Yelp reviews classification and generation

Xinyi Gao, Ning Ju, Bowen Chen

***Abstract*—Text classification and generation has been a great challenge over years. In this project, we aimed to predict restaurant review score and generate the desired positive or negative text using Yelp dataset. We explored character level and word level CNN, LSTM, LSTM-CNN approaches and achieved preferred results. We also compared and showed the difference.**

I. Problem Statement

Yelp, a worldwide used reviews website containing over 5 millions reviews and over 174 thousands facilities, attracts millions of users to write reviews and choose restaurants, hotels, and all other kinds of services. Both business owners and customers are keen on the reviews. However, there are cases that reviews are gathered from other source including social media, questionnaires or other review sites. How to add these reviews to yelp as reliable rating source? Moreover, when the reviews are insufficient, some restaurants might like to generate good or bad reviews to attract more customers. Thus, generating realistic fake reviews are of high demand and also requires exploration on training such discriminator to filter fake reviews from the real ones.

In this project, we intend to classify an unrated reviews to five rating levels as in Yelp. Also, we generate the five star and one star reviews for further usage. Different approaches are utilized and compared to achieve the best results. We mainly explored character level CNN[1], word based LSTM, word based CNN-RNN for text classification. For text generation, word and character level LSTM[2] are implemented and compared.

II. Dataset

The dataset we used is from Yelp dataset challenge[4]. We mainly focused on restaurant reviews and English reviews. 3221555 reviews are used for classification, and a subset of reviews are used for reviews generation, which will be described in details in the following section.

Fig. 1 and Fig. 2 illustrate the word cloud of positive reviews(reviews with 4 or 5 stars) and negative reviews(reviews with 1 or 2 stars) accordingly. We can observe that there exists an obvious distinction between two kinds of reviews. The words that have the highest frequency among positive reviews are "really good", "first time" and "will definitely", while "go back", "customer service" and "will never" appear more among negative reviews. This distinction is fascinating and might influence our further classification and generation tasks.

Fig. 1: Word Cloud of Positive Reviews

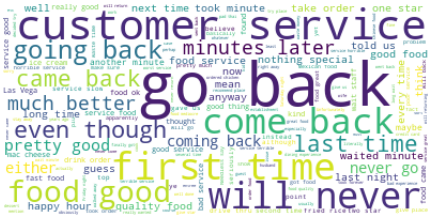

Fig. 2: Word Cloud of Negative Reviews

III. Methodology

A. Text Classification

1) Character-level convolutional networks: In this section, we introduce the design of character-level convolutional networks for text classification[1]. To build and train the model, we first build an alphabet which includes English characters, digits, and other characters and then quantize each character using one-hot encoding. Then the sequence of characters is transformed to a sequence of vectors with fixed length. Any character exceeding the given length is ignored, and any character which is not in the alphabet is represented by zero vector.

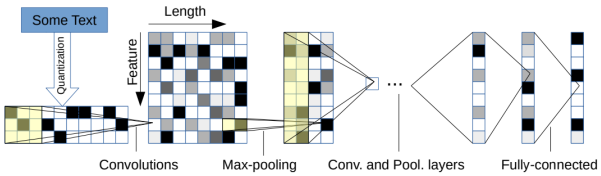

Fig. 3: Model Structure

The convolutional networks consists of 6 convolutional layers and 3 fully-connected layers and Figure 3 gives an illustration of the model. ReLU is used as activation function in covolutional layers. Dropout layers are used between the three fully-connected layers to regularize. Table I lists the configurations for convolutional layers and fully-connected layers.

TABLE I: Character-level CNN Classification Model

Convolutional Layers			
Layer	Filter	Kernel	Pool
1	256	7	3
2	256	7	3
3	256	3	N/A
4	256	3	N/A
5	256	3	N/A
6	256	3	3
Fully-connected Layers			
Layer	Output		
7	1024		
8	1024		
9	5		

2) Word-level CNN-LSTM networks: To build a word-level model, we first clean the text and convert them into sequences. Data processing involves removing stop words, punctuation, numeric texts and empty texts, converting words to lowercase, and stemming. Each review is limited to 100 words. Short texts less than 100 words are padded with zeros and the long ones are truncated. After processing, we tokenize the strings and create sequence of words. In this approach, convolutional layers are added before LSTM layer to reduce the training time, and the output of the LSTM layer is fed to fully-connected layers. Fig 4. gives an illustration of the structure.

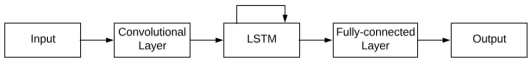

Fig. 4: CNN-LSTM Architecture

B. Text Generation

1) Character-level LSTM Generator: For generator, character-level LSTM is used. Similar to previous character-level LSTM classification task, only 96 unique characters including English characters, numbers and punctuation are kept in the text. In this step, we trained two different networks, one generating five star positive reviews, the other generating one star negative reviews. The model includes two LSTM layers and one Dense layer, as shown in Fig. 5.

```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 60, 1024)          4591616
_________________________________________________________________
lstm_2 (LSTM)                (None, 1024)              8392704
_________________________________________________________________
dense_1 (Dense)              (None, 96)                98400
=================================================================
Total params: 13,082,720
Trainable params: 13,082,720
Non-trainable params: 0
```

Fig. 5: Character Level LSTM Generation Model for One Star Reviews

For generation, the model is sent with a seed text, as shown in Fig. 6. The model would predict next character.

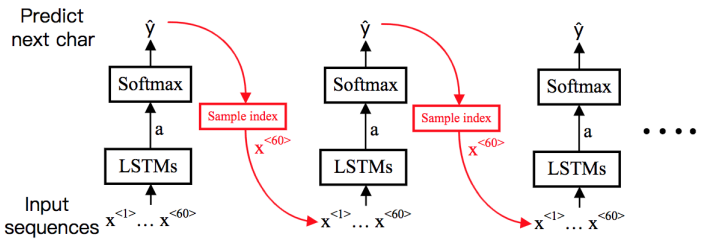

Fig. 6: Character Level LSTM Generation Model with seed[3]

2) Word-level LSTM Generator: Word-level LSTM is slightly different from character level LSTM in generation. Similar to classifier, it requires more pre-processing steps and larger dictionary to represent each word. The model includes 2 512-unit LSTM layers each followed by a dropout layer and one dense layer. The goal is to train a RNN with long text and thus obtain the next word. We used the same text as described in 1).

IV. Experiment

A. Text Classification

We test the first model: character-level CNN, in which we defined six layers. For each layer, we use 256 Filters. We use dropout to regularize this model. The model was trained using batch size of 128 and Adam optimizer. We obtained the following results after we run this model ten epochs as shown in Fig. 7:

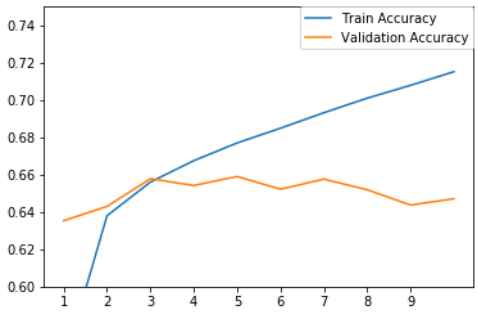

Fig. 7: Character Level LSTM Classification Accuracy

As evident, the model is not overfitting in just 10 epochs. The training accuracy went from 0.5583 to 0.7152, while the validation accuracy went from 0.6353 to 0.6471. Although the train loss decreases from 0.9779 to 0.6524, the validation loss stayed consistently around 0.81. We should be able to improve it when we train this model more epochs.

We test the second model: word-level CNN-LSTM, in which we defined two LSTM layers with dropout regularize and 100 neurons each. The model was trained using batch size of 128 and the optimizer we selected is Adam. After we run this model 10 epochs, we get the following results as shown in Fig. 8:

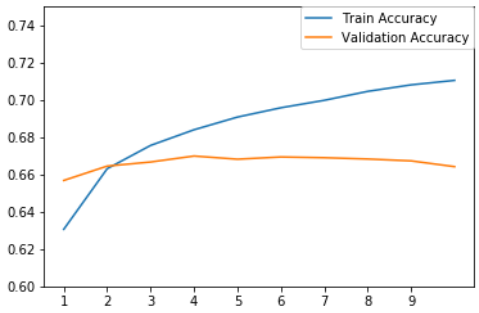

Fig. 8: Word Level LSTM Classification Accuracy

B. Text Generation

For generation, we send the model with a seed text of random characters and ask the model to predict the next character. Some examples of 5-star reviews are as following.

> "This place is great. The food is delicious and the staff are super friendly."
>
> "I love this gem of a place! The food and smoothies are amazing. Their canes are fantastic. I have been going there for years, and they've never been singing everywhere."
>
> "Great new Sunday donut place. It was delicious and the servers were very attentive. We are here on a whim and loved it. I really like the other options for price"

We also generate some negative 1-star reviews as follow.

> "We didn't enjoy lunch and my hashed the restaurant in the meal. The food was so disappointing."
>
> "Stopped in me for the pizza. Walked out and the food was not one server the worst place in my food poisoning."
>
> "wait for naan bread is ridiculous. Ordered fried noodles and the worst part of the server was with the bar and they take a burrito in the manager. "

The above examples suggest character-level LSTM generator can mimic grammatically correct sequences. We also build a word-based LSTM model but those generated texts are far from making sense. We will keep working on the word-level model.

V. Discussion

In this section, we mainly discuss the pros and cons of word-based model and character-based model. Word-based model displays lower computational cost than character-based model. This is because character-based model has much longer sequences and requires bigger hidden layers to capture long-range dependencies between how earlier parts of the sequence affect the later parts.

Character-based model better models infrequent and unknown words since character information can reveal structural (dis)similarities between words and can even be used when a word is out-of-vocabulary. [5] Besides, character-based model generates better texts than word-based model as suggested above. It can generate grammatically correct texts for a wide range of languages while word-based model generates more coherent texts but these generated texts may not make sense. Table II summarizes pros and cons of character-based model and word-based model.

TABLE II: Difference between Word-based and Character-based Model

	Word-based	Character-based
Lower computational cost	•	
Capture long-range dependency	•	
Learn large vocabulary		•
Generate text		•

VI. Future works

In the future, we could further explore building a fake review filter to identify the generated reviews. Also, Generative Adversarial Networks(GAN) has not been widely used in natural language processing because of its discrete natural. However, recent researchers combined Reinforcement Learning(RL) and conditional GAN or fill in the blanks[6]. Moreover, we intend to train on foreign languages such as Chinese reviews to see the difference. For character level LSTM, Chinese characters are difficult to train because Chinese contains much more characters than English. How to overcome the challenge still requires more research.

References

[1] Zhang, X., Zhao, J. and LeCun, Y. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, pp. 649-657, 2015.

[2] Graves, Alex. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[3] How to generate realistic yelp restaurant reviews with Keras. https://www.dlology.com/blog/how-to-generate-realistic-yelp-restaurant-reviews-with-keras/

[4] Yelp dataset challenge, 2018. https://www.yelp.com/dataset/challenge

[5] Lyan Verwimp Joris Pelemans Hugo Van hamme Patrick Wambacq, Character-Word LSTM Language Models. *European Chapter of the Association for Computational Linguistics (EACL)*, pp. 417-427, 2017.

[6] Fedus, William and Goodfellow, Ian and Dai, Andrew M, Maskgan: Better text generation via filling in the _. *arXiv preprint arXiv:1801.07736*, 2018.