CS230

Neural Networks for Topic Modeling

Gobi Dasu (gdasu) · Rahul Makhijani (rahulmj) · Michelle Song (yingze)

1 Introduction

Topic detection has a wide range of applications in industry and life. For instance, for a task of company classification - given a description of SpaceX, the output category would be Aerospace. For a task of news categorization - given an article input, we may want to label it as politics, sports, etc. Efficiently and accurately categorizing text is still a subject of research, and improvements over baselines will be well-received by industry. Topic detection models (classifiers) are well studied, and traditional methods such as latent dirichlet allocation (LDA) and bag-of-word term frequency models (TFIDF) are still quite popular. There have been recent studies (Zhang et. al. 2015) that explain how character level ConvNets can perform better than traditional TFIDF [1]. However this improvement is only present when training on the largest of datasets (millions of samples). We hence have decided to explore the question of which are the best deep learning models for topic classification given mid-size data (10-20K examples).

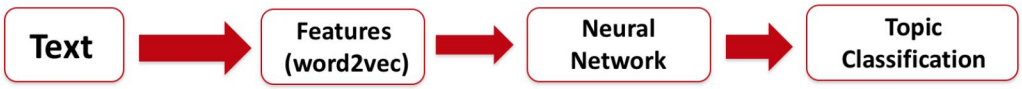

Figure 1: Overview of the Project.

2 Related work

A large section of initial techniques of text classification are based on words, in which they process a sequence of words using recurrent neural networks (RNN). The main contribution of an RNN is to use a parser that specifies the order in which the word embeddings are combined [10]. The most popular of RNNs are LSTMs [9]. LSTM learns much faster and it solves complex tasks that have never been solved by previous algorithms. However, it lacks the ability of learning task-relevant features.

Another stream of papers started to deploying knowledges of convolutional neural networks [1] [3] to learn more structures of each task. In principle, a fully connected one hidden layer neural network can learn any real valued function, but a deeper convolutional architecture can usually achieve better results [3]. Zhang et al., 2015 first perform sentiment analysis at the character level, using up to six convolutional layers, followed by three fully connected classification layers. Convolutional kernels of size 3 and 7 are used, as well as simple max-pooling layers. Conneau et al. 2017 present a new architecture (VDCNN) for text processing which operates directly at the character level and uses only small convolutions and pooling operations. They are able to show that increased depth of network layers increase performance, and their architecture uses up to 29 convolutional layers.

Given this research we decided to explore basic CNNs, LSTMs, and Character-level CNNs.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

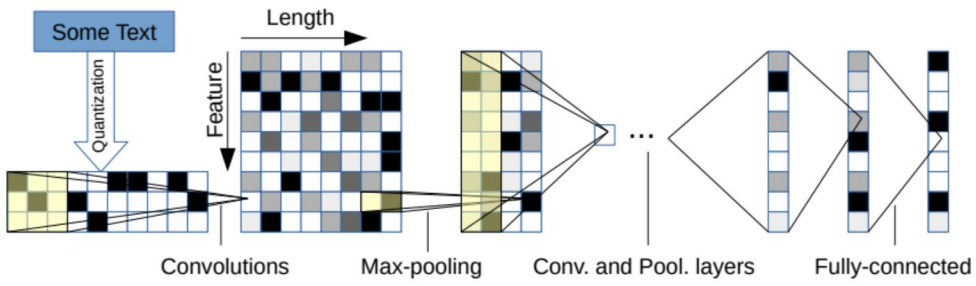

Figure 2: Illustration of the character level CNN model

3 Dataset and Features

We use the Reuters data and the Reuters dataset [7] has 20K examples (135 topic categories). An example from the snapshot from the data is

HOUSTON OIL TRUST HALTS ROYALTIES HOUSTON, April 7 - Houston Oil Trust said there will be no royalty funds available for distribution to unit holders in April. It also said that based on recent independent petroleum engineers' estimates of Oct 31, 1986, there may be no amounts available for distribution the rest of the year.

Topic Category : earn

We split the data into training, dev and test sets consisting of 14668, 3455 and 3455 sample points respectively.

We used the Reuters dataset for running our long-short term memory (LSTM) recurrent neural network, simple CNN network, and simplified character level CNN described in the next few sections. None of these methods achieved great performance because the amount of data is small. However, they do serve as baselines. For the LSTM and simple CNN network pre-processing involved extracting features using word2vec, and for the character level CNN, we had to one-hot encode the text characters and only include 1014 per example.

4 Methods

We used the standard deep learning algorithms of backprop and SGD with the AdamOptimizer. The interesting part of our methods were actually the network architectures that were used to predict, based on input text, the 135 Reuters category labels. So we describe the architectures below:

4.1 Character-level Convolutional Networks

We planned to use the same character level ConvNets that Zhang et. al. used. We removed the 2nd, 4th, and 5th convolutions due to avoid the high computational cost. The architectures are laid out in the paper by Zhang et. al. and we used Tensorflow to set them up. There are 9 layers deep with 6 convolutional layers and 3 fully-connected layers [1]. Convolutional layers include a convolution, max pool, and ReLU.

Initially we were planning on doing transfer learning, using a CharCNN (pretrained on the 1.4m Yahoo Answers dataset) and training it on the smaller Reuters dataset (14K). However, we found LSTM more promising so we decided to focus on hyperparameter tuning of LSTM.

4.2 ConvNet

This architecture is a basic two layer fully connected CNN . The non linear activation functions are RELU and softmax. We use 50 hidden states for the hidden layer.

Diagram

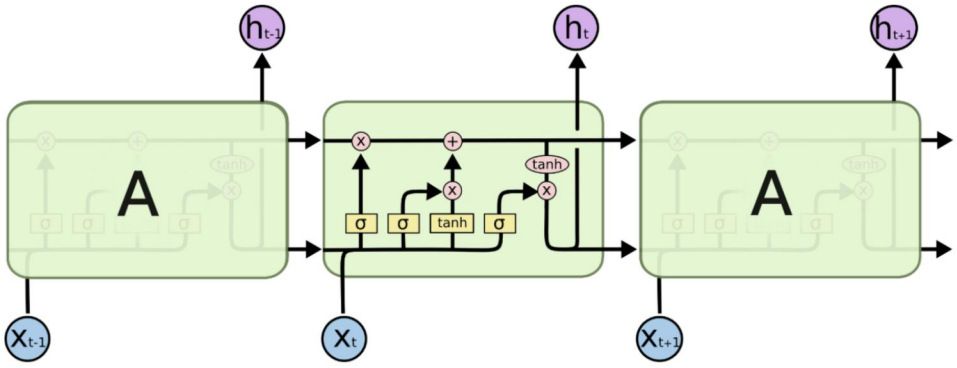

Figure 3: Diagram of an long short term memory, from Christopher Olah's LSTM blog post [8]

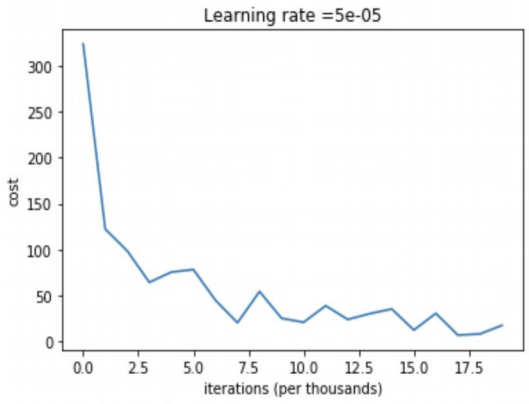

Figure 4: Cost analysis of the Basic CNN as function of the iterations. We used a hidden layer with 50 states, a learning rate of 0.0005, minibatches of 100 examples, and a cross entropy loss function. We trained for 5 epochs of training using Adam, and achieved 0.66 training accuracy and 0.53 test accuracy.

4.3 LSTM

We use a 100 state sequential model and dropout of 0.3 for the regularization.The final layer is softmax. The LSTM was coded in Keras

LSTM is a recurrent neural network that can learn long term dependencies by maintaining forget, input, and output gates, which each act as conventional neurons that perform regulation of what information to keep track of. This sort of architecture allows NLP tasks to gain a much better understanding of text, because text is temporal in nature. In other words, understanding textual information at time t involves remembering information from an earlier time. In order to do topic classification accurately it is important for the AI to actually understand the text.

5 Results

We ran the simulations for Basic CNN, LSTM and the character level CNN. The figure below shows the cost as a function of iterations for the Basic CNN. It was run for five epochs with a learning rate of 0.0005. We used the multiclass accuracy function as a metric to calculate the performance.

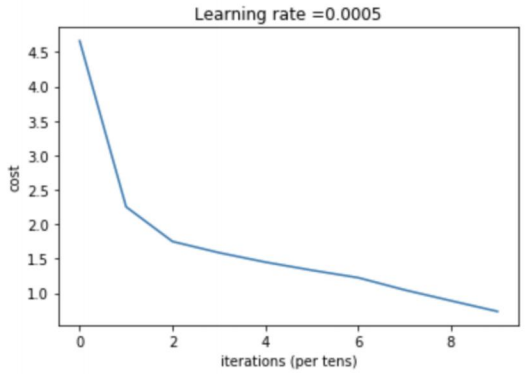

Figure 5: Cost analysis of the Character Level CNN as function of the iterations. This model was computational expensive so we only trained on 500 Reuters training examples and tested on 500 examples. Cross-entropy loss was used. 10 epochs of training were performed, with a learning rate of 0.0005, the Adam Optimizer, and minibatches of 100. We achieved 50.4% test accuracy.

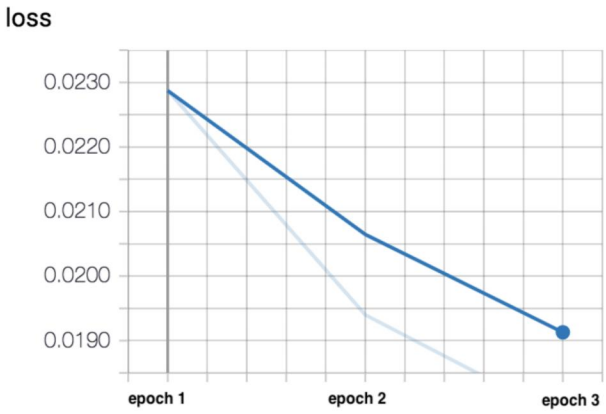

Figure 6: Cost analysis of the LSTM as function of the epochs. We ran the LSTM for 3 epochs. Minibatches consisted of 128 examples, and the learning rate was 0.0005 for the Adam optimizer.

Hidden layer Size	Learning Rate	Dev Accuracy
50	0.00005	0.537482
50	0.0001	0.564399
50	0.005	0.514616
100	0.00005	0.541534
100	0.0001	0.561795
100	0.005	0.575398
150	0.00005	0.566136
150	0.0001	0.563821
150	0.005	0.575398

Table 1: Hyperparameter tuning for Basic CNN

Dropout Rate	Hidden layer Size	Epocs	Dev Accuracy
0	100	3	0.1716
0	100	5	0.348
0	200	3	0.5314
0	200	5	0.2124
0	300	3	0.0724
0	300	5	0.5876
0.4	100	3	0.1774
0.4	100	5	0.3797
0.4	200	3	0.6634
0.4	200	5	0.4654
0.4	300	3	0.6234
0.4	300	5	0.6208
0.8	100	3	0.2538
0.8	100	5	0.1922
0.8	200	3	0.1514
0.8	200	5	0.5899
0.8	300	3	0.1838
0.8	300	5	0.6394

Table 2: Hyperparameter tuning for LSTM

6 Conclusion

We report on the performance of several architectures under the task of topic classification. Topic detection is widely applicable in industry and diverse domains, and there is no clear winner algorithm in the field. We picked vastly different architectures and a generic dataset (Reuters) in order to make sure our conclusions regarding architectures are not biased. We found a simple CNN, character level CNNs, and LSTM achieved 0.532, 0.5, and 0.66 test accuracy respectively (after hyperparameter tuning) on Reuters topic prediction.

7 Contributions

Rahul Makhijani coded up basic CNN and LSTM and hyper parameter tuning and the final report. Gobi Dasu coded up the Character-level Convolutional Networks and worked on the final report. Michelle Song worked on the poster and the final report.

References

[1] Zhang, Xiang & Junbo Zhao & Yann LeCun. "Character-level convolutional networks for text classification." *Advances in neural information processing systems. 2015.*

[2] Conneau, Alexis, et al. "Very deep convolutional networks for natural language processing." *arXiv preprint arXiv:1606.01781(2016).*

[3] https://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection

[4] Joachims, Thorsten. "Text categorization with support vector machines: Learning with many relevant features." *European conference on machine learning. Springer, Berlin, Heidelberg, 1998.*

[5] Hochreiter, Sepp & Jürgen Schmidhuber. "Long short-term memory." *Neural computation 9.8 (1997): 1735-1780.*

[6] Socher, Richard, et al. "Semi-supervised recursive autoencoders for predicting sentiment distributions." Proceedings of the conference on empirical methods in natural language processing. *Association for Computational Linguistics, 2011.*

[7] Reuters Topic Classification, Ambodi, https://github.com/ambodi/topic-classification-reuters-21578/blob/master/model/lstm.py

[8] Olah, Chris, Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/