Understanding and Predicting the Usefulness of Yelp Reviews

David Zhan Liu (*dzliu@stanford.edu*)
Project Mentor: Chris Manning

Abstract

In this paper, I investigate the effectiveness of using Recurrent Neural Network (RNN) 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 build three variations of RNN architectures ranging from bidirectional RNN to multi-stack RNN with attention mechanism to RNN with residual connections. I cast the usefulness prediction problem into regression and classification tasks and use RNN models to predict them individually as well as jointly. I observed that the attention mechanism is crucial to achieve good classification performance, and joint-task learning improves the regression prediction noticeably. The best RNN models significantly outperform the support vector machine (SVM) and linear regression baseline. Analysis of the attention weights associated with each word in the text corpus reveals insight into the underlying factors that make a review useful.

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. 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.

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 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 Recurrent Neural Network (RNN) architectures have achieved state-of-the-art performance on a number of important natural language processing (NLP) tasks ranging from language modeling [29] to speech recognition [38] to machine translation [1, 2]. In this paper, I develop a variety of RNN models to predict the usefulness of a review. The input to the RNN is the GloVe word vector representation of the review text [6]. GloVe representation greatly increase the representativeness and expressiveness of words, they have been used in many important NLP works [7, 8, 9]. Two types of objective functions can be employed to predict a review's 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 and predicts the probability a review belonging to each bucket; this model uses cross entropy as the loss function.

As an initial step, I train a bidirectional RNN model without attention mechanism, I then add attention mechanism to the bidirectional RNN, next I increase the depth of the model by stacking multiple RNN layers together, and lastly, I add residual connection from the input to the prediction layer of the stacked RNN model. In each model, I train on both regression and classification task individually as well as jointly. The attention mechanism assigns an attention weight to each hidden state and uses the combination of the hidden state's outputs to make the final prediction. The attention weights indicate the amount of attention the model paid to each word, which can be used to visualize the influence of each word. My results show that RNN models significantly outperform the SVM and linear regression baselines, the jointly trained model yields the best regression prediction, and the four-layer attention based bidirectional RNN with residual connection produces the best classification prediction. The analysis of the top 26 words ranked by attention weight reveals that there are five major categories (number/price, opinion, time, food name, and service) that make a review useful.

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 [10, 11]. The exploding gradient problem is commonly solved by enforcing a hard constraint over the norm of the gradient [12]; the vanishing gradient problem is typically addressed by Gated Recurrent Unit (GRU) or Long Short-Term Memory (LSTM) activation architectures [13,14,15]. Both the LSTM and the GRU solve the vanishing gradient problem by re-parameterizing the RNN; The input to the LSTM cell is multiplied by the activation of an input gate, and the previous values are multiplied by a forget gate, the network only interacts with the LSTM unit via gates. GRU simplifies the LSTM architecture by combining the forget and input gates into an update gate and merging the cell state with the hidden state.

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. [17, 18, 19]

Recent experiment on multiple stack RNN architecture has shown remarkable success in NLP tasks [32]. Multi-stacked architecture operates on different time scales; the lower level layer captures short term interaction while the aggregate effects are captured by the high level layers. [33] Another 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 natural language and image processing tasks. [34, 35, 36, 37]. Deeper neural networks are more difficult to train, recent development of the residual learning framework [40] ease the training of deep networks by explicitly reformulate the layers as learning residual functions with reference to the layer inputs; this approach have achieved the state-of-the-art performance in image classification.

Multi-task learning (MTL) is an important machine learning paradigm that uses multiple related objectives to improve the generalization performance of a main objective. MTL is a well-studied frame work [21,22,23,24,25], in the context of deep learning, it has been applied successfully to various problems ranging from computer vision to speech recognition to drug discovery [27, 28, 31]. Recent works have shown that MTL can improve the performance of attention-free sequence to sequence model and other language related tasks. [1, 2, 3, 4, 26]

3 Data

I use the dataset publicly available from the Yelp Dataset Challenge website[1]. The dataset includes five 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.

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 $250 - 280$ words. To reduce the complexity of the RNN implementation I ensured all input reviews contain 250 words by stripping out additional words. I converted each word into 300-dimensional GloVe 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. Each model takes a few hours to train on a single GPU.

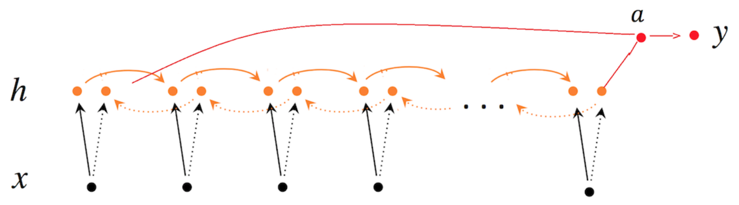
4 Model

4.1 Model Development

I begin by evaluating the performance of a bidirectional RNN without using 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 1)

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

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



$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

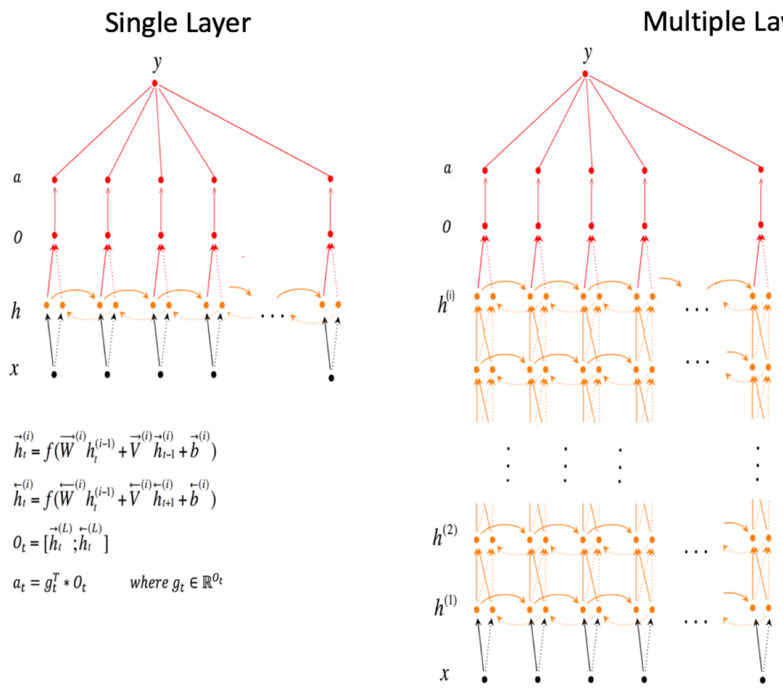
$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$a = \left(\overrightarrow{h_t}^{(last)}, \overleftarrow{h_t}^{(last)}\right), \qquad a \in \mathbb{R}^{2\times D_h}$$

Figure 1: Bidirectional RNN architecture

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. To overcome this bottleneck of information flow I implemented an attention mechanism inspired by recent results in natural language and image processing tasks. [34, 35, 36, 37]

The attention-based model is an extension of the bidirectional RNN structure, 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. The multi-layer RNN model is constructed by stacking multiple RNN models on top of each other, the hidden states of the lower layer RNN model serve as input to the next layer. The hidden states of the last layer are used to make the final prediction via the attention mechanism (figure 2). The number of stacks in the model is a hyper-parameter.



$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

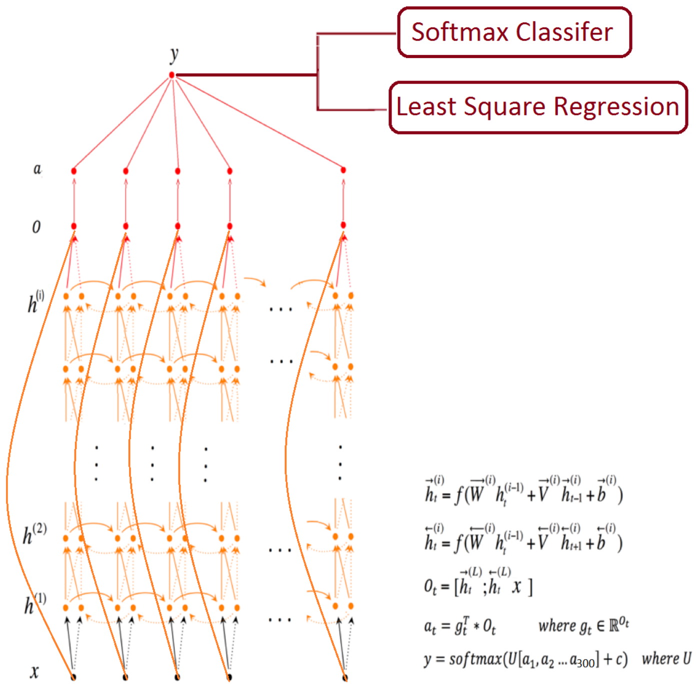
$$0_t = [\overrightarrow{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}]$$

$$a_t = g_t^T * 0_t \qquad where\ g_t \in \mathbb{R}^{0_t}$$

Figure 2: Single Layer and Multiple Layer RNN model using attention mechanism

As the depth of the stacked RNN increases, the input word vector's influence to the prediction layer decreases, however, the input word vector carries relevant and direct information regarding to the review content. Inspired by the recent success in the residual network architecture [40], I added residual connections to the stacked RNN that explicitly feed the input word vector to the prediction layer. (figure 3)

In each model, I trained classification and regression objectives separately as well as jointly. For joint objective training, I adopted the alternating training approach described in [1, 2], where each task is optimized for a fixed number of parameter updates before switching to the next task. A key parameter that impacts the model performance is the frequency of alternation. I assign a task as the main prediction task, this task will drive the majority of parameter updates during training; the secondary task only updates the parameter once in a while, its main purpose is to share training information with the main task and help regularize it. I empirically observed that running one epoch of secondary task in every 7 epochs of main task produce good results. For instance, if the main task is classification and the secondary task is regression, then for every 7 epochs of classification training, I train the model for one epoch of regression.



$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$0_t = [\overrightarrow{h}_t^{(L)} ; \overleftarrow{h}_t^{(L)} x]$$

$$a_t = g_t^T * 0_t \qquad where\ g_t \in \mathbb{R}^{0_t}$$

$$y = softmax(U[a_1, a_2 \dots a_{300}] + c) \quad where\ U \in \mathbb{R}^{2\times300}$$

Figure 3: RNN structure with residual connection

4.2 Hyper-parameter Selection

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

-	Number of hidden units:	300
-	Type of hidden units:	LSTM
-	Word Vector Size:	300
-	Dropout rate:	0.9
-	Min-batch size:	16
-	Learning rate:	0.0001(regression) 0.001(classification)
-	Multi-task alternation frequency:	7 epochs

5 Experiments and Results

5.1 Base Line

I constructed a support vector machine (SVM) and a linear regression model [2] 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 RNN Results

I examined the performance variation as the number of RNN stack changes, the results are shown in figure 4. The classification performance of the model without residual connection peaks at stack two, while the model with residual connection peaks at the fourth stack and it achieves a small performance improvement. This result confirms the assumption that the residual connection enables the model to achieve good performance in deeper layers. The experimental result indicates that stacking RNNs together does not noticeably improve the regression performance. For both prediction tasks, adding attention mechanism significantly boosted the prediction accuracy. Joint-task learning is performed using the most accurate model observed, the accuracy of the regression model improved after applying joint-task learning, and the classification accuracy did not improve. Figure 5 shows the results of the best performing model in each architecture variation.

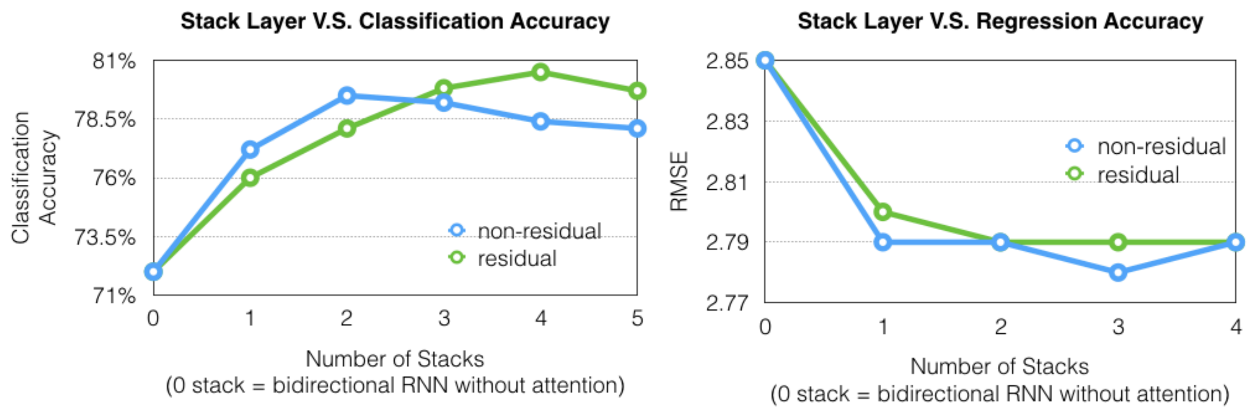


Figure 4: Number of RNN Stacks V.S. Prediction Performance

[2] 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

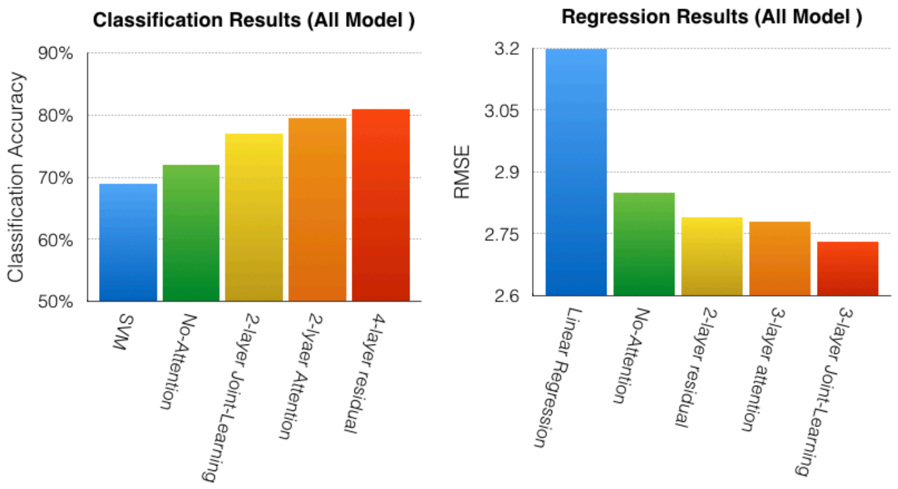


Figure 5: Best Performing Models

5.2 Attention Visualization

Attention mechanism transform the output of each hidden state into a scalar value via a set of 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 weight assigned by the model to each input word. I aggregated the attention weight for each word in the review corpus and normalized it by its frequency of concurrence (equation 1). Figure 6 shows the top 26 words ranked by attention weight; their size is proportional to their weight and they divide roughly into five categories, “numbers and price”, “data and time”, “opinion”, “food name”, and “services”. This result indicates reviews that describe food, service, opinion, exact numbers and time tend to draw more attention by the model during usefulness prediction.

$$weight\ of\ word_i = \frac{\sum_{word_i \in (+Review)} weight_{word_i} + \sum_{word_i \in (-\ Review)} -1 * weight_{word_i}}{frequency_{\{word_i\}} * \left|\log\left(\frac{frequency_{\{word_i\}}}{frequency\ of\ all\ words}\right)\right| + 1}$$

Equation 1: Aggregation of attention weights.

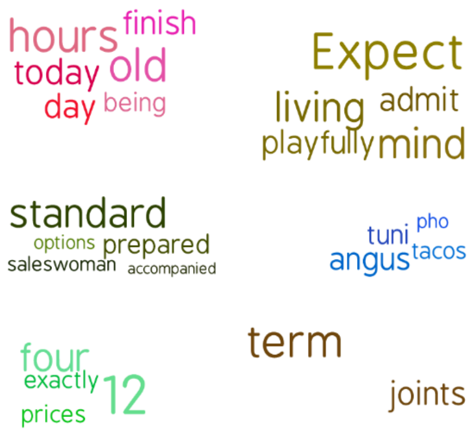


Figure 6: Word Cloud of The Top 26 Words by Attention Weight

6 Conclusion and Future Work

In this paper, I implemented three variations of RNN models to predict the usefulness of yelp reviews. The experimental results indicate the three-layer attention based bidirectional RNN jointly trained on both regression and classification tasks achieves the best regression accuracy, and the four-layer RNN with residual connection attains the best classification accuracy. The experimental results also showed that the attention weights reveal the underline dynamics of a useful review. In the future, I would like to extend the model to handle variable review length and fine tune all the hyper-parameter to further improve the model's performance. Given that the attention mechanism significantly increased the prediction accuracy, it would be interesting to further exploit the attention signals by feeding them into a convolution neural network before making the final prediction.

Reference:

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

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

[3] Vinyals, Oriol, Kaiser, Lukasz, Koo, Terry, Petrov, Slav, Sutskever, Ilya, and Hinton, Geoffrey. Grammar as a foreign language. In NIPS, 2015a.

[4]Vinyals, Oriol, Toshev, Alexander, Bengio, Samy, and Erhan, Dumitru. Show and tell: A neural image caption generator. In CVPR, 2015b.

[5] Dai, Andrew M. and Le, Quoc V. Semi-supervised sequence learning. In NIPS, 2015

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

[7] You, Quanzeng, et al. "Image captioning with semantic attention." arXiv preprint arXiv:1603.03925 (2016).

[8] Bowman, Samuel R., et al. "A fast unified model for parsing and sentence understanding." arXiv preprint arXiv:1603.06021 (2016).

[9] Bowman, Samuel R., Christopher Potts, and Christopher D. Manning. "Learning distributed word representations for natural logic reasoning." arXiv preprint arXiv:1410.4176 (2014).

[10] Bengio, Yoshua, Simard, Patrice, Frasconi, Paolo, 1994. Learning long-term dependencies with gradient descent is difficult. Neural Networks, IEEE Transactions on, 5, pp.157–166.

[11] 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.

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

[13] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.

[14] Gers, F., Schraudolph, N., & Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. Journal of Machine Learning Research, 3, 115–143.

[15] 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.

[16] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." Journal of Machine Learning Research 15.1 (2014): 1929-1958.

[17] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45, 2673–2681.

[18] 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.

[19] 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.

[20] Kingma, D. P., & Ba, J. L. (2015). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations, 1–13

[21] Thrun, Sebastian. Is learning the n-th thing any easier than learning the first? in NIPS, 1996

[22] Caruana, Rich. Multitask learning. Machine Learning, 28(1):41-75, 1997

[23] Evgeniou, Theodoros and Pontil, Massimilinao. Regularized multi-task learning

[24] Argyriou, Andreas, Evgeniou,Theodoros, and Pontil, Massimiliano. Multi-task feature learning. In NIPS, 2007

[25] Kumar, Abhishek and III, Hal Daume. Learning task grouping and overlap in multi-task learning. In ICML, 2012

[26] Liu, Xiaodong, Gao, Jianfeng, He, Xiaodong, Deng, Li, Huh, Kevin, and Wang, Ye-Yi. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In NACCL, 2015

[27] Donahue, Jeff, Jia, Yangqing, Vinyals, Oriol, Hoffman, Judy, Zhang, Ning, Tzeng, Eric, and Darrell, Trevor. DeCAF: A deep convolutional activation feature for generic visual recognition, 2014

[28] Heigold, Georg, Vanhoucke, Vincent Senior, Alan, Nguyen, Patrick, Ranzato, Marc’ Aurelio, Devin, Matthieu, and Dean, Jeffrey. Multilingual acoustic models using distributed deep neural networks. In ICASSP, 2013

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

[30] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems. 2014.

[31] Ramsundar, Bharath, et al. "Massively multitask networks for drug discovery." arXiv preprint arXiv:1502.02072 (2015).

[32] Irsoy, Ozan, and Claire Cardie. "Opinion Mining with Deep Recurrent Neural Networks." EMNLP. 2014.

[33] Michiel Hermans and Benjamin Schrauwen. 2013. Training and analyzing deep recurrent neural networks. In Advances in Neural Information Processing Systems, pages 190–198

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

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

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

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

[38] 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. 6645–6649. IEEE, 2013.

[39] Raskutti, Garvesh, Martin J. Wainwright, and Bin Yu. "Early stopping and non-parametric regression: an optimal data-dependent stopping rule." Journal of Machine Learning Research 15.1 (2014): 335-366.

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