Implementation of Densely Connected CNN with Multi-scale Feature Attention for Text Classification

Vishnu Chirukandath Ramesh 1921535

Abstract: Feature selection has a major role to play in the performance of a deep learning process, especially in the case of natural language processing we have to deal with meaning, emotion and, many other complicated characteristics. This work is the realisation of the mechanism to choose variable length features, which is crucial to significantly improve the performance. The normal n-gram will split the data into fixed sized features, in some scenarios it won't work well. sometimes we will get the opposite result also, especially in the case of sentimental analysis and other related fields. There are so many experiments going around the world in search of a good attention mechanism. The densely connected neural network has been a hot topic for a while and has also proven to be highly efficient. Therefore, this implementation makes use of the densely connected structure for the multi-scaled features attention. This work will be tested against the real and fake news data-set available in kaggle.

Keywords: Densely connected convolutional neural networks, Multi-scale feature attention, News classification

1 INTRODUCTION

Natural language processing, one of the major areas of application for deep learning algorithm have to deal with many cases including text classification. Deep learning model such as Recurrent Neural Networks (RNN), recursive autoencoders have been commonly used for NLP tasks, specifically Convolutional Neural Network [CNN] has proven to be very successful in computer vision and Text Classification [5, 2, 6, 12, 1].

Since, Most of the CNN Models make use of convolution filters of fixed size window which can only extract fixed-size n-gram [12, 1, 9] features as a drawback it does not extract variable size features such as, *phrases* for better representation[11]. The issue we may have to face is **the inability to adaptively select multi-scale features in a CNN model for text classification.** Multi-scale features refers ton-gram features with variable n, such as unigram ($n = 1$), bigram ($n = 2$), trigram ($n = 3$) and so on. For instance, Sentiment classification for sentence - *"He's nice to talk to without being patronizing"* (as shown in Figure1) requires an extraction of unigram feature *"nice"* and a trigram feature *"without being patronizing"*, which are both positive words or phrases. As shown in figure 1, applying size 1 filter results in one positive and one negative feature, meanwhile size 3 filter puts out an ideal (positive phrase) neural response for phrase *"without being patronizing"* indicating the unigram response *"nice"* is undesirably decreased as the model includes unnecessary information in either *"He's nice to"* or *"nice to talk"*.

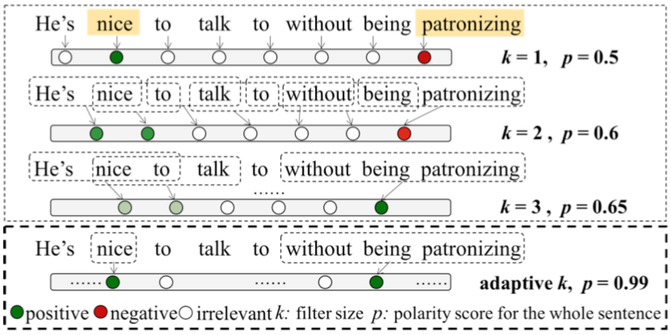
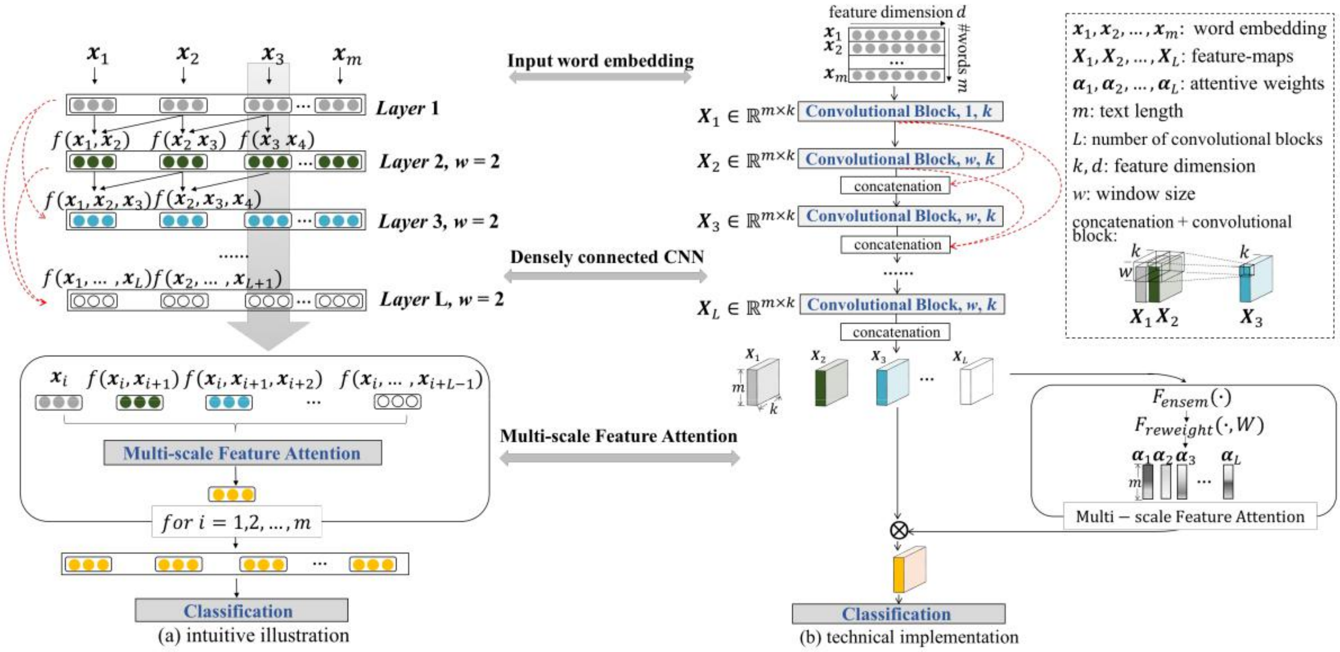

He's nice to talk to without being patronizing
k = 1, p = 0.5
k = 2, p = 0.6
k = 3, p = 0.65
adaptive k, p = 0.99
● positive ● negative ○ irrelevant k: filter size p: polarity score for the whole sentence

Figure 1: An example illustrating importance of multi scaled feature attention. [10]

An effective approach would be to apply multiple filters with variable window sizes to extract multi-scale features, but even with a large amount of experimental efforts to find optimal combinations of different filter sizes this approach will require learning of several separate, disconnected networks in parallel without exploiting the feature maps from different filter sizes. Therefore, Authors in study [10] believe in deeper model which can create an hierarchical representations by multi-level abstraction and eliminating the need to re-learn redundant feature maps. As shown in figure 2, the model can create trigram features in third layer (e.g.$f'(x_1, x_2, x_3)$). by stacking another layer of window size 2 followed by a layer with window size 1. With the dense connection it will be able to make use of the data of the layer above. To select the task friendly feature from all the possible multi scale features, we design attention module before the classification layer. In summary this will implement the Densely connected CNN with multi-scale feature attention for text classification[10] and test against the real and fake


x_1 x_2 x_3 x_m
Layer 1
$f(x_1,x_2)$ $f(x_2,x_3)$ $f(x_3,x_4)$
Layer 2, w = 2
$f(x_1,x_2,x_3)$ $f(x_2,x_3,x_4)$
Layer 3, w = 2
$f(x_1,\dots,x_L)$ $f(x_2,\dots,x_{L+1})$
Layer L, w = 2
x_i $f(x_i,x_{i+1})$ $f(x_i,x_{i+1},x_{i+2})$ $f(x_i,\dots,x_{i+L-1})$
Multi-scale Feature Attention
for i = 1,2,…,m
Classification
Input word embedding
Densely connected CNN
Multi-scale Feature Attention
feature dimension d
#words m
$X_1 \in \mathbb{R}^{m\times k}$ Convolutional Block, 1, k
$X_2 \in \mathbb{R}^{m\times k}$ Convolutional Block, w, k
concatenation
$X_3 \in \mathbb{R}^{m\times k}$ Convolutional Block, w, k
concatenation
$X_L \in \mathbb{R}^{m\times k}$ Convolutional Block, w, k
concatenation
$F_{ensem}(\cdot)$
$F_{reweight}(\cdot, W)$
$\alpha_1 \alpha_2 \alpha_3$ α_L
Multi – scale Feature Attention
Classification
$x_1, x_2, \dots, x_m$: word embedding
$X_1, X_2, \dots, X_L$: feature-maps
$\alpha_1, \alpha_2, \dots, \alpha_L$: attentive weights
m: text length
L: number of convolutional blocks
k, d: feature dimension
w: window size
concatenation + convolutional block:
$X_1 X_2$
X_3

(a) intuitive illustration

(b) technical implementation

Figure 2: This picture indicate the the Densely connected CNN with multi scale feature attention. The figure (a) Indicates how the model generate the multi scale features and how the features are used for classification. (b) Technical implementation all the convolution and concatenation operation, dense connection and multi scale feature attention.

news data-set.

2 RELATED WORK

Several noticeable advancement can be noticed in the models for the natural language processing. The Recurrent neural network, including Long short term memory and gathered recurrent unit stated to use widely in the text processing. several other variants are also proposed like [7]. The CNN is another most widely used model, the convolution filters on sliding window for next sequence and applied a max operation to capture most useful feature. [5] adopted multiple filters with different window sizes to extract multi- scale convolutional features for text classification. A dynamic k-max pooling mechanism was proposed in [2]. aother proposal is a novel feature mapping operator to generate non consecutive n-gram features [6].

Several advancement can be seen in the attention mechanisms also, if we equip the neural network models like CNN and RNN, its very likely to get good performance. Among the proposed attention mechanisms, main noticeable are Soft and hard attention [4] and local and global attention like that list will progress

3 METHOD

The model we are making use is the one mentioned in [10]. The figure 2 we can find more detailed technical view on the right, and the one on the right it emphasis on the overall view. The inputs to the model begins with text input $x_0x_1x_2x_3$ and generates multi scale features by convolution blocks with dense connections. The convolution block on the top will generate small gram and the one on downstream ones for large n-grams. The upstream blocks, downstream blocks gets the flexibility with the dense connection, as the the downstream blocks have access to all the upstream blocks. With the attention mechanism, the feature maps will re-weighted with filter ensemble and scale re weight.

$$X = [x_0, x_1, x_2, ..., x_n]_{m\times d}$$

In this case m is the number of words in the text. The output of each layer can be can be be represented as $X_l = [x_{l1}, x_{l2}, x_{l3}, ..., x_{lm}]_{m\times k}$. Where l is the layer index it can be $l \leq L$, L is number of blocks convolution blocks and k is the dimension of the transformed feature representation.

Each convolution block X_l which is the function composition of three cascaded operations: a convolution, a back normalization and rectified linear unit ($X_l = f(W_l, X_{l-1})$)[8]. Each X_l depends on the previous feature X_{l-1} and the learn able weights W_l, its weight consist of k filters with size $w \times k$. Normally in traditional Convolution block use to form a sequential hierarchy, but in this model we follow a dense connection[3]. With the dense connection, for a layer l can make use of all inputs and outputs of upstream layers.

$$X_l = f(W_l, [X_1.X_2, ..., X_{l-1}])$$

Even though the effective use of these feature is really a key issue.In order to make use of these features for classification the multi-scale feature attention mechanism is useful which will adaptive selects feature of different size [10]. This mechanism contains two operators: **filter ensemble** and **scale reweight**.

Filter ensemble aims to develop scalar descriptors s_l^i to represent feature at each scale. This will be later make use of this description during scalar reweigh.

$$s_l^i = F_{ensem}(x_l^i) = \sum_{j=1}^{k} x_l^i(j)$$

scale reweight, it will make use of the descriptor developed with filter ensemble as the input to generate attention weights to re weigh the features. So the final representation

$$x_{atten}^i = \sum_{l=1}^{L} \alpha_l^i x_l^i$$

$$\sum_{l=1}^{L} \alpha_l^i = 1, 1 \leq i \leq m[10]$$

The attention module will pass the re weighted feature map into the classification layer, which has the training objective to minimize the cross entropy loss:

$$\phi = \mathcal{L}(y, h(W, X_{atten}))$$

where $\mathcal{L}(y, h(W, X_{atten}))$ is the predicted output distribution, and y is the referenced distribution.

4 EXPERIMENTS

4.1 Implementation details

We have evaluated the model mentioned in the [10] with the real and fake news dataset. The dataset is available in kaggle. So our data set will have 2 classes one the news is valid and the news is invalid. The dataset will be available like in diffenet file we should combine them and process and shuffle. We have used the Glove word embedding was of size 300. Input text is padded into a fixed size as mentioned like in the above table. we used 100 in our case. We used tensorflow V2 for implementing this model.In [10] the model was implemented with caffe. Now the caffe 2 is a part of Pytorch. we used 5 convultional blocks, window size $w = 3$ and the feature size 128. The classification have fully connected MLP with ReLu activation function and softmax output. We used Nadam optimizer with low learning rate.

4.2 Main Results

The model was trained successfully with an accuracy more than 95 percentage and loss around 0.1 on validation set in 50 epoch. The output graph can be seen in the figures 3 4. When we analyse we can see model started with initial accuracy around 50 percentage and loss about 0.9. and then it started to under fit and again aligned around epoch 34. after 30 the model started to over fit, the validation accuracy started to increase. So the best modal would be around epoch 34.

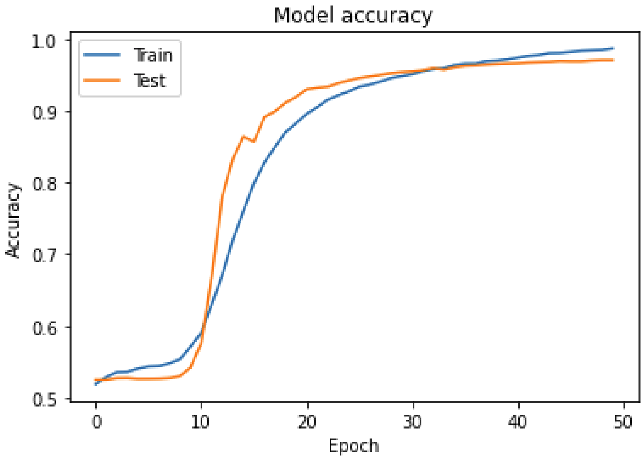


Figure 3: Model accuracy for the training with real and fake news dataset.

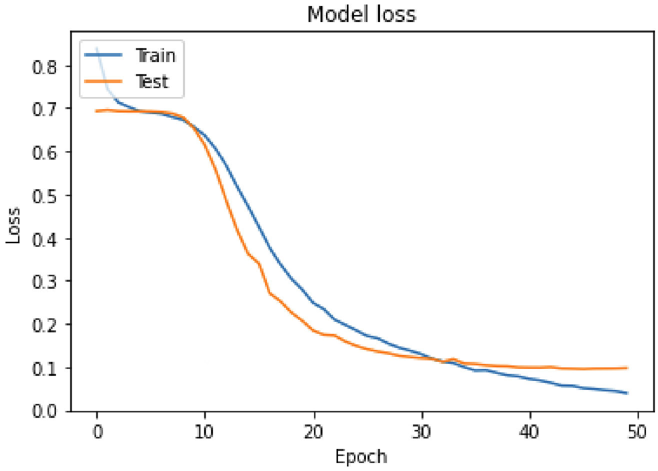


Figure 4: Model loss for the training with real and fake news dataset.

4.3 Hyper parameter tuning

The accuracy and loss is observed with the change of the parameters like learning rate, window size and schedule decay. There was significant change in the performance when changing these parameters. when the window size is reduced to 1 and 2 the performance decreased compared to what we have with window size 3. in the case if learning rate the model was finding local mini-ma at the beginning so the lower learning rate chosen. schedule decay is also we chooses a medium range value to maintain good performance.

5 CONCLUSION

The Densely Connected CNN with Multi-scale Feature Attention for Text Classification is a remarkable work. Even though the feature attention mechanism stated to show its significance a while ago with the densely connection and functions made them more power full and more accurate. The data set showed its significance in 6 benchmark data set includes agnews, yelp etc. It gave a significant performance for the data set the we choose which includes data about real and fake news. The model have several layers, it was implemented with caffe by the authors, which was a real power full framework, and it combined with pytorch. In current implementation Tensorflow was the choice, the lack group convolution feature made to think about pytorch but the concatenation process was little complicated in that, so proceeded with tensor flow by implementing group convolution by tensors. So it increased the number of layers in our model. Overall this model is perfect fit for most of the natural language processing applications.

References

[1] Loıc Barrault Alexis Conneau, Holger Schwenk and Yann Lecun. Very deep convolutional networks for natural language processing... *In EACL*, 2017.

[2] Phil Blunsom, Edward Grefenstette, and Nal Kalchbrenner. A convolutional neural network for modelling sentences. *In ACL*, 2014.

[3] Laurens van der Maaten Gao Huang, Zhuang Liu and Kilian Q Weinberger. Densely con- nected convolutional networks. *In CVPR*, 2017.

[4] Ryan Kiros Kyunghyun Cho Aaron Courville Ruslan Salakhudinov Rich Zemel Kelvin Xu, Jimmy Ba and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *In ICML*, 2015.

[5] Yoon Kim. Convolutional neural networks for sentence classification. *In ENMLP*, pages 1746–1751, 2014.

[6] Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. Molding cnns for text: non-linear, nonconsecutive convolutions. pages 1565–1575, 2015.

[7] Qiao Qian Minlie Huang and Xi aoyan Zhu. Encoding syntactic knowledge in neural net- works for sentiment classification. *In ACM Transactions on Information Systems (TOIS)*, 2017.

[8] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann ma- chines. *In ICML*, 2010.

[9] Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. Combining knowledge with deep convolutional neural networks for short text classification. *In IJCAI*, page 2915–2921, 2017.

[10] Shiyao Wang1, Minlie Huang, and Zhidong Deng. Densely Connected CNN with Multi-scale Feature Attention for Text Classification. *In IJCAI*, page 4468–4474, 2018.

[11] Tianyang Zhang, Minlie Huang, , and Li Zhao. Learning structured representation for text classification via reinforcement learning. *In AAAI*, 2018.

[12] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification.. *In NIPS*, pages 649–657, 2015.