Using Recurrent Neural Network to Predict The Usefulness of Yelp Reviews

David Zhan Liu *dzliu@stanford.edu*

Gurbir Singh gsgill@stanford.edu

Abstract

Predicting a written review's usefulness is an important and challenging task. A good review will accumulate large number of useful votes from the community through time. Knowing the usefulness of a review in advance allows businesses to recommend high quality and fresh reviews to customers. The usefulness of a review is difficult to predict due to scarcity of labeled data and challenging feature selection, there is no obvious feature that directly indicates the usefulness of a review. In this paper, we investigate the possibility of applying multi-task learning on a bi-directional recurrent neural network (RNN) to address the challenges in review usefulness prediction. Simultaneously training a model on multiple related objectives can improve the generalization performance of a main objective and compensate for lack of training data. We develop a model that jointly trains on three related tasks: usefulness classification, usefulness regression, and review star rating classification. The main objective of the model is to improve the usefulness prediction accuracy.

1 Introduction

The quality of written reviews varies significantly; high quality reviews will accumulate 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 on yelp 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 can 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 received five-star rating. Even though they share similar sentiment and language constructs, the first review received 70 useful votes while the second one received 0 useful votes:

> *"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}*

In this paper, we develop a variety of RNN models that predict the usefulness of a review. Two types of objective

functions can be employed to predict a review's usefulness: linear regression and classification. As an initial step, we train two separate bidirectional RNN models on each task, then we develop a bidirectional RNN model that trains on both tasks jointly, lastly, we add one additional related tasks, star rating prediction. The input of the bidirectional RNN is the review text, the output of the last layer of the RNN is used to make different predictions. (see figure 2). Our results show that RNN models significantly outperform the SVM and linear regression baselines, the multi-task learning model helps with model regularization, and single task bidirectional RNN models with dropout regularization produce the best results.

2 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 element in the data sequence, this makes RNN very deep. Its ability to capture long range dependencies has been exploited in many natural language process tasks. [29, 30, 5].

The depth of RNN lead 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.

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, 30]. 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

We used 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.

We have divided our data into different types of label configurations:

Configuration 1, usefulness classification with RNN:

- Class 1: Un-useful reviews (reviews with 0 useful votes)
- Class 2: Useful reviews (reviews with >3 and <6 useful votes)
- Class 3: Useful reviews (reviews with >10 useful votes)

Configuration 2, usefulness regression with RNN:

For this task, we take the same set of reviews as configuration 1 but instead of bucket them into three different classes, we take the number of useful votes as the label.

As shown in figure 1, the review distribution is heavily skewed. More than 90% of the reviews have less than three useful votes, also 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. We chose a subset

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

of reviews that contains 300 $\pm$ 30 words. The choice of label configuration is empirically determined and the data is chosen such that there is equal distribution among all classes. We converted each word into 300-dimensional GloVe word vectors[2].

Given the limitation of our computing infrastructure, we are only able to use *12K* reviews. We did not include any reviews written in 2015 and 2016 because it takes time for a high-quality reviews to accumulate large number of useful votes. We divided our data into 80% training, 10% validation and 10% test set. Each model takes a few hours to run on a single GPU.

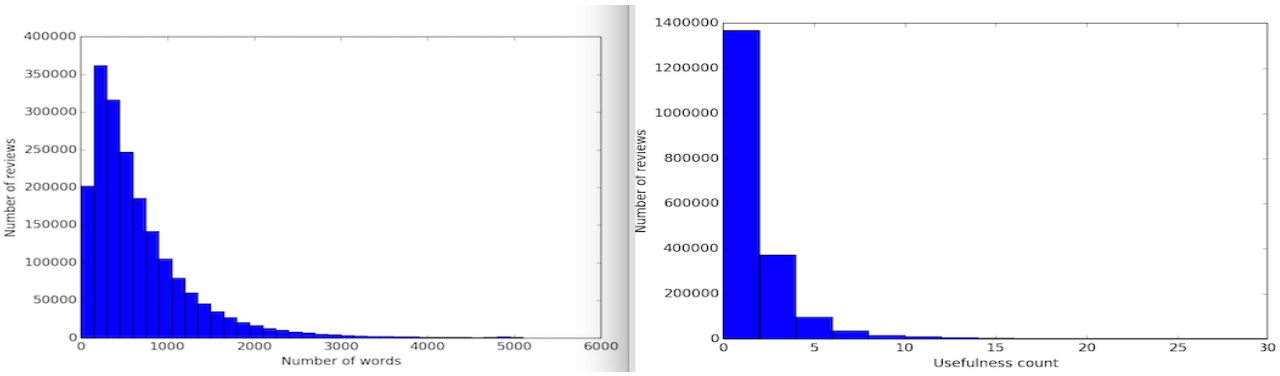

Figure 1: general statistics of the data

4 Technical Approach and Models

4.1 General Approach

We developed four different RNN models listed in table one. The input to the RNN is the GloVe word vectors; GloVe representations greatly increase the representativeness and expressiveness of words [6], they have led to significant improvements in a number of NLP tasks [7, 8, 9]. The output from the last layer of the RNN is used to make the final prediction (Figure 2).

We experimented with the two multi-task learning methods:

- Using a weighted sum of objective function as a single training objective
- Alternating objective functions during training, where we train on different tasks for different duration of time. The frequency of alternation is a hyper-parameter

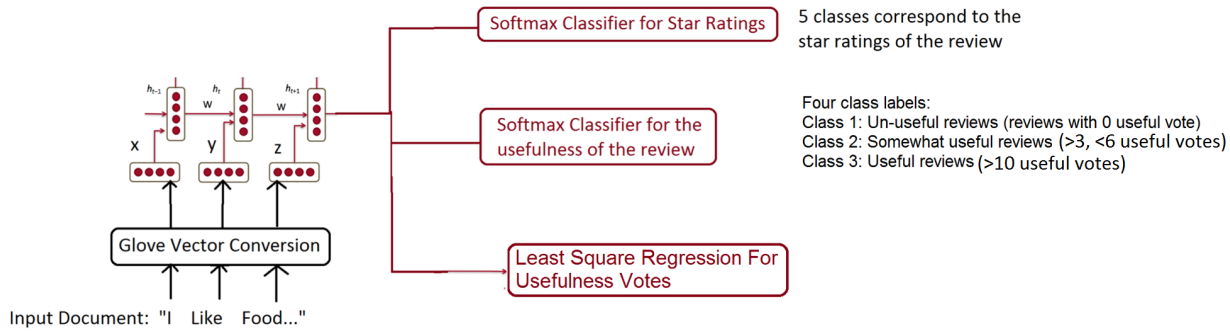

Figure 2: MTL RNN structure with detailed input and output description

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

Model	Input	Output	Loss Function
Single Task Bidirectional RNN	Review Text	Number of useful Votes	Mean Square Error Loss (EQ1)
Single Task Bidirectional RNN	Review Text	One hot vector encoding of usefulness class	Cross entropy loss (EQ2)
Two Tasks Bidirectional RNN	Review Text	- Useful Votes (for regression) - One hot encoding (for usefulness classification)	Mean Square Error Loss (for regression) Cross Entropy Loss (for classification)
Three Tasks Bidirectional RNN	Review Text	- Useful Votes (for regression) - One hot encoding (for usefulness classification) - One hot encoding (for star rating classification)	-Mean Square Error Loss (for regression) - Cross Entropy Loss for classification

Table 1: List of RNN Models

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_i - Y_i\right)^2 \qquad EQ1$$

Equation 1: *Mean Square Error, where* $\hat{Y}_i$ *is predicted value and* Y_i *is the true value*

$$J = -\frac{1}{n}\sum_{i=1}^{n}\hat{Y}_i \log Y_i \qquad EQ2$$

Equation 2: *Cross Entropy Loss, where* $\hat{Y}_i$ *is predicted probability and* Y_i *is the true probability*

4.2 Model Details

Choice of Activation Unit:

It is very challenging to successfully train an RNN with large number of layers due to vanishing and exploding gradient issues. We resolved the exploding gradient problem by clipping the gradient once the value surpasses a threshold; the vanishing gradient problem is addressed by using LSTM and GRU architectures as the activation unit. We evaluated the performance of GRU and LSTM activation unit on a unidirectional RNN, the LSTM architecture outperforms the GRU slightly, thus, we used LSTM in all our models.

Bi-directionality:

An issue inherent in the unidirectional 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 bidirectional RNN, where each training sequence is presented forward and backward to two separate recurrent nets, both of which are connected to the same output layer. [17, 18,19]

Bidirectional RNN consists of forward and backward RNN structure. 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 concatenate the output from RNNs in both direction and then apply

linear transformation to generate the input to the prediction unit.

Regularization

We observed significant overfitting in single task learning models, given the large number of parameters in the RNN, overfitting is expected. We found that the dropout rate of 0.15 is effective, we also experimented with early stop techniques [16], but it is not as effective. In the case of multi-task learning, we did not observe strong overfitting, this is because different tasks are regularizing each other.

Multi-task alternation frequency

Training different tasks in alternation yield better results compared to using a single weighted objective function, thus, we used the alternative training method in all multi-task learning models. A key parameter that impacts the model performance is the frequency of alternation, we empirically observed that running one batch of secondary task in every 7 epochs of main task produce good results. For instance, if the main task is usefulness classification and secondary tasks are usefulness regression and star classification, then for every 7 epochs of usefulness classification training, we train the model for one batch of usefulness and one batch of star classification.

Hyper Parameters:

We used the following hyper-meters for our models, the hyper-parameters are empirically determined:

- Number of hidden units: 128
- Type of hidden units: LSTM
- Word Vector Size: 300
- Dropout rate: 15%
- Min-batch size: 32
- Learning rate: 0.001
- Multi-task alternation frequency: 7 epochs

5 Experiments and Results

5.1 Base Line

We constructed a support vector machine (SVM) [2] and linear regression model to carry out the classification and regression tasks listed in table 1 respectively. We 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. We normalized this matrix (each row has mean zero and standard deviation one) and used it as the input to the SVM and linear regression. The results are shown in table 2.

5.1 RNN Results

The results of all RNN models are shown in table 2, RNN models significantly outperformed the baseline results. Binary classification is significantly accurate than three class classification, this is expected as the classes in binary classification have more definitive boundaries. It is worth noting that the baseline accuracy decreased significantly as number of classes increased, while the RNN's performance only decreased by a small amount. The multi-task learning model did not outperform single task learning model, however, we needed to

[2] We 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. We 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

apply a dropout rate of 15% on single task models while no dropout was needed for multi-task learning models, this indicates that each task in the multi-task learning model is regularizing other tasks. We believe with the correct task selection and proper parameter tuning, multi-task learning models can achieve better performance.

The classification performance on the three-task model (star rating classification, usefulness classification, and usefulness regression) is slightly better than the two-task model, while the regression performed worse. This is expected since star rating is a classification task, its prediction mechanism is related more to the useful classification tasks.

Task	RNN Result	Baseline
Single Task Model: Regression	**2.82**	3.27
Single Task Model: Binary Classification	**75%**	69.5%
Single Task Model: Three Class Classification	**71%**	57.62%
Two Tasks Model: Regression	2.99	3.27
Two Tasks Model: Three Class Classification	68.5%	57.62%
Three Tasks Model: Regression	3.1	3.27
Three Tasks Model: Three Class Classification	69.2%	57.62%

Table 2: Experiment results

6 Conclusion and Future Work

In this paper, we implemented a variety of different RNN models to predict the usefulness of yelp reviews. Our experimental results indicate that single task bidirectional RNN models achieve the best usefulness prediction accuracy among all models studied in this paper, and in general RNN models outperform the baseline models. We observed that multi-task learning can help regularize related tasks. Since we were limited in computing power, we did not have the opportunity to fine tune all the hyper-parameters, it would be interesting to see how different hyper-parameters affect the model. It is also meaningful to construct an ensemble of all the architectures studied in this paper, as it is well known that ensemble of related models tend to improve performance of the 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).