Combining Image and Language to Predict and Understand the Usefulness of Yelp Reviews

David Z Liu
dzliu@stanford.edu

Abstract

In this paper, I investigate the effectiveness of combining image and language to predict and gain insight into the usefulness of Yelp Reviews. Predicting a written review's usefulness is an important and challenging task. Knowing the usefulness of a review in advance, businesses can recommend high quality and fresh reviews to their customers and gain insights into their products and services. I first associate each review with an image base on the review content, and then convert the image and the review text into two feature vectors using a Convolution Neural Network (CNN) and a Recurrent Neural Network (RNN) respectively; the weighted combination of the two feature vectors is used to make a usefulness prediction. To enhance the RNN's performance, I implemented a Convolution-RNN (C-RNN) structure that applies convolution layers on top of RNN's attention signal. I cast the usefulness prediction problem into regression and classification tasks, the final prediction result using the combined model significantly outperformed the SVM and linear regression baseline.

1. Introduction

The quality of written reviews varies significantly; high quality reviews will accumulate a large number of useful votes from the community over time. The freshness of a review is an important quality feature, a model that correctly estimates the number of useful votes a written review will receive in the future can add important commercial value to businesses.

It is particularly challenging to predict the usefulness of a review due to the scarcity of labeled data, only less than 10% of the reviews in the released yelp dataset have significant indication of usefulness (more than three useful votes). Furthermore, unlike sentiment analysis, where the positive and negative sentiment words play an important role in the classification task, there is no obvious features that directly indicate the usefulness of a document. The inherent difficulty of usefulness prediction is clearly demonstrated in the two example reviews listed below. Both reviews are written for the same business around the same time and both give five-star ratings. Even though they share similar sentiment and language constructs, the first review received 70 useful votes while the second review received 0 useful vote:

```
We decided to go here for the $32 summer
special.  We went early to have a glass
of wine and an appetizer before dinner.
It was very comfortable and enjoyable
sitting and listening to the live music.
The food and service was amazing both
upstairs and downstairs.  We would highly
recommend the entire experience.
What a great find!!

Votes: {funny:0, useful:70, cool:70}


I love this place!  The food is excellent;
the atmosphere is fabulous! Went there for
a girls night out and we were treated to a
wait staff of all very handsome men!  They
were courtesy and attentive and that, along
with the decor, music and food made it a
terrific night out!

Votes: {funny:0, useful:0, cool:0}
```

Recent development of RNN and CNN architectures have achieved the state-of-the-art performance on a number of important natural language processing (NLP) and computer vision tasks, ranging from language modeling [1] to speech recognition [2] to image classification [3] to machine translation [4, 5]. In this paper, I combine RNN and CNN model to predict the usefulness of a review. I associate each review with one image base on the text overlap between the review content and the image caption. The review and image is consumed by a Convolution-RNN (C-RNN) and a CNN respectively, the output of the two network is combined to make the final prediction.

Two types of objective functions can be employed to predict a reviews usefulness: regression and classification.

The regression model directly predicts the number of useful votes a review receives; it uses root mean square error (RMSE) as the loss function. The classification model buckets reviews into two classes base on the number of useful votes, the model predicts the probability a review belongs to each bucket; this model uses cross entropy as the loss function.

I first train a bidirectional RNN models without using image or attention mechanism, this RNN's performance serves as another baseline in addition to the SVM and linear regression baseline. I then add attention mechanism and convolution layers to this RNN to create the C-RNN architecture. Lastly, I construct a CNN to process the image associated with the review. The output of the C-RNN and CNN is combined to make the final prediction. The attention weights of the RNN indicate the amount of attention the model paid to each word, which can be used to visualize the influence each word has on the final prediction. My results show the final CNN + C-RNN model significantly outperform the bidirectional RNN, SVM and linear regression baselines.

2. Related Work

The RNN is an extremely expressive model that learns highly complex relationships from an arbitrarily long sequence of data. The RNN maintains a vector of activation units for each element in the data sequence, this makes RNN very deep. The depth of RNN leads to two well-known issues, the exploding and the vanishing gradient problems [7, 8]. The exploding gradient problem is commonly solved by enforcing a hard constraint over the norm of the gradient [9]; the vanishing gradient problem is typically addressed by Gated Recurrent Unit (GRU) or Long Short-Term Memory (LSTM) activation architectures [10,11,12]. Both the LSTM and the GRU solve the vanishing gradient problem by re-parameterizing the RNN.

Another issue with the basic RNN implementation is the output of each layer depends solely on the previous context. The meaning of words or sentences typically depend on the surrounding context in both directions, capturing only previous context lead to less accurate results. An elegant solution to this problem is provided by bidirectional RNN, where each training sequence is presented forward and backwards to two separate recurrent nets, both of which are connected to the same output layer. [13, 14, 15].

A recent advancement in RNN architecture is the incorporation of attention mechanisms. Attention mechanisms enable RNN model to pay different amount of attention to different part of input sequences. RNN models with attention mechanisms have achieved state-of-art performance in a number of NLP and computer vision tasks. [16, 17, 18, 19].

The CNN architecture has produced state-of-the-art results in recent image classification challenges [20, 21]. The standard CNN structure stacks multiple convolutional and pooling layers together and followed by one or more fully connected layers [22]. As the number of convolutional layer increases, the network become increasingly difficult to train, recent development of the residual learning framework [3] ease the training of deep networks by explicitly reformulate the layers as learning residual functions with reference to the layer inputs.

CNN and RNN have been combined in many innovative ways to address challenges in the cross section of computer vision and NLP. Architectures that sequentially cascade CNN and RNN have achieved remarkable results in image captioning [23], recent attempt to combine RNN and CNN into a unified architecture have reached the state-of-the-art results in multi-label image classification [25]. Moreover, CNN has been successfully used to model sentences [24].

3. Data

I use the dataset publicly available from the Yelp Dataset Challenge website[1]. The dataset includes JSON formatted objects containing businesses, users, and review data. The business object holds information such as business description, location, category, rating, and name etc. The review object contains star rating, review text, user information and usefulness voting etc. The yelp corpus contains roughly 2.7million reviews for 86K businesses written by 687K different users. In addition to the text data, this year, for the first time, Yelp released 200,000 pictures from 85,901 businesses described in the main dataset.

3.1. Review Selection and Label Generation

For classification task I have divided the data into two classes:

- Class 1: Un-useful reviews (reviews with 0 useful votes)
- Class 2: Useful reviews (reviews with >9 useful votes)

The review distribution is heavily skewed. More than 90% of the reviews have less than three useful votes; the review length distribution ranges from 2 words to 5000 word, almost half of the reviews fall into the 200-600 words range; a quarter of the reviews contains 250-350 words. I chose a subset of reviews that contains 300-350 words. To reduce the complexity of the RNN implementation I ensured all input reviews contain 300 words by stripping out additional words. I converted each word into 300-dimensional GloVe

[1] https://www.yelp.com/dataset_challenge

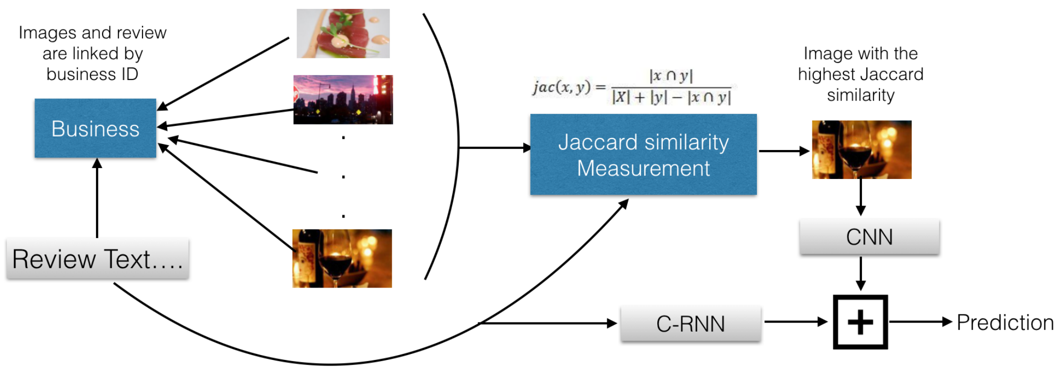
Images and review are linked by business ID
Business
Review Text….
$jac(x,y) = \frac{|x \cap y|}{|X| + |y| - |x \cap y|}$
Jaccard similarity Measurement
Image with the highest Jaccard similarity
CNN
C-RNN
Prediction

Figure 1. Image and review pairing flow.

word vectors[2]. When selecting the data, I made sure class populations are balanced.

Given the limitation of the computing infrastructure, I'm only able to use 15.5K reviews. I did not include any reviews written in 2015 and 2016 because it takes time for a high-quality review to accumulate a large number of useful votes. I divided the data into 80% training, 10% validation and 10% test set.

3.2. Image Review Pairing

The released image data is exclusively associated with businesses, there is no directly linkage between a written review and images. The reviews and images are indirectly linked by business ID; given a review, we know the business it is written for, and for that business we have access to images taken for it. In addition to raw image pixels, each image also contains a small caption that offers a short description of the image. The way I chose to associate an image to a review is by measure the Jaccard similarity (unigram match) between the review text and the image caption (I removed stop words and applied stemming and lemmatization). More specifically, given a review for a business, I measure the Jaccard similarly between the review's text and captions of all images taken for that business. I associate the image with the highest Jaccard similarity to the review; ties are broken arbitrarily. I excluded reviews that have 0 Jaccard similarity with images.

This way of linking image to review is very crude, the caption of the image is very short and often do not aptly describe the image content. However, given the way data is presented, this might be the only way to associate image to review. Enforcing the requirement that every review must have one associated image, the data set shrinks from 15.5K to roughly 10K. Figure 1 shows the process of generating image and review pairs.

[2]Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors): glove.42B.300d.zip from http://nlp.stanford.edu/projects/glove/

4. Model

4.1. Bidirectional RNN

I begin by evaluating the performance of a bidirectional RNN using only the review text without attention mechanism. A bidirectional RNN consists of a forward and a backward RNN structure. In the forward RNN, the input sequence is arranged from the first input to the last, the model computes a sequence of forward hidden states. The backward RNN takes the input sequence in reverse order, resulting in a sequence of backward hidden states. I concatenate output from both RNNs to make the final prediction. (figure 2)

4.2. RNN with Attention

The bidirectional RNN model must propagate dependencies over long distances to make the final prediction. The last layer of the network must capture all information from previous states, this is a challenging task for long sequential input. The attention-based model is an extension of the bidirectional RNN structure that overcomes this bottleneck of information flow; the hidden state of each forward and backward hidden layer is concatenated into a single output vector, this concatenated vector is transformed into a scalar value via a set of attention weight vectors. The resulting scalar value from each hidden state is concatenated into a new vector, this vector goes through an additional projection layer to generate the final prediction. (figure 3).

4.3. Convolution RNN (C-RNN)

To further exploit the attention output from the RNN, I feed the attention output into a multi-layer CNN and use its output to make the final prediction. The size of the attention output is RNN-hidden-unit-size $\times$ number of words in the review, in this case it is 300 $\times$ 300 $\times$ 1; I'm treating the attention output as a 2D image (figure 4). The output of the last layer of the C-RNN goes through a fully connected

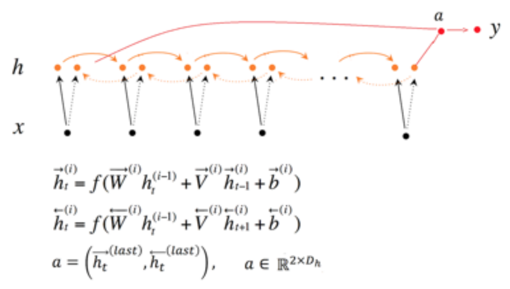

Figure 2. Bidirectional RNN

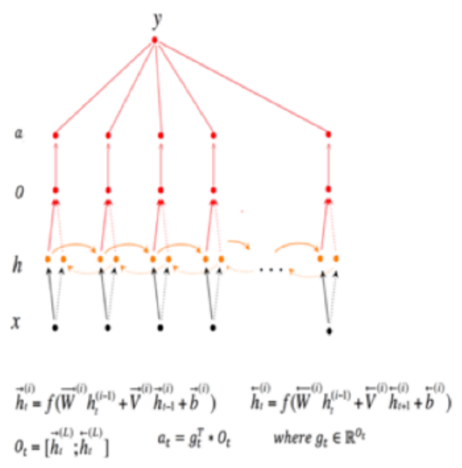

Figure 3. RNN With Attention

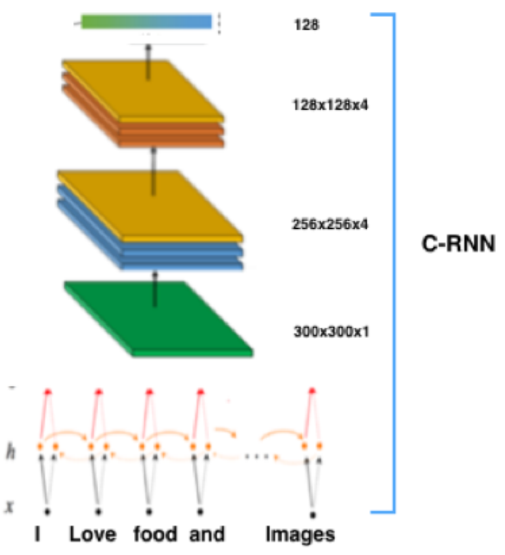

Figure 4. Convolutional RNN

layer and then through a softmax/regression layer to make the final prediction.

The intuition behind this approach is following: by scanning convolutional filters across the entire attention output, the model can better utilize the relationships between strong signals that are spatially separated. Moreover, the hieratical structure of the CNN can potentially capture latten themes/style in the writing.

The size of the convolutional layers is empirically determined through trial and error. I observed that scaling down the layer size too aggressively lead to poor performance, this is likely because small details in language are much more relevant as compare to image; for example, the meaning of ”good” and ”not good” are completely different.

The final architecture has two convolution layers and one fully connected layer. The first layer convolves 4 filters of 45 $\times$ 45 with stride 1 and applies a rectifier nonlinearity. The second layer convolves 4 filters of 4 $\times$ 4 with stride 2 and padding 1, again followed by a rectifier nonlinearity. This is followed by a fully connected layer of dimension 512 and a final layer that projects the output to a 128-dimensional vector. I used batch normalization between each layer. I did not use any pooling.

4.4. C-RNN + CNN

The final model uses both the image and language to make predictions. I constructed a CNN to convert the image into a 128-dimensional feature vector, I used the C-RNN described in 4.3 to convert the review text into a 128-dimensional feature vector. I weighted each vector by model parameters β and γ respectively, before summing them. A standard softmax/regression layer operates on the final feature vector to make the prediction. The motivation behind the usage of model parameter β and γ is to allow the model to decide the importance of each feature vector during training. If the image feature does not add value to the prediction task, the model can choose to zero it out.

In the CNN architecture, I converted all images to 64 $\times$ 64 $\times$ 3. The first layer convolves 8 filters of 2 $\times$ 2 with stride 2 and applies a rectifier nonlinearity. The second layer convolves 16 filters of 2 $\times$ 2 with stride 2, again followed by a rectifier nonlinearity. This is followed by a third convolutional layer that convolves 64 filters of 2 $\times$ 2 with stride 2 followed by a rectifier. The fourth layer convolves 64 filters of 2 $\times$ 2 with stride 2 followed rectifier units, this output goes through a fully connected layer with size 512, a final projection layer produces the 128-dimensional output vector. I used batch normalization between each layer. I experimented with a few architectural variations, this architecture is empirically the best. (see figure 5)

4.5. Hyper-Parameters

I used the following hyper-meters, the hyper-parameters are empirically determined:

- Number of hidden units:	300

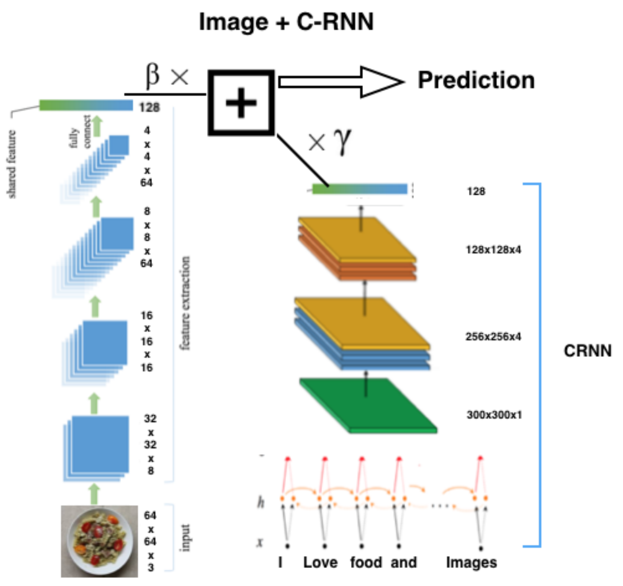

Figure 5. CNN + C-RNN

- Type of hidden units:	LSTM
- Dropout rate:	0.82
- Min-batch size:	32
- Learning-rate(regression):	0.0001
- Learning-rate(classification):	0.0007

5. Experiments and Results

5.1. Base Line

I constructed a support vector machine (SVM) and a linear regression model [3]to carry out the classification and regression tasks respectively. I converted the review corpus into a document-word matrix, where rows correspond to reviews in the corpus and columns correspond to words in the corpus. Each entry in the matrix corresponds to the number of occurrence of a word in a document. I normalized this matrix (each row has mean zero and standard deviation one) and used it as the input to the SVM and linear regression.

5.2. C-RNN Layer Sensitivity Analysis

The structure of C-RNN is new and unique; the convolution operation is applied to input that is quite different from the typical image pixels. It is meaningful to conduct a sensitivity study on the model accuracy versus number of the layers. Figure 6 and 7 show the model performance versus number of layers. Base on the results, we can conclude that increasing the number of layers does not always help the performance; this is different from the behavior of applying CNN on image. One reason for this behavior could be that the meanings in language is not formed in a very hierarchical way. I use two convolution layers in C-RNN throughout the rest of the experiments.

[3]I used the SVM and Linear Regression implementation from sklearn library (python); Specifically, the SVC implementation of SVM, which internally is based on libsvm. The Kernel is RBF and penalty parameter is set to 1.0. I use default values provided by the library for all the optional parameters: degree=3, gamma=0.0, coef0=0.0, shrinking=True, probability=False, tol=1e-3, cache_size=200, class_weight=None, verbose=False, max_iter=-1, random_state=None

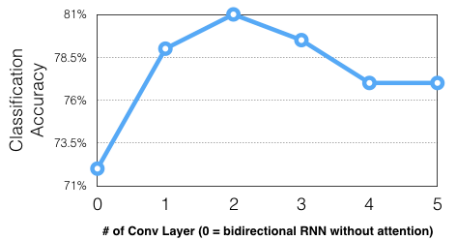

Figure 6. Number of convolution layer V.S. classification accuracy

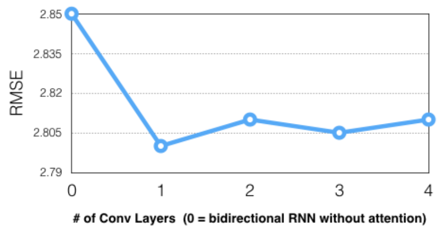

Figure 7. Number of convolution layer V.S. regression accuracy

5.3. All Model Performance

Figure 8 and 9 show the performance of all model variants for regression and classification task. The result indicates neural network architectures significantly outperform the SVM and linear regression baseline, and adding attention mechanism significantly improves the model performance in both regression and classification task.

In the classification setting, using C-RNN boosted the performance slightly, and using CNN + C-RNN further improved the performance by a very small amount; it seems like more complex models does help classification performance. The situation is different in the regression setting, it seems like increasing the model complexity did not improve the model performance in a noticeable way; since predicting the exact number of votes a review will receive is a much more difficult task than learning a binary decision boundary, this result is not too surprising.

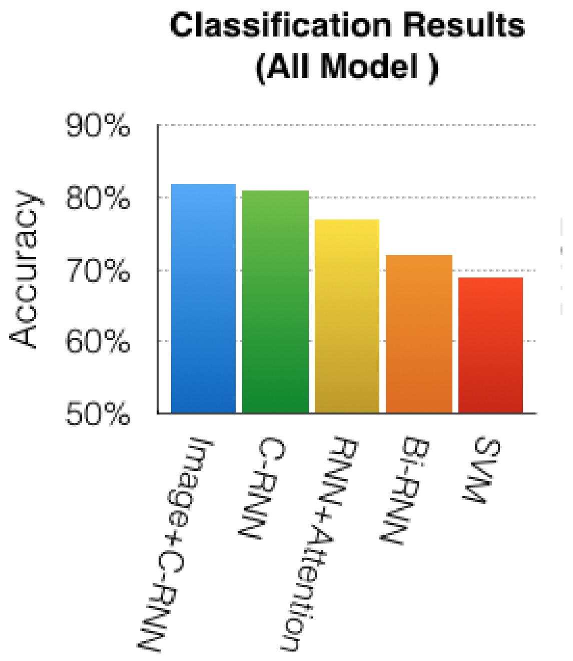

Figure 8. Classification performance for all models

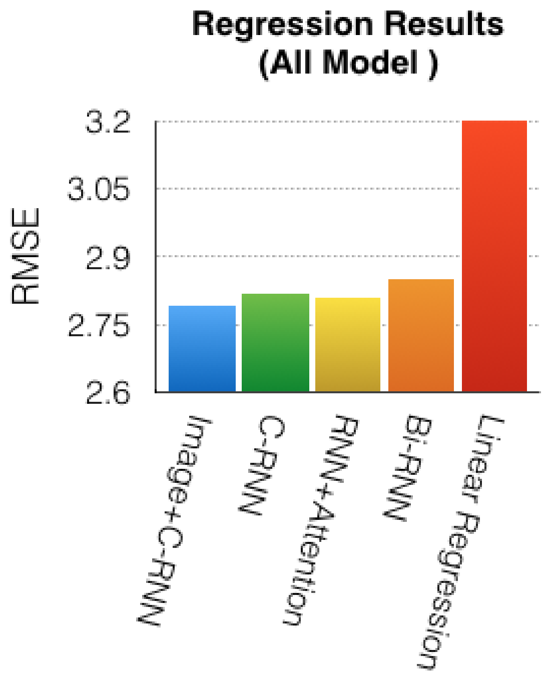

Figure 9. Regression performance for all models

6. Attention Visualization

The attention based RNN model (figure 3) transform the output of each hidden state into a scalar value via a set of

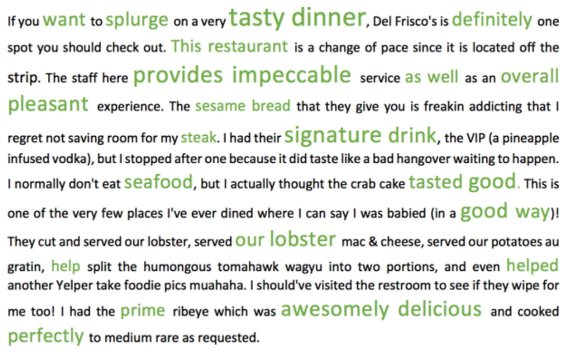
If you want to splurge on a very tasty dinner, Del Frisco's is definitely one spot you should check out. This restaurant is a change of pace since it is located off the strip. The staff here provides impeccable service as well as an overall pleasant experience. The sesame bread that they give you is freakin addicting that I regret not saving room for my steak. I had their signature drink, the VIP (a pineapple infused vodka), but I stopped after one because it did taste like a bad hangover waiting to happen. I normally don't eat seafood, but I actually thought the crab cake tasted good. This is one of the very few places I've ever dined where I can say I was babied (in a good way)! They cut and served our lobster, served our lobster mac & cheese, served our potatoes au gratin, help split the humongous tomahawk wagyu into two portions, and even helped another Yelper take foodie pics muahaha. I should've visited the restroom to see if they wipe for me too! I had the prime ribeye which was awesomely delicious and cooked perfectly to medium rare as requested.

Figure 10. An example of attention weight distribution of a correctly classified useful review

attention weights, each scalar is then used to generate the final prediction. We can interpret the scalar value produced from each hidden state as the attention paid by the model to each input word. Figure 10 shows a correctly classified useful review with top 25 most weighted words colored in green, their size is proportional to their attention weight. We observed that the model paid a large amount of attention to expressive and meaningful words. In addition, words that describe food and service tend to draw more attention whereas descriptions of personal feelings are down weighted. I believe further study of attention weights can reveal more insight into the underline dynamics of useful reviews.

7. Conclusion and Future Work

This work has two main contributions. First, it demonstrated that applying convolution operation on RNN's attention signal can improve the performance of classification tasks. Secondly, it showed a feasible way to combine image and language to make classification and regression prediction. I believe the performance of the model can improve greatly if we have direct association between images and review.

For future work, it would be interesting to experiment with different CNN and C-RNN architectures. It would be worth well to explore the possibility of using multiple images. I showed that the attention weight reveals some level of the underline dynamics of a useful review, it would be interesting to further investigate the relationship between attention weight and review content.

Acknowledgments

I would like to thank Gurbir Singh for his generous help to process the Yelp data. I would also like to thank Professor Chris Manning for his helpful suggestions

8. Reference

[1] Mikolov, Tomas, et al. ”Recurrent neural network based language model.”Interspeech. Vol. 2. 2010.

[2] Graves, Alex, Mohamed, Abdel-rahman, and Hinton, Geoffrey. Speech recognition with deep recurrent neural networks. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, pp. 66456649. IEEE, 2013.

[3] He, Kaiming, et al. ”Deep residual learning for image recognition.”Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[4] Luong, Minh-Thang, et al. ”Multi-task sequence to sequence learning.”arXiv preprint arXiv:1511.06114(2015).

[5] Dong, Daxiang, Wu, Hua, He, Wei, Yu, Dianhai, and Wang, Haifeng. Multi-task learning for multiple language translation. In ACL, 2015.

[6] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. ”Glove: Global Vectors for Word Representation.”EMNLP. Vol. 14. 2014.

[7] Bengio, Yoshua, Simard, Patrice, Frasconi, Paolo, 1994. Learning long-term dependencies with gradient descent is difficult. Neural Networks, IEEE Transactions on, 5, pp.157166.

[8] Jozefowicz, Rafal, Zaremba, Wojciech, and Sutskever, Ilya. An empirical exploration of recurrent network architectures. In Proceedings of the 32nd International Conference on Machine Learning (ICML-15), pp. 2342 2350, 2015.

[9] Pascanu, Razvan, Mikolov, Tomas, and Bengio, Yoshua. On the difficulty of training recurrent neural networks. arXiv preprint arXiv:1211.5063, 2012.

[10] Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 17351780.

[11] Gers, F., Schraudolph, N., Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. Journal of Machine Learning Research, 3, 115143.

[12] Cho, Kyunghyun, van Merrienboer, Bart, Gulcehre, Caglar, Bougares, Fethi, Schwenk, Holger,and Bengio, Yoshua. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.

[13] Schuster, M., Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45, 26732681.

[14] A. Graves and J. Schmidhuber, Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures, Neural Networks, vol. 18, nos. 5-6, pp. 602-610, 2005.

[15] Baldi, P., Brunak, S., Frasconi, P., Soda, G., Pollastri, G. (1999). Exploiting the past and the future in protein secondary structure prediction. BIOINF: Bioinformatics , 15.

[16] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. ”Neural machine translation by jointly learning to align and translate.”arXiv preprint arXiv:1409.0473(2014).

[17] Mnih, Volodymyr, Nicolas Heess, and Alex Graves. ”Recurrent models of visual attention.”Advances in Neural Information Processing Systems. 2014.

[18] Karol Gregor, Ivo Danihelka, Alex Graves, and Daan Wierstra. DRAW: A recurrent neural network for image generation. CoRR, abs/1502.04623, 2015.

[19] Hermann, Karl Moritz, et al. ”Teaching machines to read and comprehend.”Advances in Neural Information Processing Systems. 2015.

[20] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. ”Imagenet classification with deep convolutional neural networks.”Advances in neural information processing systems. 2012.

[21] Zeiler, Matthew D., and Rob Fergus. ”Visualizing and understanding convolutional networks.”European conference on computer vision. Springer International Publishing, 2014.

[22] LeCun, Yann, et al. ”Backpropagation applied to handwritten zip code recognition.”Neural computation1.4 (1989): 541-551.

[23] Xu, Kelvin, et al. ”Show, attend and tell: Neural image caption generation with visual attention.”International Conference on Machine Learning. 2015.

[24] Blunsom, Phil, Edward Grefenstette, and Nal Kalchbrenner. ”A convolutional neural network for modelling sentences.”Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, 2014.

[25] Wang, Jiang, et al. ”Cnn-rnn: A unified framework for multi-label image classification.”Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.