Semantic Article Categorisation using Convolutional Neural Networks

Ramee El-Shabasei
Australian National University

Abstract

We apply a Convolutional Neural Network (CNN) to two different numerical text representations of news articles for binary classification from the SatiricLR dataset. Benchmarking against a simple CNN, we experiment with network depth by employing different methods to reduce input parameter space whilst simultaneously increasing the number of convolutional layers. We empirically evaluate each experiment and present competitive results over previous accuracy on the SatiricLR dataset.

1 Introduction

Text classification has various forms, we explore a prevalent and readily applicable form, providing semantic labels (categories) given a textual input (e.g. a news article). This problem of text categorisation has been studied for decades [7], and has benefited from advances in the wider NLP field. This includes greater study in text representation as vectors, imperative for many ML methods including the neural networks utilised in the present work. Without a numerical vector representation, at present there is no way of handling text input [16]. Initial algorithms targeting categorisation involved purely the use of statistics and vector space distances, including Rocchio's algorithm and a Naive Bayes approach [7, 11].

A prominent representation method is the text-frequency inverse document-frequency (TF-IDF) method, though not often applied to a neural network based models due to its lack of word-based embeddings as a weighted bag of words [3]. For neural networks, a key method of representing text has been the word2vec approach [19]. This approach uses a shallow linear neural network to learn text representations that provide a vector based word embedding and has seen popular use in NLP [3].

In the present work both of the above methods are explored. We further build upon previous work by the author exploring the problem of text categorisation using neural networks. In this instance, we reduce the number of classes into a binary problem: satire or non-satire, using the SatiricLR dataset. Whilst this may appear to simplify the task, the individual class is arguably more complex. Satire is often distinguished by selective use of figurative language and mimicry of actual news but with implied humour, hence the reduction in the number of categories [2].

Recent work has shown the success of applying CNNs to text, building upon their success in computer vision [13], [15]. By applying CNNs to text, we are able to better capitalise upon textual structure [17], including feature detection of names [4] and effective character-level features (e.g. verb suffixes) [23]. We investigate this in the context of satire detection by empirical analysis of various CNNs in their application to humour detection.

2 Method

2.1 Dataset

The binary classification consists of 3411 labels for 3411 articles. Almost half (1705) of the articles are non-satirical, sourced from the online mediums of Reuters, CNET, and CBS News. The 1706 remaining satirical articles are taken from Daily Currant, DailyMash, and other outlets [8]. Political, entertainment and technology articles are drawn with similar numbers for each subtype, the distribution is provided in Table 1. To generate the test set we randomly sample 1/7 of the articles, maintaining a roughly equal

Table 1: Distribution of articles within the SatiricLR dataset.

	Satire	Non-Satire	Total
Politics	545	574	1119
Entertainment	557	578	1135
Technology	604	553	1157
Total	1706	1705	3411

distribution between satire and non-satire articles. For each article, only the core text was extracted: titles and metadata including authorship and publication dates have been removed.

2.2 Representations

To process the textual input, it is necessary to represent the text in vector form. Despite success with character-level encodings [26], [14], in the context of satire it seems appropriate to keep with convention and encode words [18]. However, before embedding, there are some variations in document size, to resolve this we apply zero padding, i.e. zero vectors will act as substitute words appended to the end of documents, padding each document up to the maximum length document for fixed input size into the networks. We then tokenise the document and generate both a TF-IDF representation and a word vector embedding using the GloVe model. Finally, in both representations we preserve stop words punctuation marks, as they have previously provided a feature with high correlation to humour [6].

2.2.1 TF-IDF

A key difficulty in processing language is representation of text and words. In this case of SatiricLR, the sheer prevalence of certain words (e.g. profanity and negativity) in certain posts often indicates the article type. This suggests a standard bag of words (BoW) approach is suitable, but this approach can be improved with incorporation of frequency information based on the bag of words representation [1].

Utilising inverse-frequency over the interval [0, 1] intuitively provides better outcomes (i.e. emphasises the commonality of words by weighting them relative to other words). Consider groups that are indicated not by common words, such as 'the', but rather by a diverse number of unusual words (e.g. pop-culture references typical of satirical analogies). By taking a word's frequency within its text, we can determine its use and relevance to the article, but by multiplying it by its inverse document frequency, we are able to gain information of frequency in relation to the entire corpus of words in the SatiricLR set. Hence, words that are characteristic to a group will not result in a low inverse-document frequency that results in a negligible weighting, whereas words which do appear in every document will, e.g. English determiners ('the', 'this' etc.). This TF-IDF representation, which first builds a bag of words and converts via a transform, follows the below formula in (1)

$$\begin{aligned} \text{TF-IDF}(d,t) &= tf(t)idf(d,t) \\ idf(d,t) &= log\big(\frac{N+1}{df(d,t)+1}\big) \end{aligned} \tag{1}$$

where $df(d,t)$ provides the number of documents in which t appears, $tf(t)$ its within document frequency. A key consequence of this representation is the size of the input, each sentence will be converted into an array of float values as opposed to a space-efficient binary indexing that is common in NLP (see [21]).

As this representation is essentially a modified bag of words, we lose spatial information of adjacent words, which reduces the applicability of a CNN. However, application of a CNN in this context may provide useful information in regards to the nature of feature detection in a trained CNN model.

2.2.2 Skip-gram

This second representation is based on the word2vec model, specifically the GloVe negative skip-gram model [20]. This model vectorises words by utilising a pretrained neural network to develop representations. In contrast to the TF-IDF approach, the similarity between words is not disproportionately defined by the frequency of the words. By using skip-gram method, vectors are more closely tied to their expected context (in a defined window), which in theory provides the ability to determine patterns inherent in each word [19]. This provides useful prior information in the word embedding which should speed up and assist the training of later neural network approaches. Whilst the GloVe model is still based on the occurrence counts of words, in that it differs primarily by relying in co-occurrence probabilities, it learns the word representation by developing much richer features via a shallow neural network approach, including name recognition. The GloVe model used in the present work is trained on 1.6 billion tokens from Wikipedia 2014 plus the Gigaword 5 corpus containing 4.3 billion tokens. To facilitate fast training of our deep learning approach, we utilise the 100-dimensional variation, which provides a more compact representation of words but at the cost of representational power versus the 300 dimensional embedding. In any case, this representation is typically less compact overall than the TF-IDF approach which is proportional to the vocabulary size as opposed to the GloVe model which for each document is proportional to the number of tokens embedded as a 100 dimensional vector.

Not all words in the SatiricLR dataset are present in the pretrained vocabulary, in this instance we follow [13] and randomly initialize their embedding - unknown words are treated as random noise. However, our process differs, we take the 10 random existing word embeddings and produce a corresponding mean vector, aiming to maintain similar variance. Where computational performance is not an issue, it seems better to take a fixed number of most infrequent words in the document and then generate the mean vector. Ergo, if the word is not in the vocabulary it seems best to assume it is uncommon and, hence, generate it from uncommon word embeddings.

2.3 Model

We investigate two separate convolutional networks.

First, a Simple CNN over the text representation using 1D kernels of varying sizes grouped into layers is used. The network consists of two convolutions, first a length 7 kernel with 256 filters is convolved over the

input volume, followed by batch normalisation (BN), RReLU and Max Pooling with a kernel size and stride of 2. A second convolution is applied but with a smaller length 3 kernel, in this instance depth reduction occurs, applying only 128 filters. After applying BN and RReLU, the resultant volume is passed to a fully connected layer which results in a binary output after softmax. This network provides fast computations being relatively shallow whilst still allowing us to employ the benefits of convolutions (added contextual information and shared weights).

Second, we extend the Simple model by increasing network depth and, thus, increasing representation power. Though we wish to increase the number of input parameters and develop heirarchical abstractions by increasing the depth, we wish to maintain similar GPU requirements: it should be able to train on a single GPU within a day. With this in mind, we do not develop depth to the level of a 50-layer ResNet, and further we design the model around the notion of one-dimensional textual input.

Deep CNNs for text classification have been developed in the past, ranging from 6 layers [26] to as much as 29 layers [5]. Though [26] examined character-level text representation, the success in nearing state of the art results with only small increases in network depth highlight that textual tasks do not necessarily require the number of layers characteristic of image classification tasks. An experimental analysis of a variety of sixteen-layer convolution models on both word and text representation provided improvements over six layer or other ML approaches [15]. However, as the improvements were relatively minor, we take an intermediary approach and utilise 8-convolution layers, each followed by BN and RReLU.

Simply increasing the number of layers without further optimisation is unlikely to assist. This is verified by our suboptimal testing accuracy in Table 2, suggesting the network runs into the degradation problem [9] as well as possibly requiring more epochs to train than possible within the constraints. In this network, we extended the original network by simply adding convolutions intermediary convolutions with 256 filters with kernel size 3.

We optimise the model by dividing the layers into three groups of two convolutional layers. We have reduced the layers slightly but we increase the width, this allows us to make better utilisation of GPUs [24]. After two convolutions with 256 filters, we add a residual connection using a standard identity mapping [9]. The residual mappings are of the form

$$y_i = F(y_{i-1}) + y_{i-1}$$

I.e. we no longer model the intermediary output of $y_i = F(y_{i-1})$ which, despite theoretical equivalence in context, proves to be an easier task experimentally. We utilise the standard highway path but with full pre-activation as part of the identity mappings [10].

2.4 Training Procedure

Experimentally, high testing and training accuracy was achieved after a small number of epochs. Coupled with computational constraints, we capped all experiments at 100 epochs, which is reasonable for a standard GPU to compute relatively quickly.

A smaller learning rate is often desirable with deeper networks to prevent the gradient from exploding [22], however due to the smaller number of epochs this must be coupled with the limited number of iterations. We set the learning rate to 0.01 but add decay: at epochs 10, 30, and 60 the learning rate is reduced by half, i.e. the final learning rate is 0.00125.

Following [12], we utilise SGD with a batch size of 16 in place of more complex optimisers due to the proven generalising capabilities. Similarly, the experimental results in [25] demonstrate the proven performance of RReLU in reducing overfitting and increasing overall accuracy, partially due to the ability to reduce the possibility of dead units. Thus, we also employ RReLU in place of standard ReLU for all activations.

3 Results

Under the Simple CNN model, the GloVe approach provides relatively good results, but still below the existing dataset best in [8]. As per Table 2, at 100 epochs the deep model is able to compete with the BoW approach but fails to improve upon the result, however it is possible that further accuracy increases may occur as partially suggested by the improving and changing loss in Figure 2. All models are able to minimise error very quickly, in addition it appears possible that a well designed shallow network may actually provide better results than a deep variation, given the limited dataset size and the absence of difficulties in achieving reasonable test accuracy (i.e. avoiding the optimisation issues of depth but improving the design to increase the accuracy on par with the Deep (Optimised) version).

Table 2: Summary of testing accuracy (after 100 epochs of training). A comparison of both text representations and models are provided, each cell contains the Testing Accuracy (%) to two decimals.

	GloVe	TF-IDF
Simple	81.92%	71.34%
Deep (Unoptimised)	69.03%	-
Deep (Optimised)	90.01%	60.13%
BoW (Means) [8]	93%	

Table 2 shows that the GloVe representation provides an empirically superior prior to a TF-IDF representation. This suggests that TF-IDF is not suitable for CNNs as they are vocabulary based lacking contextual

Figure 1: Error over 100 epochs for the Simple CNN approach using GloVe.

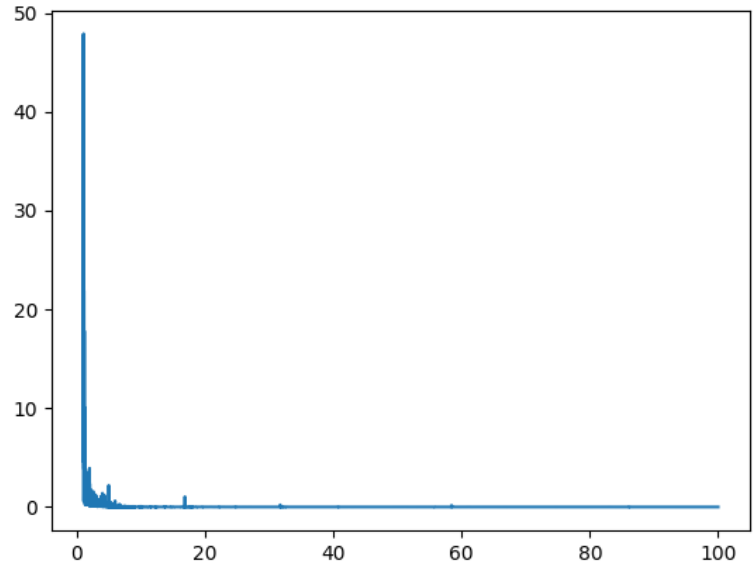
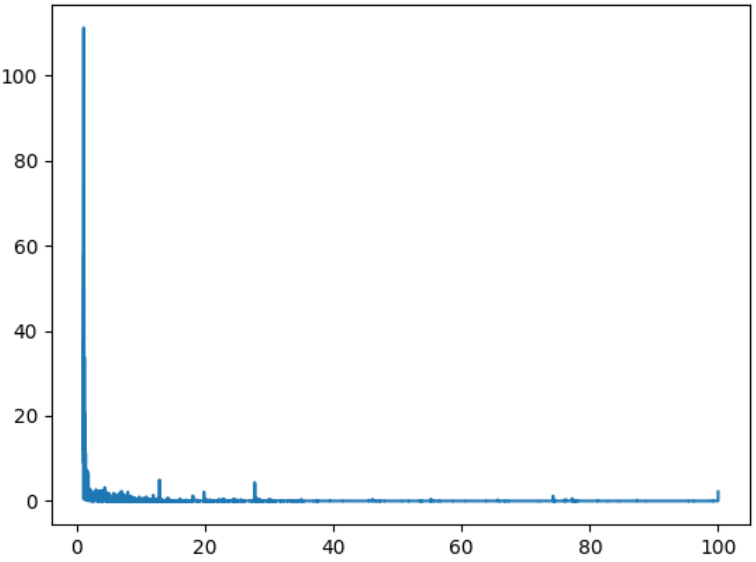


Figure 2: Error over 100 epochs for Deep (Optimised) approach using GloVe.



information such as word adjacency that GloVe provides. The Deep model performs considerably worse under TF-IDF than its simple counterpart, this is possibly due to the large number of convolutions being unable to extract features from the non-spatial representation, at least within the limitation of 100 epochs. Ultimately, whilst all models fall below the previous best BoW approach, further computational power in this case may lead to superior test accuracies. Whether clever CNN design can negate the need for increased epochs in general is unclear.

4 Conclusion

We have shown that different text representations can materially affect accuracy when utilising CNNs for text classification. However, though we have explored multiple text representations, there is a more fundamental consideration of processing that has been avoided. The characters that should be removed when generating tokens is a tricky question, e.g. how to deal with quotation marks (e.g. treat as a separate token). In our results, the binary text classification task proved achievable to high accuracy without requiring intensive training (several hours on a normal GPU in many cases). Though producing higher input parameter

space, the GloVe model is empirically a more suitable prior and comes at negligible extra computational cost in most instances. Further improvements in design to the Simple CNN, including changing the optimiser to RMSProp and utilising dropout, may prove beneficial in improving the test accuracy. It is clear from all the network models that CNNs appear to have strong applicability to text based tasks, with careful construction of text representation. Exploration of different CNNs designed for text (e.g. character-level networks) to larger classification tasks without high accuracy results currently may be a useful test of this. Finally, we note that the present dataset is still relatively small (compare to the nearly 20,000 articles in the 20-newsgroups dataset), expanding the dataset by incorporating new sources and article varieties may assist in highlighting the benefits of a deep learning approach given more training data.

References

[1] Akiko Aizawa. "An information-theoretic perspective of tf–idf measures". In: *Information Processing & Management* 39.1 (2003), pp. 45–65.

[2] Clint Burfoot and Timothy Baldwin. "Automatic Satire Detection: Are You Having a Laugh?" In: *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*. ACLShort '09. Suntec, Singapore: Association for Computational Linguistics, 2009, pp. 161–164.

[3] J. Camacho Collados and M. Taher Pilehvar. "From Word to Sense Embeddings: A Survey on Vector Representations of Meaning". In: *ArXiv e-prints* (May 2018). arXiv: 1805.04032.

[4] Jason P. C. Chiu and Eric Nichols. "Named Entity Recognition with Bidirectional LSTM-CNNs". In: *CoRR* abs/1511.08308 (2015). arXiv: 1511.08308.

[5] Alexis Conneau et al. "Very Deep Convolutional Networks for Natural Language Processing". In: *CoRR* abs/1606.01781 (2016). arXiv: 1606.01781.

[6] Dmitry Davidov, Oren Tsur, and Ari Rappoport. "Enhanced Sentiment Learning Using Twitter Hashtags and Smileys". In: *COLING*. 2010.

[7] A.G Fallis. "Text Categorisation: A Survey". In: *Journal of Chemical Information and Modeling* 53.9 (1999), pp. 1689–1699. arXiv: arXiv:1011.1669v3.

[8] Alice Frain and Sander Wubben. "SatiricLR: a Language Resource of Satirical News Articles". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Ed. by Nicoletta Calzolari (Conference Chair) et al. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016.

[9] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385.

[10] Kaiming He et al. "Identity Mappings in Deep Residual Networks". In: *CoRR* abs/1603.05027 (2016). arXiv: 1603.05027.

[11] Thorsten Joachims. "A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization". In: *the 14th International Conference on Machine Learning (ICML '97)* (1997), pp. 143–151.

[12] Nitish Shirish Keskar and Richard Socher. "Improving Generalization Performance by Switching from Adam to SGD". In: *CoRR* abs/1712.07628 (2017). arXiv: 1712.07628.

[13] Yoon Kim. "Convolutional Neural Networks for Sentence Classification". In: (2014). arXiv: 1408.5882.

[14] Yoon Kim et al. "Character-Aware Neural Language Models." In: *AAAI*. 2016, pp. 2741–2749.

[15] Hoa T. Le, Christophe Cerisara, and Alexandre Denis. "Do Convolutional Networks need to be Deep for Text Classification ?" In: *CoRR* abs/1707.04108 (2017). arXiv: 1707.04108.

[16] Q Le, T Mikolov - International Conference on Machine Learning, and undefined 2014. "Distributed representations of sentences and documents". In: *Jmlr.Org* 32 (2014).

[17] Xuezhe Ma and Eduard H. Hovy. "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF". In: *CoRR* abs/1603.01354 (2016). arXiv: 1603.01354.

[18] Amit Mandelbaum and Adi Shalev. "Word Embeddings and Their Use In Sentence Classification Tasks". In: *CoRR* abs/1610.08229 (2016). arXiv: 1610.08229.

[19] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *CoRR* abs/1301.3781 (2013). arXiv: 1301.3781.

[20] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543.

[21] Juan Ramos. “Using TF-IDF to Determine Word Relevance in Document Queries”. In: *Proceedings of the first instructional conference on machine learning* (2003), pp. 1–4.

[22] M. Ravaut and S. Gorti. “Gradient descent revisited via an adaptive online learning rate”. In: *ArXiv e-prints* (Jan. 2018). arXiv: 1801.09136 [stat.ML].

[23] Cicero Dos Santos and Bianca Zadrozny. “Learning Character-level Representations for Part-of-Speech Tagging”. In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1818–1826.

[24] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. “Wider or Deeper: Revisiting the ResNet Model for Visual Recognition”. In: *CoRR* abs/1611.10080 (2016). arXiv: 1611.10080.

[25] Bing Xu et al. “Empirical Evaluation of Rectified Activations in Convolutional Network”. In: *CoRR* abs/1505.00853 (2015). arXiv: 1505.00853.

[26] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. “Character-level Convolutional Networks for Text Classification”. In: *CoRR* abs/1509.01626 (2015). arXiv: 1509.01626.