Comparison of word embeddings in text classification based on RNN and CNN

*Merlin Susan David¹, and Shini Renjith¹

¹Department of Computer Science and Engineering, Mar Baselios College of Engineering and Technology, Thiruvananthapuram, Kerala, 695015, India
merlindavid92@gmail.com, shini.renjith@mbcet.ac.in

Abstract. This paper presents a comparison of word embeddings in text classification using RNN and CNN. In the field of image classification, deep learning methods like as RNN and CNN have shown to be popular. CNN is most popular model among deep learning techniques in the field of NLP because of its simplicity and parallelism, even if the dataset is huge. Word embedding techniques employed are GloVe and fastText. Use of different word embeddings showed a major difference in the accuracy of the models. When it comes to embedding of rare words, GloVe can sometime perform poorly. Inorder to tackle this issue, fastText method is used. Deep neural networks with fastText showed a remarkable improvement in the accuracy than GloVe. But fastText took some time to train when compared to GloVe. Further, the accuracy was improved by minimizing the batch size. Finally we concluded that the word embeddings have a huge impact on the performance of text classification models.

Keywords: Convolutional neural network, recurrent neural network, fastText, text classification, word embeddings.

1. Introduction

We have both structured and unstructured data at our fingertips. Chats, web pages, email, and other social media platforms are examples of unstructured data. Since the unstructured data is a rich source of information, extracting information or knowledge from these messy data is challenging and time consuming.

Humans can easily comprehend different languages but when it comes to computers, it is difficult for the machines to understand natural languages. But nowadays we have Siri, Alexa, Cortana, which will get activated at our command and perform actions accordingly. This is only possible due to Natural Language Processing (NLP) along with elements like deep learning and machine learning. NLP allows the machine to comprehend human language, measure its sentiments, and identify the important prominent features from it.

Basic NLP tasks include tokenization, stemming, lemmatization, and so on. In natural language processing, text classification is crucial. It is the process of applying labels to the input document or categorising it (Fig. 1). It can also be used to organize, store, and manage information effectively. Different applications of text classification include tag recommendation, spam detection, sentiment...
analysis, and intent analysis.

![Text classification diagram](image)

**Fig. 1.** Text classification

Manual categorization of the corpus is challenging and time-consuming. Automated text classification include Rule-based system, Machine learning-based system, and Hybrid approaches. Manual categorization of the corpus is challenging and time-consuming. Automated text classification include Rule-based system, Machine learning-based system, and Hybrid approaches.

2. Related Work

2.1 Text classification using string kernels [4].

The author of this study introduced a different kernel for comparing two text documents. The kernel is an inner product of sub-sequences of length k in feature space. In the text, a sub-sequence is an ordered sequence of k letters. The advantages are that once a proper kernel function is chosen, we can work in any dimensional space without incurring any additional costs. The most notable feature is that they use sequence alignment techniques to map the corpus to vectors without explicitly representing them. The disadvantages are, The results reported in the paper are very precursive (Fig. 2) and yet, there are lot of questions that still remains to be answered. The calculation of the new kernel takes longer, and more investigation is required to figure out how to proceed with this step of the calculation [4]. Other works in this area include [10, 11].

![Performance measures of different string kernels](image)

**Fig. 2.** Performance measures of different string kernels [4]

2.2 Comparison of SVM and some older classification algorithms in text classification tasks [5].

KNN and Naive Bayes are two well-known classifiers that are compared to the SVM. The performance of the classifier was assessed using 10-fold cross validation. Initial tests on 20Newsgroups to determine the optimal parameter settings. The value of C was also altered for a number of kernel functions. As a result, the default value C=100 is employed, along with a linear kernel. The benefits include the fact that the SVM does not require high precision to train. As the size of the feature space changes, so does the training time. The benefits include the fact that the SVM does not require high precision to train. As length of feature space alters, so does the training time. KNN and Naive Bayes have outstanding performance and are much quicker than SVM. The disadvantages are computational cost for training SVM for a huge training dataset [5]. Similar works in this area include [12, 13, 14].

2.3 Text classification based on multi-word with support vector machine [6].

This paper's major objective is to study the impact of employing multi-word textual in-cooperation on text classification. First, extract the multi-word from the document and it can be done using a linguistic
method or by using a statistical method. If there are fewer terms in the corpus, the terms will have no discriminative capability in classifying the documents. After extracting the multi-word from the document, documents are represented as multi-word. So we have two strategies

1) Decomposition strategy
2) Combination strategy

Information retrieval and text mining in multi-word is a recently investigated feature, and it has a number of advantages. There are three advantages to employing, Multi-word: It has a little size when compared to individual words, but it performs well. MCL, for example, has a precision of up to 0.8673. It is simple to extract many words from a corpus using corpus learning without using a dictionary. Multi-word phrases have more semantics and are more powerful than single words. As a result, if multi-words are used for information finding, patterns will be more interpretable and comprehensible.

The disadvantages are that due to a lack of theoretical proof, a broad judgement was not reached during the assessment. The mathematical examination of the statistical aspects of multi-words on text categorization is a significant problem that has to be addressed further. For more trustworthy work, additional inspection and research are required [6]. Other attempts in this area include [15, 16, 18].

2.4 Comparative effectiveness of CNN and RNN architectures for radiology text report classification [7].

Other published NLP experiments show that the accuracy measures are either superior or comparable. The results indicated the practicability of CNNs and RNNs in the classification tasks of imaging test reports and also supported the use of these techniques at a large scale in classifying text reports for other applications, including radiology patient prioritisation and clinical trial eligibility screening and imaging utilization.

They are also utilized in other areas of research like large scaled data production for automatic medical image interpretation. And perfect scores are yet to be achieved by neural nets.

RNN model accurately predicted the classes and situated the relevant sentences in the reports, but it is hard to agree upon how the model made that inference. Many of these errors require subtle reasoning to reach the correct conclusion, which may be a limitation of the models [7].

3. Convolutional Neural Network

Computers visualize things in a different way than we humans do. They only see things in terms of numbers and are called pixels. Because of their architecture, CNN differ from other types of neural networks. Other Neural Nets process their input by passing it through a sequence of layers. Each of the layer consists of a group of nodes, each of which is linked to all nodes in the previous layer. The predictions are represented by a fully linked layer and an output layer.

3.1 Architecture

Convolutional Neural Networks are a unique type of neural network. To begin, CNN comprises of 3 dimensions and they are depth, height and width. Furthermore, the nodes in one of the layer don’t link to all of the nodes in the following layer, merely a little portion (Fig. 3). Finally, the result is condensed into probabilistic scores grouped with depth axis. There are two halves to CNN. The portion about feature extraction. To detect the features, the network will perform a multiple convolutions operation
and pooling operations in this stage. On top of these retrieved features, fully linked layers would function as a classifier in the classification section. They will give the object on the screen a likelihood of being what the algorithm believes it is.

The most significant characteristics of a CNN is convolution. It is a mixture of two sets of data. The combination of two other functions to form a third function was referred to as convolution. Convolution is then applied to the input data with the help of a filter or kernel in the case of CNN to build a feature map. Then, sliding the filter function on the input, we perform a convolution. Matrix multiplication is also calculated at each point, and then the resulting is added to the feature map. The receptive field, is another term for this part of the filter. This filter is 3x3 in size. [19, 20].

![Fig. 3. Employed CNN architecture](image)

We performed multiple convolution operation on the input, using different filter each time. This will result in various feature maps. We aggregate all feature maps at the conclusion of the process and output them as the convolution layer's final result. To introduce non-linearity, we employ an activation function. Convolution's final output is processed through an activation function, the most often utilized of which is Rectified Linear Unit (ReLu). The stride of a convolution filter is the number of steps it moves each time. A stride in most case is 1, that means the filter will slide pixel by pixel. As the size of the stride increases, eventually it also the filter will cover larger interval of the input and there will be less overlap between cells.

![Fig. 4. Max pooling](image)

And Since the size of the feature map is less when compared to the input, we should prevent further shrinkage of the feature map. This is where padding comes into play [19, 20]. To prevent the feature map from being lowered, a layer of zeros is applied to result. Padding also increases performance by ensuring that the kernel and stride sizes are correct for the input. After convolutional layer there will be pooling layer between them. Convolution layer and pooling layer can be repeated many times. Dimensionality reduction is one of the advantage of pooling operation and hence the number of parameters, lowering computing costs and preventing over-fitting. Max pooling (Fig. 4) is the most often used pooling layer, that projects the highest value from every window. However, this window size must be selected ahead of time. This reduces the size of the feature map while preserving the important information [7,19].

After feature extraction part, the classification part which consist of fully connected layer that only accommodate 1-D information. For converting 3-D data to 1-D, the flatten function is used which is
available python. This will convert the 3-D data into a 1-D data. Last of the layers in CNN are fully connected. There will be nodes that are fully connected to the preceding layers. Training of CNN is very much similar to other neural networks, using back propagation or gradient descent. Our ultimate objective is to input photos into it, so that CNN may assign a probability to an target it believes it sees or to interpret an image with the words.

4. Recurrent Neural Network

The output from the previous step is provided as an input to the current step in recurrent neural networks. The inputs and outputs of other neural nets are independent of one other, but when it comes to word prediction, they are not. We need prior knowledge of the word that comes before it. RNN (Fig. 5) addresses this with the help of a hidden layer [8,19].

![Recurrent Neural Network](image)

**Fig. 5. Recurrent Neural Network**

The prominent characteristic of Recurrent neural net is its hidden state, that help it to remember the information of the sequence [8]. Recurrent neural networks are of two common variants which include Gated Recurrent Unit (GRU) and Long Short Term Memory (LSTM). RNN stores all the information that was previously calculated. It uses the same parameter for all the input and hidden layers so that it reduces the complexity of parameter.

Because each layer has individual weights and biases, all of the layers in classic neural networks are independent of one another. RNN, on the other hand, applies the same weights and biases to all layers, reducing the complexity of rising parameters and memorising past outputs by feeding each output into the next hidden layer.

4.1 Bidirectional LSTM

The idea behind Bidirectional LSTM is from bidirectional recurrent neural network. They process sequential data coming in both forward pass and backward pass and also have two hidden layers. It is already established that bidirectional networks are best when compared to unidirectional network. Unfolded bidirectional LSTM it consists of both forward and backward LSTM layers. The BiLSTM layer generates an output vector, $y$, in which each element is calculated using the equation 1 [9]

$$y_t = \sigma(h_t^+, h_t^-)$$  (1)

where sigma function is used for combining two output sequences.
5. Feature Engineering

Feature engineering is the process by which the text information will be changed into feature vectors and new features will be made utilizing the dataset. The way towards representing the words and documents utilizing a dense vector is called word embedding [1, 2, 3]. Word embeddings can be prepared with the given input document/corpus or with the pre-trained word embeddings like GloVe, fastText, and Word2Vec. After making a vocabulary that is made out of words. We create a matrix of low-dimensional word vectors. Furthermore, the matrix is provided as input to Neural Net [21].

GloVe is a method for collecting vector representations of words using an unsupervised learning algorithm. The final representation exhibits an intriguing linear substructure of the word vector space, which is based on collective global word-word co-occurrence insights from corpus [22].

FastText is a package that allows you to learn word representations. It is used for both supervised and unsupervised representations of sequence of words and the sentences to be trained. These embeddings are then incorporated for a variety of purposes, including data compression and transfer learning initializers. It is largely written in C++ and allows for multiprocessing during training. fastText also supports training models that use negative sampling, and softmax, such as the continuous bag of words or Skip-gram models [17].

Embeddings from Language Models (ELMo) is for learning words and their context, it uses a deep biLSTM language model. The deep BiLSTM architecture also allows ELMo to capture more of context-dependent aspects of words in the higher layers along with their syntax aspects in lower layers. This results in better word embeddings, and different representation of words depending on the context in which they are used.

Bidirectional Encoder Representations from Transformers (BERT) is built on the bidirectional concept from ELMo, but uses new transformer architecture to calculate word embeddings. It has been shown to produce excellent embeddings of word, achieving good performance on different NLP problems [24].

6. Dataset

TREC dataset collection has a question categorization consisting of domain, fact-based questions separated into several groups or labels. TREC dataset consist of 6 labels and has 5,452 training set data and 500 test set data. Models are assessed based on accuracy. The labels associated with TREC-6 are Abbreviation (ABBR), Entities (ENTY), Locations (LOC), Numeric values (NUM), Human beings (HUM), and Description and abstract concepts (DESC).

7. Classifier Comparison and Result

fastText: When it comes to embedding of rare words, it can sometime perform poorly. Inorder to tackle this issue, fastText method is used. It uses sub-word information so that rare words can be represented well. It still works based on skip-gram model. This model enhance the performance on syntactical problems fundamentally, yet very little in semantic questions. fastText consider individual words as made out of character n-grams. Since training is based on character n-gram, it takes more time to generate fastText embeddings compared to word2Vec [17].

GloVe: Glove's goal is rather straightforward - it executes word vectors in order to obtain sub-linear correlations in vector space. Thus, it proved that GloVe pull off better than Word2Vec. Besides, GloVe incorporate more reasonable significance into word vectors by considering the relation between the word
pair rather than the word to word. GloVe takes up a lot of memory since it is trained on the co-occurrence matrix of words [22].

Word2Vec: Word2Vec regard each word in the document as single entity and generates a vector for each word. Word2Vec is similar to GloVe. When it comes to rare words Word2Vec perform poorly because, in Word2Vec a rare word has fewer neighbor, compared to a word that commonly occurs. The model is hard to train if we employ the softmax function, since the number of categories is enormous [23].

Results show that fastText has comparatively higher accuracy in both RNN and CNN when it is applied to the TREC dataset (Tables 1 and 2). Result may vary for other dataset based on attributes like dataset size, and so on.

| Word Embeddings | Accuracy | Loss |
|-----------------|----------|------|
| GloVe           | 89.09%   | 10.91% |
| FastText        | 90.93%   | 9.07%  |

| Word Embeddings | Accuracy | Loss |
|-----------------|----------|------|
| GloVe           | 24.59%   | 74.51% |
| FastText        | 83.33%   | 16.67% |

8. Conclusion

In natural language processing, text classification is crucial. It can be used for many applications like sentiment analysis, spam detection, review analysis, and more. For all these applications a faster, efficient, and logical system is required. A text classification model must be capable of classifying text data even if it is small or large, must be able to identify meaningful words from the text. For this work, the comparison of the text classification using deep neural networks like CNN and RNN is performed. Both fastText and GloVe word embedding techniques are applied to both. Deep neural networks with fastText showed a remarkable improvement in the accuracy of than glove. But fastText took more time to train than a GloVe. Further, the accuracy was improved by minimizing the batch size. Finally we concluded that the word embeddings have a huge impact on the performance of text classification models.

References

[1]. Guo, Bao, Chunxia Zhang, Junmin Liu, and Xiaoyi Ma. "Improving text classification with weighted word embeddings via a multi-channel TextCNN model." Neurocomputing 363 (2019): 366-374. https://doi.org/10.1016/j.neucom.2019.07.052

[2]. Helaskar, Mukund N., and Sheetal S. Sonawane. "Text Classification Using Word Embeddings." In 2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA), pp. 1-4. IEEE, 2019. https://doi.org/10.1109/ICCUBEA47591.2019.9129565

[3]. Rabut, Benedict A., Arnel C. Fajardo, and Ruji P. Medina. "Multi-class document classification using improved word embeddings." In Proceedings of the 2nd International Conference on Computing and Big Data, pp. 42-46. 2019. https://doi.org/10.1145/3366650.3366661
[4]. Lodhi, Huma, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. "Text classification using string kernels." Journal of Machine Learning Research 2, no. Feb (2002): 419-444. https://doi.org/10.1162/153244302760200687

[5]. Colas, Fabrice, and Pavel Brazdil. "Comparison of SVM and some older classification algorithms in text classification tasks." In IFIP International Conference on Artificial Intelligence in Theory and Practice, pp. 169-178. Springer, Boston, MA, 2006. https://doi.org/10.1007/978-0-387-34747-9_18

[6]. Zhang, Wen, Taketoshi Yoshida, and Xijin Tang. "Text classification based on multi-word with support vector machine." Knowledge-Based Systems 21, no. 8 (2008): 879-886. https://doi.org/10.1016/j.knosys.2008.03.044

[7]. Banerjee, Imon, Yuan Ling, Matthew C. Chen, Sadid A. Hasan, Curtis P. Langlotz, Nathaniel Moradzadeh, Brian Chapman et al. "Comparative effectiveness of convolutional neural network (CNN) and recurrent neural network (RNN) architectures for radiology text report classification." Artificial intelligence in medicine 97 (2019): 79-88. https://doi.org/10.1016/j.artmed.2018.11.004

[8]. Kim, Jihyun, Jaehyun Kim, Huong Le Thi Thu, and Howon Kim. "Long short term memory recurrent neural network classifier for intrusion detection." In 2016 International Conference on Platform Technology and Service (PlatCon), pp. 1-5. IEEE, 2016. https://doi.org/10.1109/PlatCon.2016.7456805

[9]. Cui, Zhiyong, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction." arXiv preprint arXiv:1801.02143 (2018).

[10]. Gonçalves, Teresa, and Paulo Quaresma. "Text classification using tree kernels and linguistic information." In 2008 Seventh International Conference on Machine Learning and Applications, pp. 763-768. IEEE, 2008. https://doi.org/10.1109/ICMLA.2008.78

[11]. Zheng, Shiqiang, Yujui Yang, Haiping Wu, and Wenhuang Liu. "Chinese Text Classification Using Key Characters String Kernel." In 2009 Fifth International Conference on Semantics, Knowledge and Grid, pp. 113-119. IEEE, 2009. https://doi.org/10.1109/SKG.2009.59

[12]. Kalcheva, Neli, Milena Karova, and Ivaylo Penev. "Comparison of the accuracy of SVM kernel functions in text classification." In 2020 International Conference on Biomedical Innovations and Applications (BIA), pp. 141-145. IEEE, 2020. https://doi.org/10.1109/BIA50171.2020.9244278

[13]. Basu, Atreya, Christine Walters, and M. Shepherd. "Support vector machines for text categorization." In 36th Annual Hawaii International Conference on System Sciences, 2003. Proceedings of the, pp. 7-pp. IEEE, 2003. https://doi.org/10.1109/HICSS.2003.1174243

[14]. Dilrukshi, Inoshika, and Kasun De Zoysa. "Twitter news classification: Theoretical and practical comparison of SVM against Naive Bayes algorithms." In 2013 International Conference on Advances in ICT for Emerging Regions (ICTer), pp. 278-278. IEEE, 2013. https://doi.org/10.1109/ICTer.2013.6761192

[15]. Silva, Catarina, and Bemardete Ribeiro. "Text classification from partially labeled distributed data." In Adaptive and Natural Computing Algorithms, pp. 445-448. Springer, Vienna, 2005. https://doi.org/10.1007/3-211-27389-1_107

[16]. Aggarwal, Charu C., and ChengXiang Zhai. "A survey of text classification algorithms." In Mining text data, pp. 163-222. Springer, Boston, MA, 2012. https://doi.org/10.1007/978-1-4614-3223-4_6

[17]. Yao, Tengjun, Zhengang Zhai, and Bingtao Gao. "Text Classification Model Based on fastText." In 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIIS), pp. 154-157. IEEE, 2020. https://doi.org/10.1109/ICAIIIS49377.2020.9194939

[18]. He, Ming, Jianjun Sun, and Ying Cheng. "Text Classification Based on Naive Bayes: A Review." Information Science 34 (2016): 147-154.

[19]. Liu, Tengfei, Shuangyuan Yu, Hongtao Zhang, and Hongfeng Yin. "Recurrent Neural Networks and Convolutional Neural Networks for Text Classification." Computer Engineering &
[20]. Cai, Jingjing, Jianping Li, Wei Li, and Ji Wang. "Deeplearning model used in text classification." In 2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), pp. 123-126. IEEE, 2018. https://doi.org/10.1109/ICCWAMTIP.2018.8632592

[21]. Wang, Bin, Angela Wang, Fenxiao Chen, Yuncheng Wang, and C-C. Jay Kuo. "Evaluating word embedding models: Methods and experimental results." APSIPA transactions on signal and information processing 8 (2019). https://doi.org/10.1017/ATSI.P.2019.12

[22]. Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543, 2014. http://dx.doi.org/10.3115/v1/D14-1162

[23]. Goldberg, Yoav, and Omer Levy. "word2vec Explained: deriving Mikolov et al.’s negative-sampling word-embedding method." arXiv preprint arXiv:1402.3722 (2014).

[24]. Gao, Zhengjie, Ao Feng, Xinyu Song, and Xi Wu. "Target-dependent sentiment classification with BERT." IEEE Access 7 (2019): 154290-154299. https://doi.org/10.1109/ACCESS.2019.2946594