International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT217695

A Comprehensive study on Text Classification: Application of Convolutional Neural Networks and Deep Learning methods

Tiyasa Chatterjee, Asoke Nath
Department of Computer Science, St. Xavier's College (Automous), Kolkata, India

Article Info
Volume 7, Issue 6
Page Number : 364-371

Publication Issue :
November-December-2021

Article History
Accepted : 12 Dec 2021
Published : 26 Dec 2021

ABSTRACT

Text classification is an essential part in many applications, such as web searching, information filtering, language identification and sentiment analysis such as predicting the sentiment of tweets and movie reviews, as well as classifying email as spam or not. Classifying our content and products into categories help users to easily search and navigate within website or application, Deep learning methods are proving very good at text classification. Deep learning is a set of algorithms and techniques to imitate how the human brain works, called neural networks. Different Neural networks such as Convolutional Neural Networks (CNN when used with Deep learning algorithms, like Word2Vec or GloVe , obtain better vector representations for words and also improve the accuracy of classifiers trained with traditional machine learning algorithms.
The authors have made a comprehensive study on Text Classification using convolutional neural network (CNN) .The authors will discuss different models and methods and the experimental results based on variety of datasets.
Keywords: Neural networks, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Deep learning, bag-of-words (BoW), word embedding, NLP, Word2Vec, semantics, TREC, Naive Bayes, Support Vector Machines SVM, vectors, filters, features, tokens, pooling, rule-based, hybrid systems, kernels, matrix, regularization.

I. INTRODUCTION

Neural networks form the base of deep learning where the algorithms are inspired by the structure of the human brain. Neural networks rely on training data to learn and improve their accuracy over time. Text classification is a method of extracting generic tags from unstructured text. These generic tags come from a set of pre-defined categories. Text classification is one of the fundamental tasks in natural language processing and is an emerging field of study such as such as Marketing, Product Management, Academia, entertainment industries and Governance that are already leveraging the process of analyzing and extracting information from textual data.[1] Therefore, it has attracted immense attention from many researchers across the globe.

A key problem in text classification is feature representation and extraction, which

is mainly based on the bag-of-words (BoW) model.[2] Recently, the rapid development of pre-trained word embedding and deep neural networks has brought new inspiration to various NLP tasks.[3] Word embedding
is the neural representation of words for text analysis in a form of a real valued vector.
With the pretrained word embeddings, neural networks describe their great performance in many NLP tasks. Recurrent Neural Network (RecurrentNN) and Convolutional Neural Network(CNN) are used widely for text or sentence classification. RNN analyzes a
sentence word by word and stores the semantics of all the previous text in a fixedsized hidden layer. This could be suitable to capture semantics of long texts. But, RNN is a biased model, where later words are more dominant than earlier words.[4] Therefore, it reduce the effectiveness when it is used to capture the semantics of a whole document, because key

components could appear anywhere in a document rather than at the end. To tackle the bias problem, the Convolutional Neural Network (CNN), an unbiased model is introduced to NLP proceedings, which can fairly determine discriminative phrases in a text with a max-pooling layer. Thus, the convolutional neural network may better capture the semantic of texts as compared to recursive or other networks.

II. LITERATURE SURVEY

In Convolutional Neural Networks for Sentence Classification by Yoon Kim, a series of experiments with Convolutional Neural Networks (CNN) built on top of word2vec was presented.
The suggested model was tested against several benchmarks. In Movie Reviews (MR) and Customer Reviews (CR), the task was to detect positive/negative sentiment. In Stanford Sentiment Treebank (SST-1), there were already more classes to predict: very positive, positive, neutral, negative, very negative. In Subjectivity data set (Subj), sentences were classified into two types, subjective or objective. In TREC the goal was to classify a question into six question types, the results show that after little tuning of hyperparameters the model performs excellent with the use of pre-trained vectors.[5]

The article Text Understanding from Scratch by Xiang Zhang and Yann LeCun shows that it's possible to apply deep learning to text understanding from character-level inputs all the way up to abstract text concepts with help of temporal Convolutional Networks (ConvNets) (CNN). Here, the authors assert that ConvNets can achieve excellent performance without the knowledge of words, phrases, sentences and any other syntactic or semantic structures with regards to a human language.[6] The model was tested on the DBpedia ontology classification data set with 14 classes. The results indicate both good training (99.96%) and testing (98.40 %) accuracy, with some improvement from thesaurus augmentation. In addition, the sentiment analysis test was performed on the Amazon Review data set. In this study, the researchers constructed a sentiment polarity data set with two negative and two positive labels. The result is 97.57% training accuracy and 95.07% testing accuracy. The

model was also tested on Yahoo! Answers Comprehensive Questions and Answers data set with 10 classes (Society & Culture, Science & Mathematics, Health, Education & Reference, Computers & Internet, Sports,
Business & Finance, Entertainment & Music, Family & Relationships, Politics & Government) and on AG's corpus where the task was a news categorization into four categories (World, Sports, Business, Sci/Tech.). Obtained results confirm that to achieve good text

understanding ConvNets require a large corpus in order to learn from scratch.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao introduced recurrent convolutional neural networks for text classification without human-designed features in their document Recurrent Convolutional Neural Networks for Text Classification [7]. The team tested their model using four data sets: 20Newsgroup (with four categories such as computers, politics, recreation, and religion), Fudan Set (a Chinese document classification set that consists of 20 classes, including art, education, and energy), ACL Anthology Network (with five languages: English, Japanese, German, Chinese, and
French), and Sentiment Treebank (with Very Negative, Negative, Neutral, Positive, and Very Positive labels). After testing, the model was compared to existing text classification methods like Bag of Words, Bigrams + LR, SVM, LDA, Tree Kernels, RecursiveNN, and CNN.

III. METHODS AND MODELS

There are many approaches to automatic text classification:

- rule based systems
- machine language based systems that use different machine learning text classification algorithms such as: Naive Bayes, Support Vector Machines(SVM), deep learning methods and hybrid systems.[8]
- Deep learning methods are proving very good at text classification, achieving state-of-the-art results on a suite of standard academic benchmark problems.

Let's look at some proposed methods and models one by one for text classification using neural network.
Word embeddings with CNN : word embedding is a type of word representation for text analysis, in the form of a real-valued vector that that allows words with similar meaning to have a similar representation. Yoav Goldberg, in his primer on deep learning for natural language processing, states that neural networks offer better performance than classical linear classifiers, especially when used with pre-trained word embeddings.[9] He also comments that networks with convolutional and pooling layers are effective at text classification as they easily take out salient features (tokens) in a way that is independent to their position within the input sequences.

Single Layer CNN Model: With a single layer CNN architecture, one can get good results for document classification using differently sized kernels with the filters to allow grouping of word representations at different scales.

Yoon Kim in his study of the use of pretrained word vectors for Text classification tasks with Convolutional Neural Networks discovered that using pre-trained static word vectors for the task works very well.

In this paper, let's discuss the model architecture proposed by Yoon Kim. Yoon Kim in his study of the classification tasks with Convolutional Neural Networks found that pre- trained word embeddings that were trained on very large text corpora, (such as the word2vec vectors) may offer good universal features for use in natural language processing. Kim in his model shows that the sentences are mapped to embedding vectors and are available as a matrix input to the model. Using differently sized kernels (2 or 3 words at a time) convolutions are performed across the input word by word. The output mappings are then processed using a max pooling layer to filter out the extracted features. Bellow is a a diagram provided by Kim that describes the sampling of the filters with differently sized kernels as different colours (red and yellow).

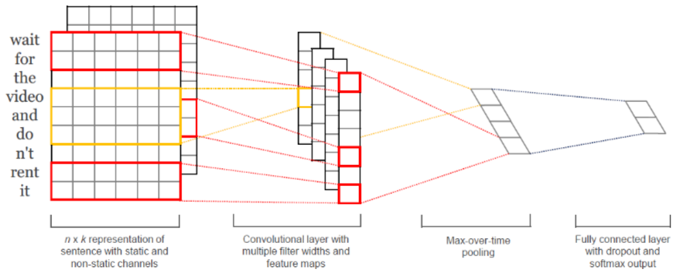

wait
for
the
video
and
do
n't
rent
it
n x *k* representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Fig 2 : “Convolutional Neural Networks for Sentence classification".[8]

Let xi ∈ R^k be the k-dimensional word vector corresponding to the ith word in the sentence. A sentence of length n (padded where necessary) is represented as x1:n = x1 ⊕ x2 ⊕ . . . ⊕ xn --(1)

where ⊕ is the concatenation operator. In general, let xi:i+j refer to the concatenation of words xi , xi+1, . . . , xi+j . A convolution operation involves a filter w ∈ R^{hk}, which is applied to a window of h words to produce a new feature. For example, a feature ci is generated from a window of words xi:i+h−1 by ci = f(w · xi:i+h−1 + b) (2)

Here b ∈ R is a bias term and f is a non- linear function such as the hyperbolic tangent. This filter is applied to each possible window of words in the sentence

{x1:h, x2:h+1, ... , xn−h+1:n} to produce a feature map c = [c1, c2, . . . , cn−h+1], (3)

with c ∈ R^{n-h+1}. The authors then apply a max pooling operation over the feature map and take the maximum value cˆ = max{c} as the feature corresponding to this particular filter. The idea is to capture the most important feature—one with the highest value—for each feature map.

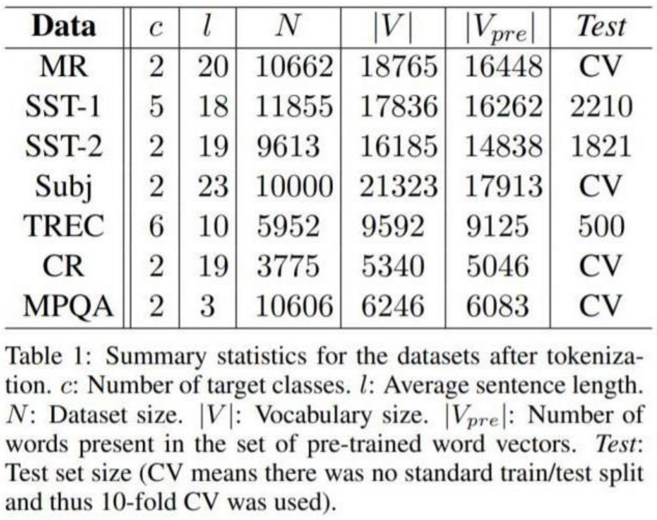
The authors have described the process by which one feature is extracted from one filter. The model uses multiple filters (with varying window sizes) to obtain multiple features. These features form the penultimate layer and are passed to a fully connected softmax layer whose output is the probability distribution over labels. In one of the model variants, the authors experiment with having two ‘channels’ of word vectors—one that is kept static throughout training and one that is fine-tuned via backpropagation. In the multichannel architecture, illustrated in figure 1, each filter is applied to both channels and the results are added to calculate ci in equation (2).

The model is otherwise equivalent to the single channel architecture.

Regularization : For regularization the authors employ dropout on the penultimate layer with a constraint on l2-norms of the weight vectors.

Dropout prevents coadaptation of hidden units by randomly dropping out—i.e., setting to zero—a proportion p of the hidden units during forward-backpropagation. That is, given the penultimate layer z = [ˆc1, . . . , cˆm] (using m filters), instead of using y = w · z + b--(4) for output unit y in forward propagation, dropout

uses y = w · (z ◦ r) + b, (5) where ◦ is the element-wise multiplication operator. Gradients are backpropagated only through the unmasked units. At test time, the learned weight vectors are scaled by p such that wˆ = pw, and wˆ is used (without dropout) to score unseen sentences. We additionally constrain l2norms of the weight vectors by rescaling w to have ||w||2 = s whenever ||w||2 > s after a gradient descent step.



Data	c	l	N	$\|V\|$	$\|V_{pre}\|$	*Test*
MR	2	20	10662	18765	16448	CV
SST-1	5	18	11855	17836	16262	2210
SST-2	2	19	9613	16185	14838	1821
Subj	2	23	10000	21323	17913	CV
TREC	6	10	5952	9592	9125	500
CR	2	19	3775	5340	5046	CV
MPQA	2	3	10606	6246	6083	CV

Table 1: Summary statistics for the datasets after tokenization. c: Number of target classes. l: Average sentence length. N: Dataset size. $|V|$: Vocabulary size. $|V_{pre}|$: Number of words present in the set of pre-trained word vectors. *Test*: Test set size (CV means there was no standard train/test split and thus 10-fold CV was used).

Table2: Summary statistics for the datasets.[5]

Usefully, Kim reports his chosen model configuration, discovered via grid search, and used across a suite of 7 text classification tasks, summarized as follows:

- Transfer function: rectified linear.
- Kernel sizes: 3, 4, 5.
- Number of filters: 100
- Dropout rate: 0.5
- Weight regularization (L2): 3
- Batch Size: 50
- Update Rule: Adadelta
- Adadelta update rule.[16]

Datasets: Movie reviews: MR with one sentence per review where the classification involves detecting positive/negative reviews.[10]

SST-1: Stanford Sentiment Treebank—an extension of MR but with train/dev/test splits provided and fine-grained labels (very positive, positive, neutral, negative, very negative), re-labelled by Socher et al. (2013).[11]

SST-2: Same as SST-1 but with binary labels and neutral reviews removed.

Subj: Subjectivity dataset where the task is to classify a sentence whether it is subjective or objective.[12]

TREC: TREC question dataset—task involves classifying a question into 6 question types (whether the question is about person, location, numeric information, etc.). [13]

CR: Customer reviews of various products (cameras, MP3s etc.). Task is to predict positive/negative reviews.[14]

MPQA: Opinion polarity detection subtask of the MPQA dataset. [15]

Hyperparameters and Training: For all datasets the author used: rectified linear units, filter windows (h) of 3, 4, 5 with 100 feature maps each, dropout rate (p) of 0.5, l2 constraint (s) of 3, and mini-batch size of 50. These values were chosen via a grid search on the SST-2 dev set. The authors did not perform any dataset- specific tuning other than early stopping on dev sets. For datasets without a standard dev set, it was randomly selected 10% of the training data as the dev set. Training is done through stochastic gradient descent over shuffled mini-batches with the

Pre-trained Word Vectors: Initializing the word vectors and those found in the moderated neural model is a popular way to improve performance when there is no large set of supervised training set. The authors used the publicly available word2vec vectors trained on 100 billion words from Google News. The vectors were trained using the continuous bag-of-words architecture and have dimensionality of 300(Mikolov et al., 2013). Words that were not present in the set of pre-trained words are initialized randomly.[17]

IV. RESULTS AND DISCUSSION

The above mentioned datasets are experimented with different variant of Yoon Kim's model as follows:

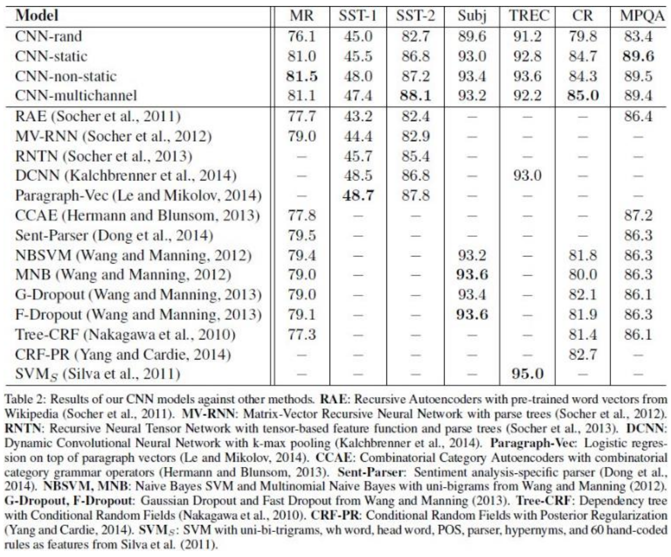
CNN-rand: the base model where all words are randomly initialized and then modified during training.

CNN-static: A model with pre-trained vectors from word2vec. All words— including the unknown ones that are randomly initialized— are kept static and only the other parameters of the model are learned.

CNN-non-static: Same as above but the pretrained vectors are fine-tuned for each task.

CNN-multichannel: A model with two sets of word vectors. Each set of vectors is treated as a 'channel' and each filter is applied to both channels, but gradients are backpropagated only through one of the channels. Hence the model is able to fine- tune one set of vectors while keeping the other static. Both channels are initialized with word2vec.

Results of the model proposed by Yoon Kim against other methods are listed in the table bellow:



Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	**89.6**
CNN-non-static	**81.5**	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	–	–	–	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	–	–	–	–
RNTN (Socher et al., 2013)	–	45.7	85.4	–	–	–	–
DCNN (Kalchbrenner et al., 2014)	–	48.5	86.8	–	93.0	–	–
Paragraph-Vec (Le and Mikolov, 2014)	–	**48.7**	87.8	–	–	–	–
CCAE (Hermann and Blunsom, 2013)	77.8	–	–	–	–	–	87.2
Sent-Parser (Dong et al., 2014)	79.5	–	–	–	–	–	86.3
NBSVM (Wang and Manning, 2012)	79.4	–	–	93.2	–	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	–	–	**93.6**	–	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	–	–	93.4	–	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	–	–	**93.6**	–	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	–	–	–	–	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	–	–	–	–	–	82.7	–
SVM$_S$ (Silva et al., 2011)	–	–	–	–	**95.0**	–	–

Table 2: Results of our CNN models against other methods. **RAE**: Recursive Autoencoders with pre-trained word vectors from Wikipedia (Socher et al., 2011). **MV-RNN**: Matrix-Vector Recursive Neural Network with parse trees (Socher et al., 2012). **RNTN**: Recursive Neural Tensor Network with tensor-based feature function and parse trees (Socher et al., 2013). **DCNN**: Dynamic Convolutional Neural Network with k-max pooling (Kalchbrenner et al., 2014). **Paragraph-Vec**: Logistic regression on top of paragraph vectors (Le and Mikolov, 2014). **CCAE**: Combinatorial Category Autoencoders with combinatorial category grammar operators (Hermann and Blunsom, 2013). **Sent-Parser**: Sentiment analysis-specific parser (Dong et al., 2014). **NBSVM, MNB**: Naive Bayes SVM and Multinomial Naive Bayes with uni-bigrams from Wang and Manning (2012). **G-Dropout, F-Dropout**: Gaussian Dropout and Fast Dropout from Wang and Manning (2013). **Tree-CRF**: Dependency tree with Conditional Random Fields (Nakagawa et al., 2010). **CRF-PR**: Conditional Random Fields with Posterior Regularization (Yang and Cardie, 2014). **SVM$_S$**: SVM with uni-bi-trigrams, wh word, head word, POS, parser, hypernyms, and 60 hand-coded rules as features from Silva et al. (2011).

Table 2 : Results of the model.[5]

Results of the base model against other methods are listed in the above table and thus we can conclude the followings:

1. The baseline model with CNN-rand does not perform well on its own. But, with the use of pre-trained vectors, the gains were high.
2. The model with static vectors (CNNstatic) also performs well, giving almost same results as deep learning models.
3. These results suggest that the pretrained vectors are good, 'universal' feature extractors and can be utilized across datasets.

Random vs static representations: From the results, we can say the static word embedding using pre-trained Word2Vec always performs better. On the other hand, the dynamic vector representation model will fine-tune the parameters initialized by Word2Vec vectors to learn the meaningful representation for each task which sometimes may result in better performance than the static one but not true for all cases and can have lower accuracy. For example, good is most similar to bad in word2vec as they are almost syntactically equivalent. But for vectors in the non-static channel that were finetuned on the SST-2 dataset, this is no longer the case. Similarly, good is closer to nice and is great for expressing sentiment which is reflected in the learned vectors.

Multichannel vs. Single Channel Models: Though the multichannel models can prevent overfitting (ensuring that the learned vectors do not deviate too far from the original values) the results, however are mixed. Instead of using an additional channel for the non-static portion, one could maintain a single channel but employ extra dimensions that are allowed to be modified during training.

BoW vs. Word Embedding: For text related problems in the NLP field Word embeddings is indeed a huge success and outperform Bag of Words (BoW) as BoW has the limitations as large feature dimension, sparse representation etc.

However, when building a baseline model or when our data set is small and context is domain specific which means that we cannot find corresponding vector from pretrained word embedding models. (GloVe, fastText etc), in such cases BoW may work better than word embedding.[18]

CNNs role as a feature extractor: Yoav Goldberg in his book "Neural Network Methods for Natural Language Processing" highlights that CNN is actually a feature generating architecture. It does not include an independent, useful network in itself, but rather aims to be integrated into a larger network, and trained to work in conjunction with it to produce the final results. CNN's responsibility is to extract meaningful substructures that are useful for all prediction tasks.

V. LIMITATIONS

Which deep neural network performs better when dealing with text data highly depends on how often the comprehension of global or long-range semantics is required. For tasks where length of text is the prior, it's a good practice to go with RNN variants. Such tasks include: translations, question answering etc. The convolutions and pooling operations of a CNN model lose information about the base order of words.

However, one can add positional features to the input to reduce the problem. Pooling also reduces the output dimensionality. By using the max operation, one keep the information about whether or not the feature appeared in the sentence, but we lose the information about where exactly it appeared.

VI. CONCLUSION and FUTURE SCOPE

Any model that is built on the base of word embedding causes the model to perform extremely well. If hidden layers and the kernel sizes are increased to any neural network it is expected do better in the task.

Researchers can also explore other pre-trained word embedding options such as GloVe and FastText with static and dynamic modes and then we can compare the results with Word2Vec.[18]

There are several directions in which text classification can play crucial role like Ontology learning, Knowledge extraction, Storytelling, Text summarization etc.

VII. REFERENCES

[1]. Jason Brownlee."Text classification using neural network". 2017. (online) Link:https://machinelearningmastery.com/ best-practices-document-classificationdeep-learning/

[2]. Edward Ma. "Bag of words and word embedding". 2018. (online) Link: https://towardsdatascience.com/3basic-approaches-in-bag-of-words-whichare-better-than-word-embeddingsc2cbc7398016

[3]. Aravindpai Pai. "Different types of neural networks." 2020. (online) Link: https://www.analyticsvidhya.com/blog/ 20 20/02/cnn-vs-rnn-vs-mlp-analyzing- 3types-of-neural-networks-in- deeplearning/

[4]. Jason Brownlee. Recurrent neural network for deep learning. 2016. (online) Link:https://machinelearningmastery.com/ crash-course-recurrent-neural-networksdeep-learning/

[5]. Yoon Kim. Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014.

[6]. Xiang Zhang and Yann LeCun. "Text Understanding from Scratch". 2015.

[7]. Siwei Lai, Liheng Xu, Kang Liu, Jun Zhao. Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. "Recurrent Convolutional Neural Networks for Text Classification" . 2015

[8]. Sunil Ray. "Naive Bayes algorithm". 2017. (online) Link:

https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[9]. Yoav Goldberg.“A Primer on Neural Network Models for Natural Language Processing”. 2015.

[10]. Bo Pang and Lillian Lee. “Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales”, Proceedings of ACL 2005.

[11]. Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng and Christopher Potts. “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank”. 2013.

[12]. Bo Pang, Lillian Lee. “A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts”.2004.

[13]. Xin Li and Dan Roth. “Learning Question Classifiers”. Research supported by NSF grants IIS-9801638 and ITR IIS0085836 and an ONR MURI Award. 2002.

[14]. Minqing Hu and Bing Liu. “Mining and Summarizing Customer Reviews”. 2004.

[15]. J Wiebe, T wilson and C Cardi.“Annotating Expressions of opinions and emotions in language”. 2005.

[16]. Zeiler. “ADADELTA: “An adaptive learning rate method”.2012.

[17]. Josh Barua. “Word Embeddings Versus Bag of Words: The Curious Case of a Recommender system.” 2020. Link: https://medium.com/swlh/wordembeddings-versus-bag-of-words-thecurious-case-of-recommender-systems-6ac1604d4424

[18]. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. “Efficient Estimation of Word Representations in Vector Space”.2013.

[19]. Shiva verma. “1d convolutional neural network.”. 2019. (online) Link:https://towardsdatascience.com/understanding-1d-and-3d-convolution-neuralnetwork-keras-9d8f76e29610

Cite this article as :

Tiyasa Chatterjee, Asoke Nath, "A Comprehensive study on Text Classification: Application of Convolutional Neural Networks and Deep Learning methods", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 7 Issue 6, pp. 364-371, November-December 2021.
Available at
doi : https://doi.org/10.32628/CSEIT217695
Journal URL : https://ijsrcseit.com/CSEIT217695

AUTHOR PROFILE:

Dr. Asoke Nath is working as Associate Professor in the Department of Computer Science, St. Xavier’s College (Autonomous), Kolkata. He is engaged in research work in the field of Cryptography and Network Security, Steganography, Green Computing, Big data analytics, Li-Fi Technology, Mathematical modelling of Social Area Networks, MOOCs, Quantum Computing etc. He has published more than 257 research articles in different Journals and conference proceedings.

Ms. Tiyasa Chatterjee is a student of St. Xavier’s College, currently pursuing M.Sc. in Computer Science. Her interests lie in the field of Machine Learning, Deep Learning, Coding, Animation, Cyber Security, AI and real-world project implementation of these fields.