei et al., 2007), traffic signs (P. Sermanet et al., 2011), and recently the work of A. Krizhevsky et al. (2012) produced a 1000-category ImageNet data set. The choice of using neural networks to create natural language processing (NLP) applications is attracting huge interest in the research community and they are systematically applied to all NLP tasks (Y. Kim, 2014).

The fundamental idea of CNNs is to consider feature extraction and classification as one joined task. The scope of using this methodology in text analytics has proven to be advantageous in various ways (D. Santos et al., 2014; A. Severyn et al., 2015; S. Srinivas et al., 2016). In deep learning techniques, there is supervised learning, unsupervised learning, hybrid learning and reinforced learning (A. Gibson and J. Patterson, 2017), but supervised learning and unsupervised learning are the most common techniques. The main difference is: in supervised learning, the data is labeled and known prior to training. This technique is suited for classification and regression problems. In unsupervised learning, the data is not labeled, which makes it good for clustering problem where algorithms can find different types of patterns within the unlabeled data (M. Mohri et al., 2012). With machine learning, there is deep structured learning, commonly known as deep learning. It can be used in different learning frameworks such as unsupervised, supervised and hybrid networks, in addition of different classification, regression and vision problems (L. Deng and D.Yu, 2014). A deep learning model can be described as a model of two nodes, where one is an input, and the other an output. Data is sent between these two nodes through the input layer. The data is examined at different levels and features once it is sent onto the hidden layers.

Recently, CNNs have been adopted in natural language processing, sentiment analysis, text, topic and document classification for the following key reasons: CNN can extract an area of features from global information, it is able to consider the relationship among these features (Y. Kim et al., 2014), and text data features are extracted piece by piece and the relationship among these features, with the consideration of the whole sentence, thus, the sentiment can be understood correctly.

2.2 Sentiment Analysis

There are a number of different problems that deep learning is trying to solve. From classification problems where the algorithms assign categories to items, for instance, news categories, and to regression problems where the algorithm gives predictions on real values like a prediction on the stock market (M. Mohri et al., 2012). Another problem is sentiment analysis, also known as opinion mining. Sentiment analysis is an active research field in natural language processing, where people's emotions, opinions, and sentiments towards different entities like products, services, and organizations are studied and analyzed. Sentiment analysis is important for companies, organizations and individual persons (D. Tang, 2018). Companies want to know what people think about their products and services while on the other hand, individual people want to know what others think about a product they are considering purchasing. Daniel Angus stated: "This not only provides insight into what people think about your brand, but it can go a lot deeper. It can expose why people are thinking it."

In sentiment analysis, the goal is to determine whether a given piece of text is positive, negative or neutral. Various work has been done in the field of sentiment analysis in recent years where text is analyzed in several ways. In general, there are three levels of sentiment analysis: document-level, sentence-level and aspect-based level (A. Kharde, 2016).

Document-level: at this level, the analysis takes in consideration that the entire document has only one opinion.

Sentence-level: this level takes in consideration each sentence as containing one opinion and thus, the polarity of the entire document depends on the polarity of the sentences.

Aspect-based level: is also known as feature-based sentiment analysis. At this level, each sentence can contain more than one aspect in order to determine the polarity of the document (A. Kharde, 2016).

The main advantage of deep learning approaches in sentiment analysis remains in the fact that networks train themselves on the same data to learn the structures and context of the data. The data can vary and is often in the form of electronic data collected and made available for analysis. The crucial aspects of the data are the size and quality of the information. The better the quality of the data used in training, the better the results of predicting data in the future (J. Heaton, 2015).

2.3 Natural Language Processing

Natural language processing (NLP) is an industry term for algorithms designed to take a document consisting of symbols and deduce associated semantics (Russell. M, 2011). Research in NLP deals with the application of computational models to analyze text or speech data. Much work has been done in the field of NLP (Mikolov et al., 2013; Ouayang et al., 2015; Houshmand, 2017; Kalchbrenner et al., 2014) in order to allow semantic processing. Sentiment analysis is the research area where NLP algorithms are most often used, due to the amount of available data resulting from shared information on different social media platforms such as Facebook, Twitter, Amazon, Yelp, IMDB and Netflix. Until now, most sentiment analysis work has been done on short texts derived from social media sites. In this work, we analyze review texts because they provide sentiment about products or movies, therefore, when the result of this analysis is applied, it will help companies around the world to improve the decision-making process. Further, to automate sentiment analysis, different approaches have been applied to predict sentiments of words, expressions or documents (Mikolov et al., 2013; Ouayang et al., 2015; Houshmand, 2017; Kalchbrenner et al., 2014). These include NLP and deep learning methods. In our attempt to analyze the sentiment of movie review data and topic classification, we propose a deep learning approach that combines the advantages of available techniques such as CNNs along with NLP basic tasks. The following section reviews and discusses related work in the field of sentiment analysis on reviews with emphasis on deep learning techniques.

3. RELATED WORK

Recently, much work has been done in the field of sentiment analysis in natural language and social network posts. To determine whether a piece of text expresses a positive or negative sentiment, two main approaches are commonly used: the lexicon-based approach and the machine learning-based approach. In recent years, deep learning models have achieved remarkable results in computer vision (Krizhevsky et al., 2012) and speech recognition (Graves et al., 2013). In the area of natural language processing, research on deep learning approaches (Bengio et al., 2003; Mikolov et al., 2013; Yih et al., 2011) has associated learning word vector representations. Although originally invented for computer vision and image analysis, CNNs have proven to be effective for NLP. These models have achieved impressive results in semantic parsing (Yih et al., 2014), search query retrieval (Shen et al., 2014), sentence modeling (Kalchbrenner et al., 2014), and various traditional NLP tasks (Collobert et al., 2011).

Ouayang et al. (2015) proposed a CNN and Word2Vec methodology for movie review sentiment analysis using a dataset from *rottentomatoes.com*. The data set contained 11,855 reviews with five different sentiment classifications (negative, somewhat negative, neutral, positive and somewhat positive). Their CNN model used three different convolution layers with different kernels and each layer was followed by a dropout layer and normalization layers. To evaluate their results, they compared their model against other algorithms/models including Naive Bayes, SVM, Recursive Neural Network (RNN) and Matrix-vector RNN (MV-RNN). The results show that performance is best when it comes to classifying every review into the five different classifications. Their model achieved a test accuracy of 45.4% on the test data set.

Houshmand (2017) compared different neural networks architectures against the Naive Bayes algorithm to see how well they performed on movie reviews from the Stanford Sentiment Tree bank dataset. The results of their study showed similar accuracy between the neural networks used (recurrent, recursive and convolutional neural networks) and Naive Bayes. One interesting thing about the result was the fact that their model's accuracy improved significantly by adding a word vector from Word2Vec to the network. Their model reached an accuracy of 46.4% on the test data while the CNN without a word vector had 40.5% accuracy (Table 1).

Table 1 Corpus-based related work.

	Corpus	Accuracy
Semantic parsing (Yih et al. 2014) CNN model		54%
Sentence modeling/sentiment analysis (Kalchbrenner et al. 2014) DCNN model	SST movie review TREC text retrieval	Binary class 86.8% Fine-grained 48.5%
Sentiment analysis (Ouayang et al. 2015) CNN+word2vec model	Rotten tomatoes movie review	Five classes 45.4%
Sentiment analysis (Houshmand, 2017) CNN model	STT movie reviews	40.5%
Sentiment analysis (Houshmand, 2017) CNN+word2vec model	STT movie reviews	46.4%

Despite the strong empirical performance in (Yih et al., 2014) and the good results in the work of (Mikolov et al., 2013; Ouayang et al., 2015; Houshmand, 2017; Kalchbrenner et al., 2014) we concluded that in (Yih et al., 2014) their system has no room for improvement because the corpus derived from the *WikiAnswers* data and *ReVerb KB* does not contain enough data to train a robust CNN model. Still, using word embeddings significantly improves the network's performance (Houshmand, 2017).

We propose a corpus-based CNN model to do sentiment analysis on a large-scale dataset (IMDB) in order to predict sentiment orientation. Firstly, similar to (Houshmand, 2017) as an input to our network model we use the word2vec as a lexical resource proposed by Google to compute vector representations of words and reflect the distance between them. This step leads to initialize the parameters at a good point of our CNN model. Secondly, the proposed sentiment analysis approach is done using a convolutional neural network architecture with three convolution layers with padding, a flatten layer followed by two dense layers with two dropout layers in between. To the best of our knowledge, using this architecture in a CNN model with an embedding layer to analyze movie reviews sentiment classification has not been addressed before in literature. Our results with

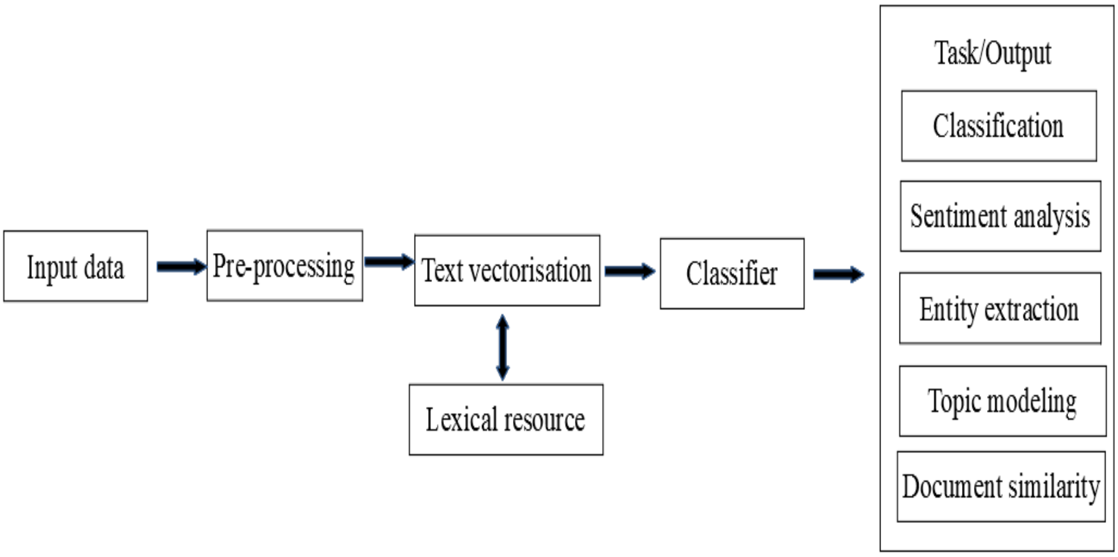


Figure 1 General architecture for text classification problems.

the proposed model have better results compared to related work.

4. PROPOSED APPROACH

With access to technology-based solutions and the rapid growth of social media platforms such as Twitter, Facebook, and online review sites such as IMDB, Amazon, and Yelp, users are sharing daily thoughts and opinions about different entities. These entities can be products, services, organizations, individuals, events, issues, or topics. This exponential growth of user-generated content draws growing attention from data scientists, as well as research and industry communities. The issue remains that reading every piece of this raw text data is time-consuming and resource demanding, therefore, analyzing this huge amount of text automatically gives companies an overview of how positive or negative users are to specific subjects will minimize losses. In order to automate this process work has been done in different fields like semantic parsing, sentence modeling and sentiment analysis (Mikolov et al., 2013; Yih et al., 2014; Ouayang et al., 2015; Houshmand, 2017; Kalchbrenner et al., 2014). Despite the results of previous work (Mikolov et al., 2013; Ouayang et al., 2015; Houshmand, 2017; Kalchbrenner et al., 2014), in addition to the strong empirical performance in Yih et al. (2014), their system has no room for improvement because the corpus does not contain enough data to train a robust CNN model. With the propose large-scale corpus-based model, we are able to obtain better results.

In this work, we use a CNN model to perform two tasks: binary-class sentiment analysis and multi-class text classification. In order to do so, first we analyze the sentiment of movie reviews using the publicly available IMDB dataset, then we classify news/ topics using the Reuters dataset. By using NLP, the computer can understand more than just the objective definitions of the words. This step includes using the word2vec model proposed by Google, which is a way of extracting features from the text for use in modeling, also using a classifier module to identify if a given piece of text is positive or negative in the case of sentiment analysis, and which topic or category the given piece of text fits into (Figure 1).

In this case, we are using a new CNN model as our classifier. Python libraries help the model learn with a faster curve, and the package “pandas” will help us read our CSV files containing both datasets. A Natural Language ToolKit (NLTK) is used to remove unnecessary data from the data sets. Figure 2 represents the process that takes place throughout the sentiment analysis process, which is divided into two sub-processes: the learning process where we train, test and validate our proposed CNN model and the classification process where new data is fed to the model. As illustrated in Figure 2, before any further analysis of the input text data, text pre-processing is needed, followed by text vectorization.

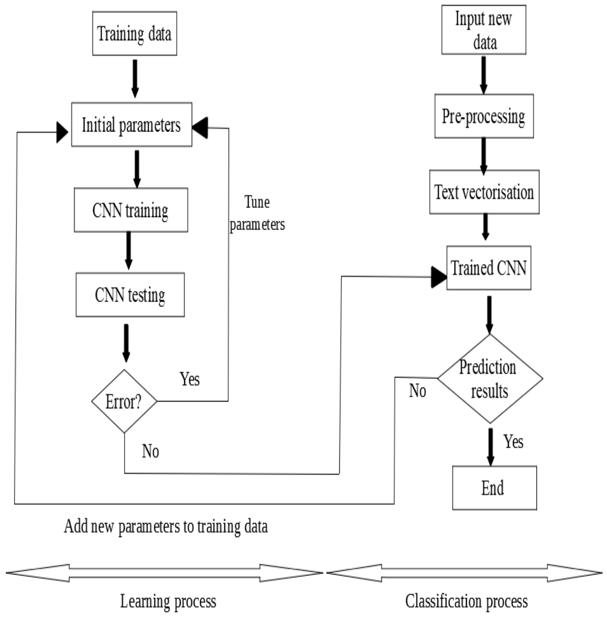


Figure 2 Global architecture for the proposed system.

4.1 Data pre-processing

It is necessary to normalize the text for any natural language processing tasks. Since it is often represented in a cryptic and informal way, systematic pre-processing of reviews is required to enhance the accuracy of our sentiment classifier. In this work, we perform a corpus-based analysis on text from users' movie reviews. Since natural language is frequently used in reviews, this type of text data contains a lot of noise as shown in Example 1, therefore, cleaning unnecessary information from raw comments (reviews) is needed. The movie review binary-class dataset used is IMDB, which contains 50,000 movie reviews labeled by sentiment (positive/negative). Similar to any NLP task, before any further processing, cleaning-up the data is crucial which involves the following steps:

1. Remove numeric and empty texts
2. Remove punctuation from texts
3. Convert words to lower case
4. Remove stop words

As demonstrated in Example 1, the datasets used contain non-relevant data (noise). Therefore, basic cleanup needs to be performed. Arbitrary characters and other useless information such as punctuation, stopwords, special characters and links/URLs were removed, since we found no significance in our classification approach. Then, text normalization was applied using regular expressions. When these NLP tasks are completed, the processed reviews are stored in a comma-separated value (CSV) file for further processing.

Stemming and lemmatization are text normalization (or sometimes called word normalization) techniques. This step is very important in order to get better accuracy for the proposed CNN model, and it consists of preparing the text, words, and documents for further processing. In order to stem and lemmatize words, sentences and documents, we used the public Python nltk package, the Natural Language Toolkit package, provided by Python for NLP tasks, as shown in Example 2.

<u>Example 1:</u>

## [1] "I was blessed to have seen this movie last night. It made me laugh, it made me cry and it made me love life. This movie is a great movie that depicts a love of a father for his son. Will Smith did an incredible job and deserves every accolade available to him. His son also did a fantastic job. There is a great lesson that is learned in this movie and it truly shares the struggles of everyday life. This movie was heart felt and touching. It was truly an experience worth having. Thank you for making this movie and I look forward to seeing it again."

## [1] "blessed night made laugh made cry made love life great depicts love father son incredible job deserves accolade son fantastic job great lesson learned shares struggles everyday life heart felt touching experience worth making forward"

<u>Example 2:</u>

"Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from data in various forms, both structured and unstructured,[1][2] similar to data mining."

4.2 Text vectorisation

In order to convert string features into numerical features, one can use one of the following methods.

One hot encoding maps each word to a unique ID, it has typical vocabulary sizes. They will vary between 10,000 and 250,000. This method is a natural representation to start with, though a poor one due to several drawbacks such as the size of input vector scales with size of vocabulary. There is the “out-of-vocabulary” problem (H. L. Trieu et al., 2016) where there is no relationship between words (each word is an independent unit vector). Also it is vulnerable to overfitting: sparse vectors which result in computations going to zero (T. Ojeda et al., 2018).

Bag of words is an approach where we set all words in the corpus (T. Ojeda et al., 2018). Its main advantage is that it is quick and simple. But it is too simple and orderless, without syntactic or semantic similarity.

N-gram model is a model with a set of all n-grams in the corpus. It tries to incorporate the order of words (T. Ojeda et al., 2018), unfortunately it still has a very large vocabulary set and no notion of syntactic/semantic similarity.

Term frequency-inverse document frequency is a model that captures the importance of a word (term) to a document in a corpus. The importance of a word increases proportionally according to the number of times a word appears in the document; but is contrarily equivalent to the frequency of the word in the corpus (T. Ojeda et al., 2018). The key advantage of this method is that it is easy to compute and has some basic metric to extract the most descriptive terms in a document. Thus it can easily compute the similarity between two documents using it, but it does not capture the position in the text, semantics and co-occurrences in different documents because it is based on the bag-of-words model.

Thus term frequency-inverse document frequency is only useful as a lexical resource, but it cannot capture semantics like topic models and word embedding. In our work we use word2vec published by Google in 2013, which is a neural network implementation that learns distributed representations for words (Mikolov et al., 2013). Prior to word2vec, other deep or recurrent neural network architectures had been proposed (Ouayang et al., 2015; Kalchbrenner et al., 2014) for learning word representations. The major problem with previous attempts was the long time required to train the models, while word2vec learns quickly compared to these models. In order to create meaningful representations, word2Vec does not need labels. Since most data in the real world is unlabeled, this feature is very useful. If the network is trained on a large dataset, it produces word vectors with interesting characteristics. As a result, words with similar meanings appear in clusters, and clusters are spaced such that some word relationships, such as analogies, can be reproduced using vector math.

4.3 Convolutional Neural Network classifier

We propose a word-based CNN architecture for both binary-class and multi-class text classification. First, there is a sentiment analysis on the IMDB movie reviews dataset, which contains 50,000 movie reviews labeled by sentiment (positive/negative), and second a text (topic) categorization for the Reuters corpus, which contains 10,788 news documents totaling 1.3 million words, where the documents have been classified into 90 topics and grouped into two sets. As shown in Figure 3, we train a CNN with an embedding layer and different convolution layers with padding. The purpose of using padding in every convolution layer is to conserve the size of the input data as it is; thus, no information is lost (Shen et al., 2014). These convolution layers are followed by a flatten layer and two dense layers with two dropout layers.

4.3.1 Sentence matrix

Instead of image pixels, the input to most NLP tasks is sentences or documents represented as a matrix. Each row of the matrix corresponds to one token, typically a word, but it could be a character (Krizhevsky et al., 2012). That is, each row is a vector that represents a word. Typically, these vectors are word embeddings

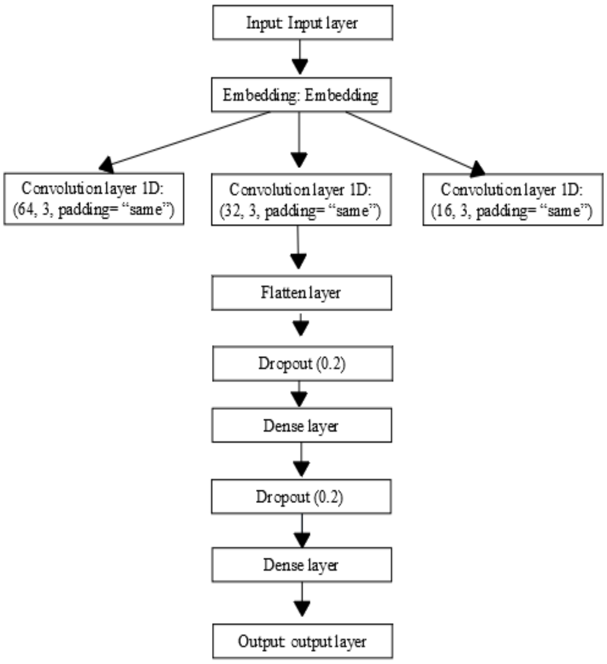

Input Input layer
Embedding: Embedding
Convolution layer 1D: (64, 3, padding= “same”)
Convolution layer 1D: (32, 3, padding= “same”)
Convolution layer 1D: (16, 3, padding= “same”)
Flatten layer
Dropout (0.2)
Dense layer
Dropout (0.2)
Dense layer
Output output layer

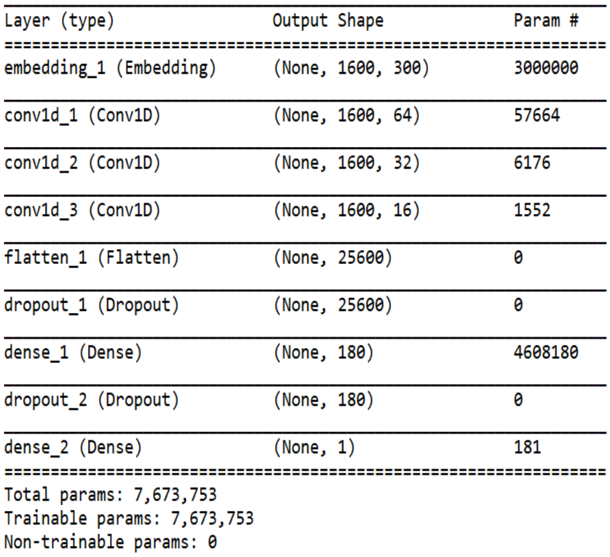
Figure 3 The layer architecture of the proposed CNN model.

like word2vec or Glove. For example in our work, a 10 word sentence using a 300-dimensional embedding, has a 10×300 matrix as input. That's our input sentence matrix (image) to the network (Y. Kim et al., 2014).

4.3.2 Embedding Layer

As input to our proposed model, the first layer is an embedding layer which is defined as the first hidden layer and its role is to transforms words into real-valued feature vectors known as embeddings. These vectors are able to capture morphological, syntactic and semantic information about the words. It must specify the following arguments: top-words, embedding-vector-length, and max-review-length. In this work, we truncate the reviews to a maximum length of 1600 words and we only consider the top 10,000 most frequently occurring words in the movie reviews dataset, and we used an embedding vector length of 300 dimensions. This is an important step in the proposed network architecture because it initializes the parameters of our CNN model.

The output of the embedding layer is a 2D vector (none, max-review-length, embedding-vector-length) with one embedding for each word in the input sequence of words. Some modification is applied to the basic convolutional operation (layer) where padding is used to conserve the original size of the input sentence matrix, therefore, there is no loss of information (Shen et al., 2014). To connect the dense layer (fully connected layer) to the 2D output matrix we must add a flatten layer in order to convert the output of the convolution layers into a single 1D vector to be used by the dense layer for final classification (Figure 4).



```
________________________________________________________________
Layer (type)                 Output Shape              Param #
================================================================
embedding_1 (Embedding)      (None, 1600, 300)         3000000
________________________________________________________________
conv1d_1 (Conv1D)            (None, 1600, 64)          57664
________________________________________________________________
conv1d_2 (Conv1D)            (None, 1600, 32)          6176
________________________________________________________________
conv1d_3 (Conv1D)            (None, 1600, 16)          1552
________________________________________________________________
flatten_1 (Flatten)          (None, 25600)             0
________________________________________________________________
dropout_1 (Dropout)          (None, 25600)             0
________________________________________________________________
dense_1 (Dense)              (None, 180)               4608180
________________________________________________________________
dropout_2 (Dropout)          (None, 180)               0
________________________________________________________________
dense_2 (Dense)              (None, 1)                 181
================================================================
Total params: 7,673,753
Trainable params: 7,673,753
Non-trainable params: 0
```

Figure 4 Total number of trainable parameters in our CNN model.

4.3.3 Fully activated Layer (Dense)

In deep learning models, activation functions are used at the fully activated layer (dense) and they can be divided into two types: linear activation functions and non-linear activation functions (ML, 2018). In our work, the first experiment is binary-class sentiment analysis using the IMDB dataset where we used the sigmoid activation function. We used a sigmoid function because it exists between 0 to 1. Therefore, it is adequate for our model since we have to predict the probability as an output. In the second experiment we train, test and validate our CNN model on a multi-class Reuters dataset. We used the soft-max activation function since it is a more generalized logistic activation function, which is used for multi-class classification.

4.3.4 Dropout Layer

With approximately 7 million trainable parameters, the proposed CNN model is very powerful. However, overfitting is a serious problem in large networks, making them slow to use and thus difficult to deal with overfitting by combining many different predictions. Dropout is a technique that prevents this problem and it refers to dropping out units (hidden and visible) in a neural network (Lai. S-H et al., 2017). By dropping a unit out, we mean temporarily removing it from the network, along with all its incoming and outgoing connections. In our model we use two dropout layers with (0.2), and the choice of which units to drop is random.

5. RESULTS AND DISCUSSION

We propose a CNN model to apply text classification. We define a CNN model and we train it on publicly available data sets: th IMDB movies reviews dataset and the Reuters dataset. Our model is word-based CNN with an embedding layer. At the embedding layer level, we tokenize text review sentences to a sentence matrix with rows where each row contains word vector representations of each token. In our work, we truncate the reviews to a maximum length of 1600 words and we only consider the top 10,000 most frequently occurring words in the movie reviews dataset. We experiment with the network model in two settings. The first experiment involves predicting sentiment classification of movie reviews and the second one is news/topic classification. The network performs well in both the binary and the multi-class experiments.

5.1 Datasets

As shown in Table 3, to evaluate the performance of our proposed model, we used two large scale datasets, the binary class IMDB dataset for sentiment classification (A. Maas et al., 2011) and the multi-class Reuters data set for news/topic classification (Table 2).

Table 2 IMDB and Reuters datasets.

IMDB	Reuters
#of sentences 50k #of positive reviews 25k #of negative reviews 25k	# of documents 10788 # of topics 90 # of word 1.3 million

We benchmark our CNN model on two different corpora from two different domains: movie reviews and news/topic classification. The movie review binary-class dataset used is IMDB, which contains 50,000 movie reviews labeled by sentiment (positive/negative). Reviews have been pre-processed, and each review is encoded as a sequence of word indexes (integers). This allows for quick filtering operations such as: "only consider the top 10,000 most common words, but eliminate the top 20 most common words" (A. Maas et al., 2011). In our experiments, we focus on sentiment prediction of complete sentences (reviews). The second corpus we use is the Reuters news wire topic classification. This dataset is a multi-class benchmark (e.g. there are multiple classes), multi-label (e.g. each document can belong to many classes) dataset (M. Thoma, 2018). Both datasets are used to validate our model, where the first dataset is the IMDB movies reviews. The data was split evenly with 25,000 reviews intended for training and 25,000 for testing. Moreover, each set has 12,500 positive and 12,500 negative reviews. We pre-processed the reviews, and each review is encoded as a sequence of word indexes (integers). And the second dataset is the Reuters dataset for document classification; it has 10,788 news documents and 90 classes/topics.

We conduct an empirical exploration on the use of the proposed word-based CNN architecture for sentiment classification on IMDB movie reviews and the Reuters corpus for text categorization, which contains 10,788 news documents totaling 1.3 million words where the documents have been classified into 90 topics and grouped into two sets. In the present work, we train a CNN with an embedding layer, convolution layers, a flatten layer and two dense layers with two dropouts. Although CNNs extract high-level features in image analysis, our model actually performs well in 2D problems and trains 50% to 60% faster as shown in Figures 5 and 6. The proposed model has ~7M trainable parameters and is trained in a Python environment which takes around 15 to 20 minutes on an Intel (R) Core (TM) i5-5200U CPU with 2.20GHz of RAM.

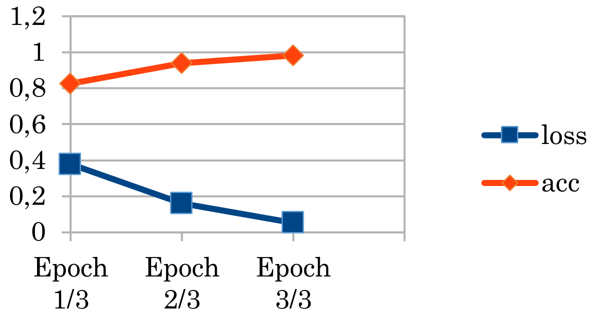


Figure 5 Loss function and accuracy values of the proposed model on the Reuters dataset.

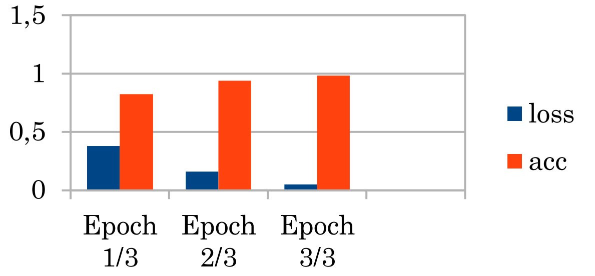


Figure 6 Loss function and accuracy values of the proposed model on the Reuters dataset.

In the sentiment classification of movie reviews using the IMDB dataset, in order to horizontally extract features, we used binary cross entropy loss because it is a binary classification problem. To avoid overfitting the training data dropout (0.2) was necessary. For reinforcing the generalization power, we disabled the network with holes during training. This way the network is forced to build new paths and extract new patterns. Despite the satisfactory performance of our model, and in addition we were able to validate the proposed model on both IMDB and Reuters datasets. After 15 to 20 minutes of training, we obtain ~86% accuracy (Table 3).

Table 3 Accuracy of the models on the IMDB dataset for binary-class and Reuters dataset for multi-class.

	Fine-grained	**Binary**
CNN model (Yih et al. 2014)		54%
DCNN model (Kalchbrenner et al. 2014)	48.5%	86.8%
CNN+word2vec model (Ouayang et al. 2015)	45.4%	
CNN model (Houshmand, 2017)		40.5%
CNN+word2vec model (Houshmand, 2017)		46.4%
CNN model	**85.95%**	**85.80%**
CNN+ LSTM model		**95%**

We tried to improve the accuracy of the model by conducting other experiments using a modified CNN and Long Short-Term Memory (LSTM) architecture. The embedding layer is still the first hidden layer of our CNN-LSTM model, we added the LSTM layer followed by GlobalMaxpool 1D layer, and 2 Dense layers with Dropout. The main difference between the CNN model and the CNN-LSTM model is at this level where we have the first dense layer with the 'ReLu' activation function instead of 'sigmoid' in the first CNN model. Similar to the experiments with our CNN model, in order to avoid overfitting, a dropout layer (0.5) was necessary. This layer is followed by the second dense layer where a 'sigmoid' activation function is used. The same NLP tasks are applied to the reviews which involve the following steps:

1. Remove numeric and empty texts
2. Remove punctuation from texts
3. Convert words to lower case
4. Remove stop words
5. Stemming

Only the IMDB dataset was used to train, test and validate the proposed CNN-LSTM model. The labeled dataset consists of 50,000 IMDB movie reviews, selected for sentiment analysis. The sentiment of reviews is binary, meaning the IMDB rating below 5 results in a sentiment score of 0, and ratings equal to or greater than 7 have a sentiment score of 1 and no individual movie has more than 30 reviews.

5.1.1 Raw Reviews

- *'With all this stuff going down at the moment...'*
- *'The Classic War of the Worlds by Timothy Hi...'*
- *'The film starts with a manager (Nicholas Bell)...'*
- *'it must be assumed that those who praised this...'*
- *'Superbly trashy and wondrously unpretentious 8...'*

5.1.2 Processed reviews

- *'stuff go moment mj ive start listen music watch...'*
- *'classic war world timothy hines entertain film...'*
- *'film start manager nicholas bell give welcome...'*
- *'must assume praise film great film opera ev...'*
- *'superbly trashy wondrously unpretentious 80 ex...'*

The 25,000 review labeled as the training set do not include any of the same movies as the 25,000 review test set. In addition, there are another 50,000 IMDB reviews provided without any rating labels.

The labeled training set is tab-delimited and has a header row followed by 25,000 rows containing an id, sentiment, and text for each review. The test set is a tab-delimited file that has a header row followed by 25,000 rows containing an ID and text for each review. The task of our CNN-LSTM model is to predict the sentiment for each. An extra training set with no labels is provided that is a tab-delimited file with a header row followed by 50,000 rows containing an ID and text for each review.

One interesting thing about the results of the CNN-LTSM model is that the accuracy improved significantly compared to the first CNN model. The CNN-LSTM model reached an F1 score of 0.95 on the test data while the

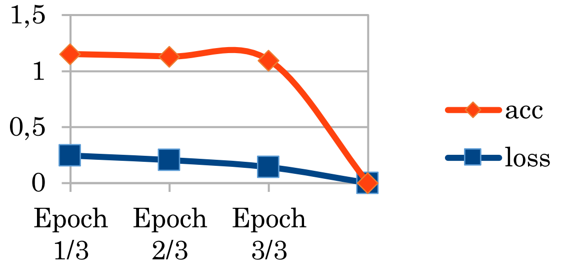


Figure 7 Loss function and accuracy values of the proposed CNN-LSTM model.

CNN without the LSTM layer got ~ 86% (Figure 7). We conclude that both models perform well and show satisfactory results against state-of-the-art methods, which is quite respectable given: (1) the large size of the data sets and (2) the number of parameters in the network.

6. CONCLUSION

With an aim of classifying the sentiment of movie reviews into two classes (positive or negative) and applying text classification on news text in order to perform topic classification, our method has been implemented with an acceptable performance. As a next step of making use of a data driven model, CNN has been taken into consideration. In this work we present a new CNN architecture that jointly uses word2vec as an input layer to the CNN model and an LSTM layer. The proposed model has yielded better results compared to previous methods with an accuracy of ~86 % for the first experiment and 95% for the CNN-LSTM (Mikolov et al., 2013; Ouayang et al., 2015; Houshmand, 2017; Kalchbrenner et al., 2014). The main contributions of the paper are: (1) the short training time despite the large size of the data sets and the number of parameters in the network; (2) the demonstration that adding an LSTM layer to the network can be effective and significantly improving the model's accuracy. In future research it will be interesting to apply the proposed model architecture to other NLP applications such as spam filtering and web searches, as well as exploring Bayesian optimization frameworks and also, conducting other experiments using recursive neural network with the long short-term memory architectures for sentiment categorization of text review.

7. REFERENCES

Bengio, Y. R. Ducharme, P. Vincent, and C. Jauvin, (2003). A Neural Probabilistic Language Model. Journal of Machine Learning Research, (3), 1137-1155.

Bing Liu, (2011). Opinion Mining and Sentiment Analysis, WEB DATA MINING. Data Centric Systems and Applications, Part 2, 459-526.

Bing Liu, (2012). Sentiment analysis and opinion mining. San Rafael, CA: Morgan and Claypool Publishers.

Britz, D. (2015). Understanding Convolutional neural networks for NLP, in WildML. Retrieved October 17th, 2018, from http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuglu, and P. Kuksa. (2011). Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research, (12), 2493–2537

Deng, L. and D. Yu, (2014). Deep learning: Methods and applications. Grand Rapids, MI, United States: Now publishers.

Fei-Fei, L., R. Fergus, and P. Perona. (2007). Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 objects categories. Journal of Computer Vision and Image Understanding, 106(1), 59-70.

Gibson, A. and J. Patterson, (2017). Deep Learning. Chapter 1: A review on machine learning. O'Reilly Media, Inc.

Graves, A. (2013). Generating sequences with Recurrent Neural Networks. Retrieved August 13th, 2018, from https://arxiv.org/abs/1308.0850

Heaton, J. (2015). Artificial intelligence for humans, volume 3: Deep learning and neural networks. United States: Createspace Independent Publishing Platform.

Houshmand, Shirani-Mehr, (2017). Applications of Deep Learning to Sentiment Analysis of Movie Reviews. Retrieved December 6th, 2018, from https://cs224d.stanford.edu/reports/Shirani-MehrH.pdf

Kalchbrenner, N., E. Grefenstette, and P. Blunsom. (2014). A Convolutional Neural Network for Modelling Sentences. In Proceedings of ACL 2014.

Kharde, A. and S. Sonawane, (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. International Journal of Computer Applications, Volume 139, No.11, 0975-8887

Kim, Y. (2014). Convolutional neural networks for sentence classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), (pp. 1746–1751)

Krizhevsky, A., I. Sutskever, and G. Hinton, (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, 1097-1105

Lai, S-H., V. Lepetit, K. Nishino, and Y. Sato, (2017). Computer Vision – ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part II, volume 10112, doi 10.1007/978-3-319-54184-6, 183-204

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. Journal of Neural Computation, 1(4), 541-551

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner. (1998). Gradient-based learning applied to document recognition. In proceeding of the IEEE, 86(11), (pp. 2278-2324).

Machine Learning Cheatsheet, (2018). Activation Functions. Retrieved December 6th, 2018, from https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html

Maas, A. et al., (2011). Learning word vectors for sentiment analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, volume 1, (pp. 142- 150)

Micolov, T., K. Chen, G. Corrado, and J. Dean, (2013). Efficient Estimation of Word Representations in Vector Space. Journal of Computing Research Repository.

Mohri, M., A. Rostamizadeh, and A. Talwalkar, (2012). Foundations of machine learning. Cambridge, MA: MIT Press.

Ojeda, T., R. Bilbro and B. Bengfort, (2018). Applied Text Analysis with Python. Chapter 4. Text Vectorization and Transformation Pipelines. O'Reilly Media, Inc.

Ouyang, X., P. Zhou, C. H. Li, and L. Liu. (2015). Sentiment analysis using Convolutional neural network. In IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing.

Russell, M. (2011). Mining the social web, O'Reilly Media.

Santos, D., and C. Gatti, (2014). Deep convolutional neural networks for sentiment analysis of short texts. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, (pp. 69–78)

Semanet, P., S. Chintala, and Y. LeCun. (2012). Convolutional neural networks applied to house numbers digit classification. In Proceeding of the 21st International Conference on Pattern Recognition (ICPR), (pp. 3288-3291).

Semanet, P., and Y. LeCun. (2011). Traffic sign recognition with multi-scale convolutional networks. In Proceeding of International Joint Conference on Neural Networks (IJCNN), (pp. 2809-2813).

Severyn, A., and A. Moschitti, (2015). Twitter sentiment analysis with deep convolutional neural networks. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, (pp. 959–962)

Shanmugamani, R., and R. Arumugam, (2018). Hands-On Natural Language Processing with Python. Packt Publishing.

Shen, Y., X. He, J. Gao, L. Deng, and G. Mesnil. (2014). Learning Semantic Representations Using Convolutional Neural Networks for Web Search. In Proceedings of WWW 2014.

Srinivas, S., R. Sarvadevabhatla, K. Mopuri, N. Prabhu, (2016). A taxonomy of deep convolutional neural nets for computer vision. Frontiers in Robotics and AI 2, 36

Tang, D., and M. Zhang, (2018). Deep Learning in Sentiment Analysis. In: Deng L., Liu Y. (eds) Deep Learning in Natural Language Processing. Springer, Singapore, 219-253

Thoma, M. (2017). The reuters dataset, Retrieved October 23rd, 2018, from https://martin-thoma.com/nlp-reuters/

Trieu, H.L., L. M. Nguyen and P. T. Nguyen, (2016). Dealing with Out-Of-Vocabulary Problem in Sentence Alignment Using Word Similarity. Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation (PACLIC 30). 259-266

Yadav, V. (2017). How neural networks learn nonlinear functions and classify linearly non-separable data?, Medium, Retrieved October 19th, 2018, from https://medium.com/@vivek.yadav/how-neural-networks-learn-nonlinear-functions-and-classify-linearly-non-separable-data-22328e7e5be1

Yih, W., K. Toutanova, J. Platt, and C. Meek. (2011). Learning Discriminative Projections for Text Similarity Measures. In Proceeding of the Fifteenth Conference on Computational Natural Language Learning CoNLL'11. (pp. 247-256).

Yih, W., X. He, and C. Meek. (2014). Semantic Parsing for Single-Relation Question answering. In ACL Proceeding.

Zhang, Y. and C. Wallace, (2016). A Sensitivity Analysis of Convolutional Neural Networks for Sentence Classification. Cornell University Library, Computer Science, Computation and Language