A Recurrent Neural Network Based Recommendation System

David Zhan Liu
Department of Computer Science
Stanford University
Stanford, CA 94305
dzliu@stanford.edu

Gurbir Singh
Department of Computer Science
Stanford University
Stanford, CA 94305
gsgill@stanford.edu

Abstract

Recommendation systems play an extremely important role in e-commerce; by recommending products that suit the taste of the consumers, e-commerce companies can generate large profits. The most commonly used recommender systems typically produce a list of recommendations through collaborative or content-based filtering; neither of those approaches take into account the content of the written reviews, which contain rich information about user's taste. In this paper, we evaluate the performance of ten different recurrent neural network (RNN) structure on the task of generating recommendations using written reviews. The RNN structures we study include well know implementations such as Multi-stacked bi-directional Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) as well as novel implementation of attention-based RNN structure. The attention-based structures are not only among the best models in terms of prediction accuracy, they also assign an attention weight to each word in the review; by plotting the attention weight of each word we gain additional insight into the underlying mechanisms involved in the prediction process. We develop and test the recommendation systems using the data provided by Yelp Data Challenge.

Introduction:

The rise in popularity of review aggregating websites such as Yelp and Trip-Advisor has led to an influx of data on people's preference and personality. The large repositories of user written reviews create opportunities for a new type of recommendation system that can leverage the rich content embedded in the written text. User preferences are deeply ingrained in the review texts, which has an amble amount of features that can be exploited by a neural network structure. In this paper, we conduct a comparative study of ten different recurrent neural network recommendation models.

A well-known issue with models that attempt to make prediction for a particular user base on that user's data is the inherent data sparsity. A typical user tends to generate only a small amount of data, despite the large overall size of the corpus. Many innovative methods have been invented to resolve the data sparsity issue [1][2][3]. Since our interest is to supply the model with adequate data in order to capture the user preferences, we decide to find the nearest neighbors for a given user base on their preferences and train the model using the reviews from all the users in the nearest neighbor cluster.

To create the input to our RNN models, we convert each word in the review text into distributed representation in the form of word vector; each word vector in the review document serves as input to a hidden layer of the RNN [4]. The output of the model is a prediction of the probability that a user will like the particular restaurant associated with the input review. Each cluster of users has its own model trained using the reviews in the corresponding cluster.

We employ a bottom-up approach to create different RNN structures. We begin by examine the performance of two RNN architectures (GRU and LSTM) that curb the vanishing gradient problem [7][8], next we enhance our models ability to capture contextual information by adding bi-directionality, lastly, we increase our model's interpretability of complex relationships by stacking multiple hidden layers. In addition to implementing known model structures, we also create a new attention-based RNN model that collects signals from each hidden layer of the RNN and combine them in innovative ways to generate prediction. The attention-based model addresses the issue of reliance on the last layer to capture information embedded in all previous layers; this model also assigns an attention measure to each word in the review, the attention measure indicates the amount of attention the model allocates to each word.

1 Related work

The RNN is an extremely expressive model that learns highly complex relationships from a sequence of data. The RNN maintains a vector of activation units for each time step in the sequence of data, this makes RNN extremely deep; the depth of RNN leads to two well known issues, the exploding and the vanish gradient problem [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 LSTM or GRU architectures [10][11][12]. Both the LSTM and the GRU solves the vanishing gradient problem by re-parameterizing the RNN; The input to the LSTM cell is multiplied by the activation of the input gate, and the previous values are multiplied by the forget gate, the network only interacts with the LSTM cell via gates. GRU simplifies the LSTM architecture by combing the forget and input gates into an update gate and merging the cell state with the hidden state. GRU has been shown to outperform LSTM on a suite of tasks. [8][13]

Another issue inherent in the uni-directional RNN implementation is the complete dependency of each layer's output on the previous context. The meaning of words or sentences typically depend on the surrounding context in both directions, capturing only the previous context leads to less accurate prediction. An elegant solution to this problem is provided by bi-directional recurrent neural networks (BiRNN), where each training sequence is presented forward and backward to two separate recurrent nets, both of which are connected to the same output layer. [14][15][16]

Recent implementation of multiple stack RNN architecture has shown remarkable success in natural language processing tasks [18]. Single layer RNNs are stacked together in such a way that each hidden state's output signal serves as the input to the hidden state in the layer above it. Multi-stacked architecture operates on different time scales; the lower level layer captures short-term interaction, while the aggregated effects are captured by the high level layers [17].

The latest development of incorporating attention mechanisms into RNN enables the RNN model to focus on aspects of a document that it believes to deserve the most amount of attention. The attention mechanism typically broadcast signals from each hidden layer of the RNN, and make prediction using the broadcast signal. Attention-based models have produced state of art results in a wide range of natural language and image processing tasks. [19][20][21][22]

In this paper we evaluate all model structures mentioned above on the task of generating recommendation based on review text. We also implement a novel attention-based model that has never been studied before.

2 Dataset

We used the dataset publicly available from the Yelp Dataset Challenge website.[1] The dataset provides five JSON formatted objects containing data about businesses, users, reviews, check-ins and tips. We only used data from business, user and review JSON objects. The business object holds information such as business type, location, category, rating, and name etc. The review object contains star rating and review text. The yelp corpus contains *2225134* reviews for *77445* businesses written by *552339* different users. We reduced the size of the corpus to *1231275* reviews from *27882* different eateries (cafes, restaurants and bars).

To overcome the inherent data sparsity in individual user data, we cluster users into groups base on their preferences using k-nearest neighbor method described in [2]. We focus our experiment on a cluster that contains eight prolific reviewers with *4800* reviews, we divide this review dataset into training-set (*4000* reviews), validation-set (*400* reviews) and test-set (*400* reviews). Each word in the review documents is converted into a 300 dimensional word vector representation using the pre-trained GloVe dataset [5].

In order to simplify the implementation of our RNN models, we normalize each review to 200 words; this is accomplished by stripping words that come after the 200th word in reviews with more than 200 words, and padding reviews with less than 200 words using repetition of the last sentence in the review. The number 200 is chosen base on statistics collected from the review corpus:

- 63% reviews ~ +/-25 from 200 words
- 7% reviews had less than 150 words and 13% has more than 250 words.
- Overall 80% of the reviews have between 150 to 250 words.

The above statistical observation indicates normalizing review text to length 200 should not significantly alter the information contained in most of the documents. The ideal approach is to build RNN models that can dynamically handle variable review length, in the interest of time, we decide to leave this implementation as part of future improvement.

3 Technical Approach and Models

3.1 General Approach

We implement ten different RNN models, each model takes reviews of a restaurant as input and classify the restaurant as favorable or unfavorable for a user.

We divide the restaurant reviews into the following two categories:

Favorable : reviews with 4 or 5 star ratings

Unfavorable : reviews with 1 or 2 star ratings

Each word vector in the review text is feed into a hidden layer of the RNN model; the final output goes through a soft-max function and returns a probability for each class label. We

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

used the cross-entropy loss as the cost function to train the model; the true class labels are represented as one-hot vector.

In practice we must develop a different model for each cluster, and generate a prediction that applies to all users in a cluster. To limit the scope of this comparative study, we only develop models for the cluster described in the data section.

3.2 Model Selection

We first compare the performance between GRU and LSTM on this specific prediction task, the result indicates the GRU structure performs slightly better than LSTM[1]. Using the GRU as the RNN cell, we implement single, double, triple, and quadruple stacked bi-directional model; the same implementation procedure is also employed to implement four stacked bi-directional attention-based structure. (Figure 1)

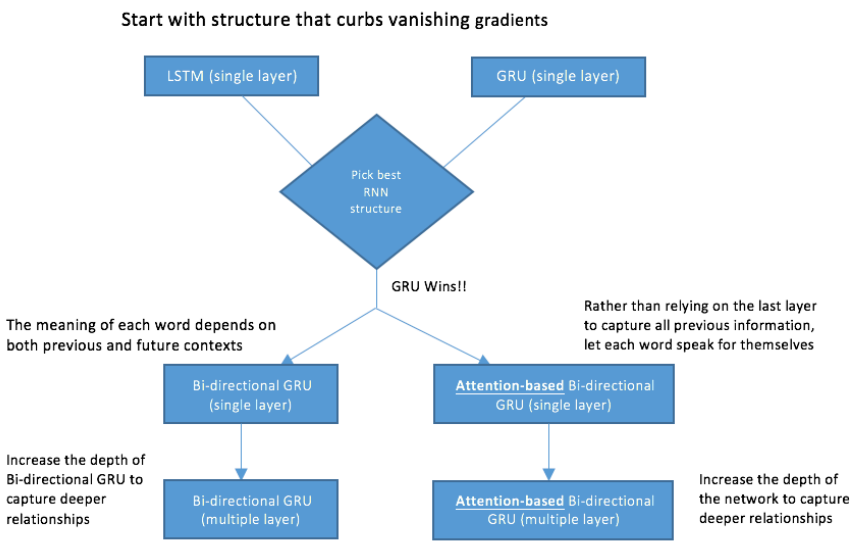

Figure 1: Model Selection Flow

3.3 Bi-directional RNN (BiRNN) Model Description

BiRNN consists of forward and backward RNN structure (GRU cell). In the forward RNN, the input sequence is arranged from the first word to the last word, and the model calculates a sequence of forward hidden states. The backward RNN takes the input sequence in reverse order, resulting in a sequence of backward hidden states. To compute the final prediction, we average the output from RNNs in both direction and then apply linear transformation to generate the input to the softmax prediction unit. (figure 2)

The multi-stack BiRNN is constructed by stacking single layer BiRNN on top of each other. The hidden state of each previous layer serves as input to the hidden state above it. Intuitively, every layer treats the memory sequence of the previous layer as the input sequence, and compute its own memory representation [18][22]. To compute the final prediction, we average the output from the last layer's RNNs in both direction and follow the same prediction scheme described above. (figure 2)

---

[1] We are using tanh as the activation function for all our experiments

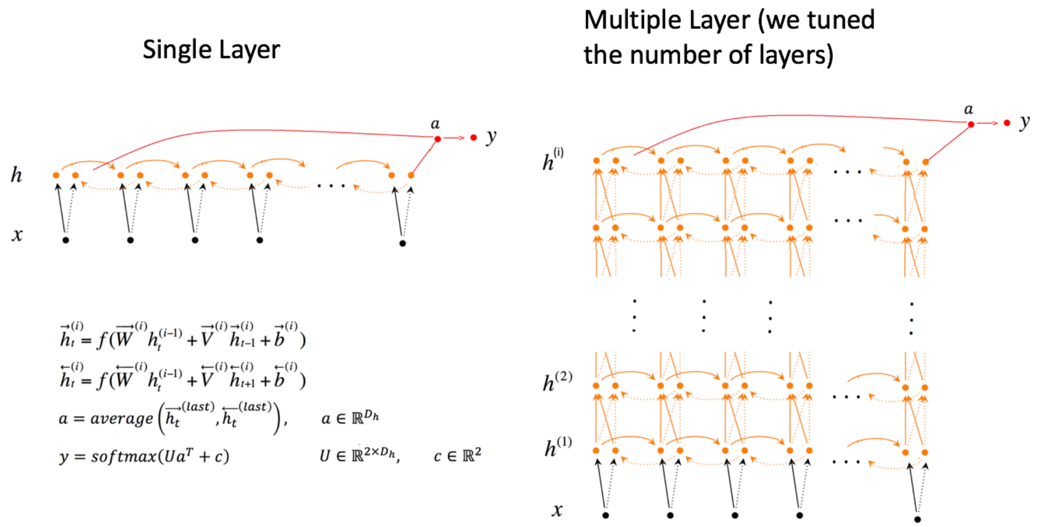

Figure 2: BiRNN with GRU Cell

3.4 Attention Mechanism Model Description

A standard RNN model must propagate dependencies over long distance in order to make the final prediction. The last layer of the network must capture all information from the previous states to make the prediction, this may make it difficult for the neural network to cope with long document size. In our case, we fix the review length to 200 words, which is quite long. To overcome this bottleneck of information flow we implement an attention mechanism inspired by recent results in natural langue and image processing tasks. [19][20][21][22]

The attention-based model utilizes the same base BiRNN structure described in section 2.3, the hidden state of each forward and backward GRU unit 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)

Intuitively, the attention-based BiRNN implements a mechanism of attention in the model. Attention weight vectors transform each hidden state into a scalar value that represents the amount of attention the model pays to the input word in the hidden state. Plotting the attention value of each word in the document reveals that the model tends to make correct predictions when it focuses more on the expressive words. (more discussion in the result section)

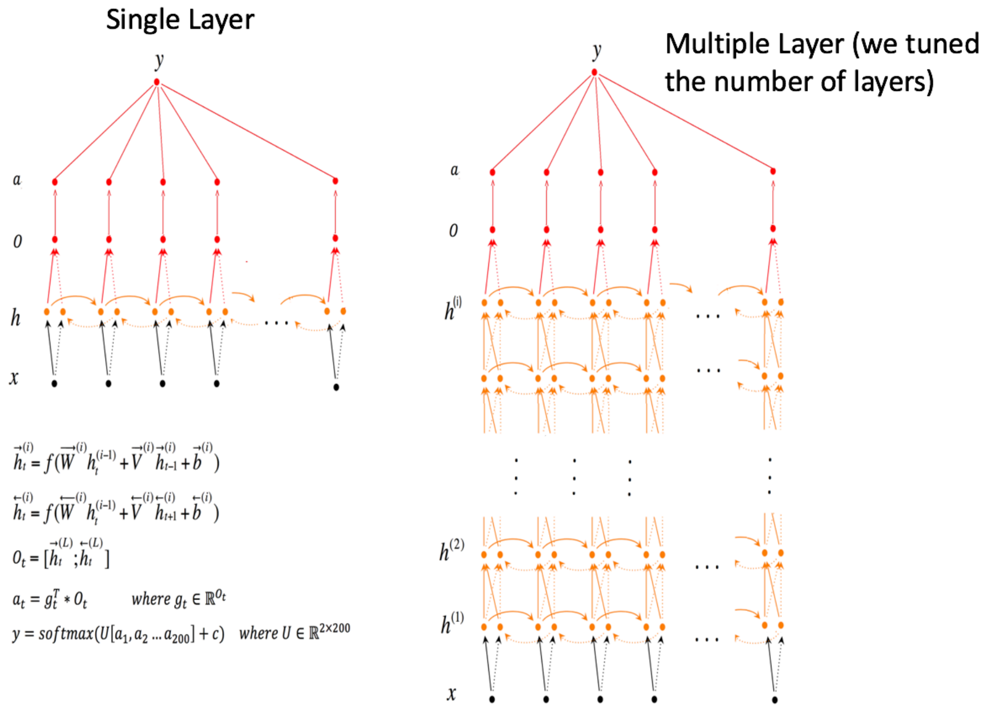

Figure 3: Attention Based BiRNN with GRU cell

4 Experiments & Results

4.1 Evaluation Metric and Hyper-Parameter Tuning

We use an off the shelf support vector machine (SVM) as the baseline for our model[1]. We collect a total of 4800 review documents from 8 users, each word in the review is converted into 300 dimensional vector representation using GloVe [5]. We use cross-validation to train each model; roughly 80% of the data is used as training set, 10% is used as validation set and the remaining 10% is used as test set. Mini-batch gradient descent (batch size 50) is used as the search algorithm. All hypermeters are tuned using the validation set. The final accuracy for each model is measured as the percentage of correct prediction on the test set.

For single layer, uni-directional LSTM and GRU we consider hidden activation unit size [64, **128**, 256], learning rate range [0.001, **0.005**, 0.0001, 0.0005], dropout range [**1**, 0.9, 08, 0.6]; in the case of LSTM we also consider forget bias range [0.1, 0.3, **0.5**, 0.8, 1]. For Bi-directional GRU we consider hidden layer size [64, **128**, 256], learning rate range [0.001, **0.005**, 0.0001, 0.0005], dropout range [**1**, 0.9, 08, 0.6]. For attention based bi-directional GRU we consider hidden layer size [64, **128**, 256], learning rate range [**0.001**, 0.005, 0.0001, 0.0005], dropout range [**1**, 0.9, 08, 0.6] (The bolded underline value represents the parameters selected). Adapting selected hyper-parameters, we measure the prediction accuracy for different level of stacks. (Table 2)

---

[1] We use SVM implementation from sklearn library (python). We use the SVC implementation of SVM, which internally is based on libsvm. The Kernel is *'rbf'* and penalty parameter is set to *1.0.* We use default values provided by the library for all the optional parameters like: 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

4.2 GRU V.S. LSTM

GRU and LSTM have similar performance, both of them performs slightly better than the SVM baseline. (Figure 4, table 1)

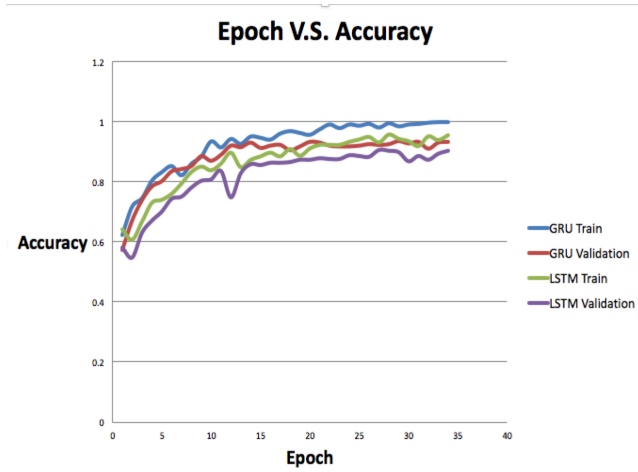

Figure 4: Epoch V.S. Accuracy for GRU and LSTM

	Train Accuracy	Validation Accuracy	Test Accuracy
SVM	87.25	82.00	76.25
GRU	99.60	93.25	82.75
LSTM	93.70	87.00	81.74

Table 1: GRU V.S. LSTM V.S. SVM

4.3 Multi-stack BiRNN V.S Attention-Based Multi-stack BiRNN

As expected, BiRNN out-performs uni-directional RNNs, and multi-stacked BiRNN out-performs the single stack BiRNN. We observe that the accuarcy does not always increase as we increase the number of stacks, this may due to the fact that aggeration of deeper meanings is optimally captured in certain depth. Attention-based model shows very similar accaracy measurement compared to BiRNN, especially in stack three; this is an indication that three stack structrue captures the best aggregate effect. To make the final prediction in the BiRNN setup, we are averaging the output of RNN from both directions, thus, the BiRNN model does not surfer the issue of reliance on a single layer to capture all previous information; this could be the reason for the slight better performance of the BiRNN model. (table 2)

	STACK 1	STACK 2	STACK 3	STACK 4
Bi-directional RNN	85.25	86.00	87.50	87.00
Bi-directional RNN with Attention	82.75	84.25	87.00	85.25

Table 2: BiRNN and BiRNN-attention test accuracy per stack

4.4 Paid Attention

The attention model transforms 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. The scalar value produced from each hidden state can be interpreted as the attention paid by the model to each input word in the hidden state. Fgure 5 shows a correctly classified review with the top 10 words ranked by attention-value colored in green, their size is proportional to their attention-value. We observe that the model paid large amount of attention to expressive and meaningful words. Figure 6 shows an incorrectly classified review with the top 10 words ranked by attention-value colored in green, their size is proportional to their attention-value. We observe that the model paid larger attention to inexpressive and meaningless words. This is a general trend we observe in all reviews studied, the attention model tends to make correct prediction when it pays large amount of attention to expressive words, and it tends to make incorrect prediction when it spends most of its attention on inexpressive words.

Figure 5

Figure 6

5 Conclusion and Future work

In this paper, we showed that neural network model is effective in predicting user perference base on their reviews, we also demonstrated that multi-stack bidirectional RNN model and attention-based RNN model produce more accurate prediction compared to single stack uni-directional RNN model. Our experimental data indicated that increasing number of stacks does not always imporve the model's performance. Our novel implementation of attention-based model produced attention demands for each word that provided additional insight into the classification problem. It would be interesting to conduct a close up study of attention demand for each word in the review corpus.

We believe the performce of the model can improve significantly using RNN implementation that can handle variable review length, additionally, the yelp review corpus for restaurants contains more than one million reviews, we used only a very small fraction of those, increasing our training data size will surly improve the prediciton accuarcy. Furthermore, it would be interesting to predict more than just two class labels, for instance we could expand the label class to like, neutral and unlike. Another idea that is worth pursuing is to create an ensemble of neural networks for this task, the prediction can be generated using a linear combination of the output from each model in the ensemble set.

References

[1] Aggarwal, C.C., Wolf, J.L., Wu, K., Yu, P.S.: Horting hates an egg: A new graph-theoretic approach to collaborative filtering. In: Proc. of the 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, KDD 1999, pp. 201-212. ACM, New York (1999)

[2] Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item based collaborative filtering recommendation algorithms. In: Proc. of the WWW Conf. (2001)

[3] Wang, F., Ma, S., Yang, L., Li, T.: Recommendation on item graphs. IN. Proc. of the Sixth Int. Conf. on Data Mining, ser. ICDM 2006, pp. 1119-1123. IEEE Computer Society, Washington, DC (2006)

[4] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. NIPS 2013.

[5] Pennington, Jeffrey, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014) 12.

[6] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 1631–1642, Stroudsburg, PA, October. Association for Computational Linguistics.

[7] 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.

[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), 1735–1780.

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

[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] Chung, Junyoung, Gulcehre, Caglar, Cho, KyungHyun, and Bengio, Yoshua. Empirical evaluation of gated re- current neural networks on sequence modeling. arXiv preprint arXiv:1412.3555, 2014.

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

[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] 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.

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

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

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

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

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

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