Learn To Rate Fine Food

Yixin Tang, Bian Lu, Jiacheng Mo

Motivation

In recent years, food reviews have become increasingly popular on social media. People are posting their reviews or comments on Facebook, Yelp, Twitter, etc. When selecting local restaurants or food, people also tend to make their decisions based on these reviews. Hence, it is important for both restaurants and individuals to quickly get the information and score of a food item or restaurant from thousands of reviews. It is also beneficial for some platform to provide different customers with their personal recommendations.

Problem Statement

Problem: Given a food review

- Predict its sentiment on a six point scale.
- Build the recommendation system.
- Automatically generate good and bad reviews.

Approach:

- Use pre-trained GloVe as our initializer of word vectors and then use skip-gram to train the word vectors for our own dataset. Then build a GRU to classify food reviews and use LSTM to generate reviews of different sentiment.
- Build a recommendation system using latent factor model.
- Use the traditional frequency method as word representation and logistic regression to get the corresponding results, and visualize the words leading to different scores by wordcloud.

Evaluation:

- For classification, use accuracy and F1 score to judge our result on test data. All neural network models are trained using cross entropy loss.
- For text generation, use intuition to see if the generated text make sense.
- For recommendation system, change learning rate in SGD and find the one that makes the smallest error after 40 iterations.

Dataset

- The data represents a set of food reviews on Amazon
- Number of instances: 568454

Product ID	User ID	Helpfulness	Score	Text
B001E4KFG0	A3SGXH7AUHU8GW	1	5	I have bought several of the …
B00813GRG4	A1D87F6ZCVE5NK	0	1	Product arrived Labeled as …
B000UA0QIQ	A395BORC6FGVXV	3	2	If you are looking for the secret …

Gated Recurrent Unit Network for Classification

$$Right\ \ direction\ \ \vec{h}_t^{(i)}:$$

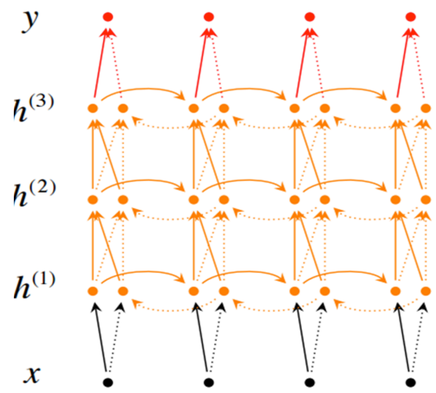
$$\vec{z}_t^{(i)} = \sigma(\vec{W}_{(i)}^{(z)} x_t^i + \vec{U}_{(i)}^{(r)} h_{t-1}^{(i)})$$
$$\vec{r}_t^{(i)} = \sigma(\vec{W}_{(i)}^{(r)} x_t^i + \vec{U}_{(i)}^{(r)} h_{t-1}^{(i)})$$
$$\tilde{\vec{h}}_t^{(i)} = tanh(\vec{W}_{(i)} x_t + r_t \circ \vec{U}_{(i)} h_{t-1})$$
$$\vec{h}_t^{(i)} = z_t^{(i)} \circ h_{t-1}^{(i)} + (1 - z_t^{(i)}) \circ \tilde{h}_t^{(i)}$$

$$Left\ \ direction\ \ \overleftarrow{h}_t^{(i)}:$$

$$\overleftarrow{z}_t^{(i)} = \sigma(\overleftarrow{W}_{(i)}^{(z)} x_t^i + \overleftarrow{U}_{(i)}^{(r)} h_{t-1}^{(i)})$$
$$\overleftarrow{r}_t^{(i)} = \sigma(\overleftarrow{W}_{(i)}^{(r)} x_t^i + \overleftarrow{U}_{(i)}^{(r)} h_{t-1}^{(i)})$$
$$\tilde{\overleftarrow{h}}_t^{(i)} = tanh(\overleftarrow{W}_{(i)} x_t + r_t \circ \overleftarrow{U}_{(i)} h_{t-1})$$
$$\overleftarrow{h}_t^{(i)} = z_t^{(i)} \circ h_{t-1}^{(i)} + (1 - z_t^{(i)}) \circ \tilde{h}_t^{(i)}$$

$$Output$$

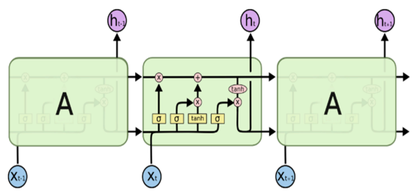
$$y_t = softmax(U[\vec{h}_t^{(top)}; \overleftarrow{h}_t^{(top)}] + c)$$



GRU Layers	Accuracy	Top 2 Accuracy (2 classes with the highest probability)
3 L	0.74	0.93
2 L	00.67	0.88

- Each x node represents a pre-trained GloVe review (text) vector on our dataset.
- After a three layers Bi-directional GRU, it outputs the predicted probability of each class.
- Finally, use cross entropy to update both weights and word embedding matrix.

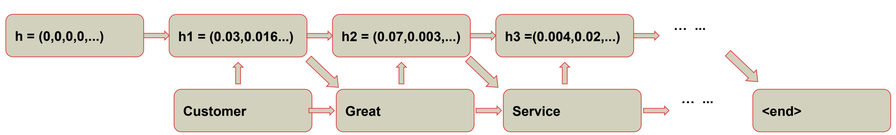
LSTM Network for Review Generation



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

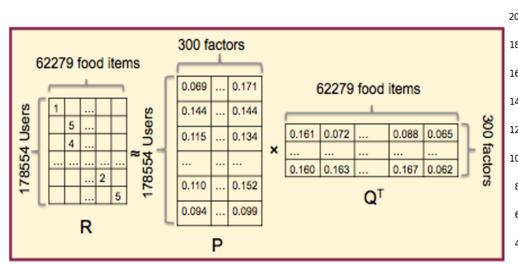
- Use Long Short Term Memory for text generation.
- Randomly initialize the first word x0, and pick the word with the highest probability in h0 to be x1, and then pick the word with the highest probability in h1 to be x2, etc.
- Consequently, with the change of training data, it will automatically generate best reviews, good reviews, or bad reviews.

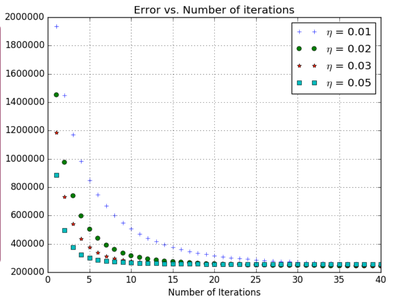
Example: Best Review (y=5) *"Customer Great Service Here !"*



Recommendation System

- Use Latent Factor Model to recommend food items to an user.
- Method:
  - Sparse user-item utility matrix is decomposed to an user-factor matrix and an item-factor matrix.
  - Use SGD to decrease the loss, and find the final user-factor matrix P and a item-factor matrix Q with relatively low error.
  - Predict what scores a user will rate an unrated item, and recommend the most highest 10 items to him/her.
- Evaluation: We tried 4 different learning rate in SGD, and finally use the 0.02 for SGD, with the smallest convergence error.
- Future improvement: adding the user/item bias, implicit feedback, temporal dynamics, user-associated attributes, and confidence level to be more accurate.





User ID	Product Highly rated	Product Recommended
A15ZCT30QMRCXY	Gerber cereal, single grain ... Gerber cereal, whole wheat ...	Gerber baby cereal with apple, Trident suger free gum, ...

Find Important Words & "Spam" Reviews

- Use logistic regression to find the most relevant words for a score.
- Method: Feature selection -> Logistic Regression -> Find greatest coefficient

- Use QDA and decision tree to model the "helpfulness" of a given review
- Vary regularization parameter to obtain optimum training and testing accuracy.