Fine-Grained Sentiment Analysis of Restaurant Customer Reviews in Chinese Language

Suofei Feng[1] *suofeif@stanford.edu*
Eziz Durdyev *eziz@stanford.edu*

***Abstract*—Chinese language processing is a challenging topic in the well-developed area of sentiment analysis. In this project we implement 3 types of 4-class classification models (SVM, XGBoost, LSTM) for the fine-grained, or aspect-level sentiment analysis of restaurant customer reviews in Chinese language. There are 20 aspects for classification, each representing one type of target information in the reviews. We separately train one model for each element. The overall results of the models on the 20 aspects shows that XGBoost has the best performance, based on the average accuracy, weighted F1 score, and efficiency.**

***Index Terms*—Chinese language, NLP, LSTM, SVM, XGBoost**

I. Introduction

THE era of information explosion, brings an increasing demanding on the ability to extract core message from billions of records of data. Sentiment analysis, or opinion mining, is widely applied to extracting and studying subjective information in texts. By quantifying the opinions or attitudes in a large bulk of texts in a few minutes, sentiment analysis has gained popularity in various business scenarios for retrieving customer responses. In recent decades, considerable progress has been achieved in sentiment analysis of English language. At the same time, a similar development comparable to the growth of market has not be seen in the scenario of Chinese language [1]. In our project, we propose to implement a fine-grained (aspect-level) sentiment analysis of restaurant customer reviews in Chinese language. The topic and data come from the 2018 AI Challenger competition.

The inputs are reviews about restaurant in Chinese language. The task is to classify each piece of review text into 4 classes (”not mentioned”[-2], ”negative”[-1], ”neutral”[0], ”positive”[1]) under 20 aspects. Each aspect (or element) represents one type of information about the business. We develop and train 3 types of models-LSTM (Long Short-Term Memory), SVM (Support Vector Machine), and XGBoost (eXtreme Gradient Boosting)-for each of the aspect.

[1]Department of East Asian Languages and Cultures, Stanford University.

II. Related Work

A. Machine Learning in Sentiment Analysis

Pang and Lee [2] briefly summarize the history of sentiment analysis, and describe the related works as computational treatment of ”opinion, sentiment, and subjectivity in text” (p8). Early machine learning approaches towards sentiment classification use unigrams (single words) and n-gram as features for subjectivity detection, and SVM, MNB (Multinomial Naïve Bayes) for classification [3] [4].

SVM has longer history with sentiment analysis comparing to the gradient boosting and LSTM. It is one of the most used classification methods in sentiment analysis due to its ability to” generalize well in high dimensional feature spaces” [5]. Pang et al [6] applied Naïve Bayes, maximum entropy classification, and support vector machines in classifying IMDb reviews by sentiment, determining whether a review is positive or negative. Although the differences were not large, SVMs achieved higher three-fold cross validation accuracy, 0.829 than Naïve Bayes, 0.804 and maximum entropy classification, 0.81. As this movie review sentiment analysis is comparable to our restaurant review sentiment analysis, we decide to build SVMs to classify sentiments in 20 elements in our project.

On the other hand, although gradient boosting is one of the most applied “off the shelf” methods in general classification tasks, surprisingly, application of boosting ensemble method is not common in text classification. Ehrentraut et al [7] applied SVM and gradient tree boosting in classification of Swedish patient records (texts) to detect hospital-acquired infections. Gradient tree boosting achieved higher performance over SVM in precision, recall, and F1 score. Chen and Guestrin [8] proposed XGBoost, a scalable and regularized version of gradient tree boosting. Encouraged with these results

and improvements, we decide to apply XGBoost in our project as we needed a scalable method to classify 20 elements in Chinese restaurant review data.

Recent years have seen a substantial progress in NLP tasks with neural network approaches. LSTM is popular in sequence modeling for sentiment classification because of its advantage against gradient vanishing or exploding issues in long texts. Wang et al. [9] proposed an attention-based LSTM model with aspect embedding for aspect-level sentiment classification. They experimented with restaurant customer reviews in English, and implemented a 3-class (positive, negative, neutral) classification on 5 aspects: food, price, service, ambience, anecdotes/miscellaneous. The accuracy of this model improved by 2% compared with standard LSTM model. However, considering the different natures of Chinese language, and the large number of aspects for classification (20), we decide to start with standard LSTM model for the neural network approach in this project.

B. Chinese NLP Researches

A review of sentiment analysis in Chinese language was given by Peng et al [10]. Apart from conducting sentiment analysis directly on Chinese language, there is another approach: transform the task to sentiment analysis on English language by machine translation. In our project, we conduct a mono-lingual experiment by directing extracting features from original Chinese language. This is mainly because that the style of our input texts is highly colloquial and context-specific, which might lose information in the process of machine translation. According to this article, good results were gained from a combination of neural network (word2vec) for word representation and SVM for classification.

Peng et al. also mentioned the different techniques for segmentation. As Chinese language does not have space between words, it is necessary to use segmentation tools to extract words as the basic units of semantic meaning. They summarized that Jieba had a high speed and good adaptation to different programming languages. For these reasons, we decide to use Jieba as our segmentation tool.

III. Dataset and Features

A. Dataset

We use the data sets provided by AI challenger official [11]. The training and validation data are manually labelled. They also provided test dataset without labels. In the training dataset, there are 105,000 records of reviews, with labels of 4 classes {"positive"[1], "neutral"[0], "negative"[-1], "not mentioned"[-2]} on 20 aspects/elements under 6 categories. The validation set has 14998 records of reviews. For the aim to evaluate our models by ourselves, we split the validation set into a smaller validation set (first 7500 records in the original validation set) and a test set (rest 7498 records in the validation set) with true labels. We make 20 plots for class distributions of each of the three datasets, which show that the class distributions are very similar across all three datasets. Table I shows all the elements and corresponding categories. Here is an example input text. Because of the limited space, we just put part of the review and its translation here:

"吼吼吼，萌死人的棒棒糖，中了大众点评的霸王餐，太可爱了。一直就好奇这个棒棒糖是怎么个东西，大众点评给了我这个土老冒一个见识的机会。看介绍棒棒糖是用德国糖做的，不会很甜，中间的照片是糯米的，能食用，真是太高端大气上档次了..."

Translation:
"Ha ha ha, the lollipop is soooo cute. I won the 'free meal prize' on Dazhongdianping [author's comment: similar to Yelp], this is so cute. I have been always curious about what the lollipop is like. Dazhongdianping gave me the bumpkin this opportunity to open my eyes. The introduction said it was made using German candy, not too sweet. The photo in the middle is made of glutinous rice, edible. It is really high-end..."

A glance of Google Translation:
"Hey, the lollipop of the dead man, the overlord meal of the public comment, so cute..."

B. Feature Extraction

The main challenge in our project is preprocessing our data. Chinese language is difficult to accurately segment because of absence of space, variant lengths of words, and high flexibility of making new words. We apply same preprocessing approaches to the training, validation, and test dataset. With reasons presented in the related work section, we use Jieba cut for segmentation. After segmentation, we gain three lists of word lists produced by segmenting the lists of sentences. We then train a Word2Vec model following the instructions in [12], and produce a vocabulary and embedding matrix of the same word order with the vocabulary. Pad token and unknown-word token are added to the vocabulary as well as the embedding matrix. The next step is to digitalize

TABLE I: Elements

Location	traffic
	distance from business district
	easy to find
Service	wait time
	waiter's attitude
	parking convenience
	serving speed
Price	price level
	cost-effective
	discount
Environment	decoration
	noise
	space
	cleanness
Dish	portion
	taste
	look
	recommendation
Others	overall experience
	willing to consume again

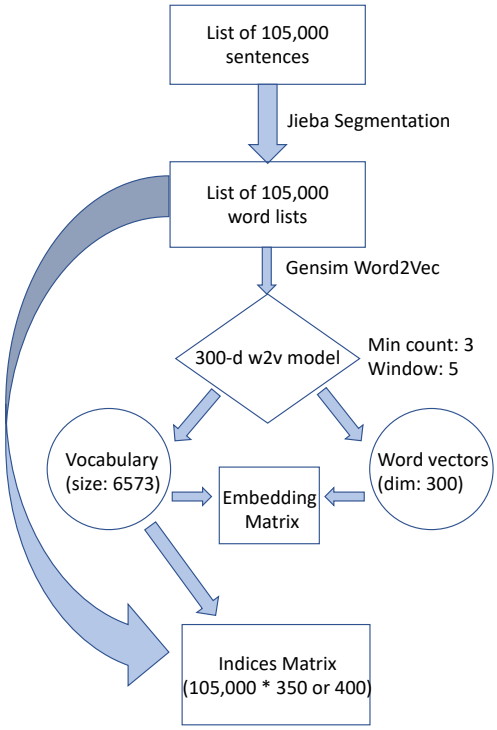

Fig. 1: Preprocessing Flowchart

the input texts. We replace the words in each sentence (after segmented) with their indices in the vocabulary. All the sentences are padded or cut to a length of 350 for SVM and XGBoost, and 400 for LSTM. The outputs are three matrices for train (105000, 350 or 400), val (7500, 350 or 400), and test data (7498, 350 or 400). The following is a graph showing the procedures of preprocessing training data (Fig.1).

IV. Classification Models

A. Baseline

The baseline model is provided by AI Challenger official [13]. The feature is extracted by TF-IDF framework, whose values for representation are based on the frequency of a word in a given document. The classification model is RBF SVC. The average F1 score across the 20 elements is around 0.2.

B. LSTM

LSTM, or Long Short-term Memory network, is a type of recurrent neural network (RNN) to process sequence of information, by feeding the output of preceding neurons to subsequent neurons. Unlike traditional RNN, LSTM networks are not vulnerable to gradient explosion or vanishing when dealing with long sequences of data. This is achieved by forget gate, input gate, and output gate in each hidden unit. These gates decide how much information to let through, and therefore can connect information with a wide gap between [14].

We build a many-to-one LSTM model for each of the 20 elements. Fig.2 shows the structure of the model. The inputs are the output indices from preprocessing step. The labels are transformed to one-hot vector, each as a (1, 4) row vector. The embedding matrix is used as the weights for the embedding layer, which is not trainable. We add arbitrary class weights to address the class imbalance problem. The loss function is categorical cross-entropy:

$$L(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{4} y_{ij}\log(p_{ij}) \tag{1}$$

with n the number of examples, j the class ID, y the true label, and p the predicted probability. Accuracy and weighted F1 score are the evaluation metrics.

C. Support Vector Machine

SVM classifier is one of the most preferred classification method among classic machine learning methods

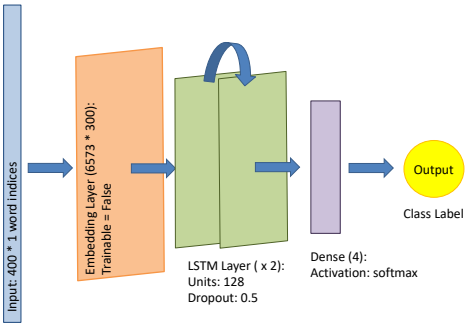

Fig. 2: Two-Layer LSTM Model

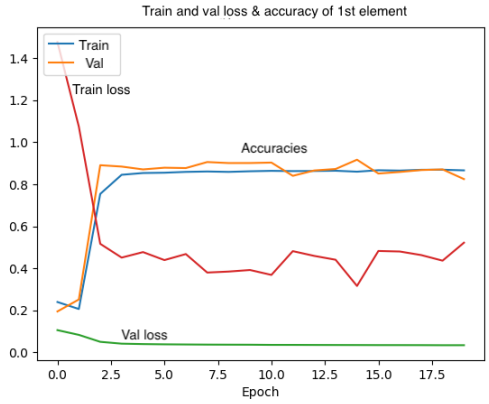

Fig. 3: Training and validation losses & accuracies of 1st element

since they generalize well in high dimensional feature spaces. We first extract the embedding vectors for each word in a sentence through indices, to form a sentence matrix. Then we create sentence feature vectors for training, validation, and test sets by averaging the sentence matrix along the vertical axis to get a vector. Each observation is a vector of size (1, 300). With 10 fold cross validation error, we get the best kernel as "linear".

D. XGBoost

We feed our XGBoost models with the same input with SVM models. We use GridSearchCV function from sklearn package in python progaramming language to tune for parameters 'learning rate': [0.01, 0.05, 0.1] and 'max depth': [3,5] with 5 and 10 fold cross validation. With both 5 and 10 fold cross validation, we get the best XGBoost parameters as learning rate=0.1 and max depth=5. Although XGBoost is not a highly preferred method in text classification, it yield the best results based on test set accuracy and F1 score.

V. Results and Discussion

A. Results

After tuning on a subset of 500 training records and 100 validation records, we choose 0.5 for LSTM layer dropout and recurrent dropout, 128 for number of hidden units, 128 for batch size, and Adam for optimizer with learning rate of 0.001, β_1 of 0.9, β_2 of 0.999. At the beginning we tried 50 epochs for all the elements, and found most of them converge after around 14 to 15 epochs. We therefore decide to train for 20 epochs. As we have 105k records in the training dataset, batch size of 128 will make the training process comparatively fast. Adding different arbitrary class weights to different elements does not have clear improvement. Here is the plot (Fig.3) of the accuracies and losses of training and validation of the first element. To make this report concise and short, we will not report all the training details and statistics of all the 20 elements here.

Generally the LSTM model performs well when sentiment features are apparent:

"一直经过这条路第一次进去拔草还是通过看美食节目首先说说环境还是很不错的感觉很适合小情侣来很温馨的感觉喝喝下午茶感觉特别好服务也很好哦都很勤快可能不是周末中午人不多很安静非常喜欢这样的气氛再说说美食点了一个新款黄桃冰激凌披萨薄薄的披萨真的蛮好吃也以前一直吃厚的现在发觉薄的也不错下次试试榴莲披萨日式猪排饭真的量好多比图片看起来还多就是有点偏咸了意式千层面咬起来都是芝士的味道厚厚的感觉好吃小食还行量挺大玫瑰巧克力和榛果海盐拿铁真的都好好喝噢下次再去必点目前大众点评买单还能享受95折真的挺划算以后还会经常光顾的"

(Rough) Translation:
"Always walk through this road. This is my first time walk in the restaurant. Know it through TV show. Environment is pretty good, suitable for couples, warm feeling. It's nice to have afternoon tea. Service is good, too...Maybe because it's not weekend, not so many people in the noon. It's quiet. Really like the atmosphere. About food, ... pizza really good, wanna try another type next time... The pork combo has a large portion, just a little too salty...Drinks are tasty. Must try next time. Using Dazhongdianping will give you 5% discount, a good bargain. Will come often in the future."

The predictions results are shown in Table II.

Our linear kernel SVM models with the new sentence features vectors showed great improvement over baseline model. SVM model prediction F1 scores and

TABLE II: Predictions of the Example Text.

1: -2	2: -2	3: -2	4: -2	5: 1
6: -2	7: -2	8: -2	9: -2	10: 1
11: -2	12: 1	13:-2	14:-2	15:1
16: 1	17: -2	18: -2	19: 1	20: 1

test accuracy for each 20 elements range from 0.44 to 0.94 (Fig.4). XGBoost yields better results in terms of test accuracy and F1 Scores. Fig.5 shows test accuracies across 20 elements with LSTM, SVM, and XGBoost. According to the graph, LSTM has the most fluctuating performance over the 20 elements, while XGBoost is relatively more stable with higher accuracies. However, we observed in the original datasets that the class imbalance problem is serious, we cannot rely on accuracies to evaluate our models. Weighted F1 scores are investigated for a better knowledge of the performances.

TABLE III: Weighted F1 scores of Test Dataset for 3 Topics

Models	Dish Recomm.	Wait Time	Traffic Convenience
LSTM	0.6524	0.8776	0.8389
SVM	0.7561	0.8381	0.8563
XGBoost	0.7582	0.8326	0.8382

We also check confusion matrices from XGBoost, which are presented as following. The models are biased towards dominating classes (such as -2 in E13, and 1 in E15). In both matrices, columns represent true labels -2, -1, 0, and 1 from left to right, and row values represent predicted values for the actual values -2, -1, 0, 1 from top to bottom.

$$E13 = \begin{bmatrix} 4444 & 1 & 8 & 265 \\ 331 & 3 & 3 & 35 \\ 565 & 2 & 7 & 89 \\ 1331 & 2 & 4 & 408 \end{bmatrix}$$

$$E16 = \begin{bmatrix} 46 & 14 & 91 & 206 \\ 8 & 26 & 159 & 87 \\ 16 & 21 & 1456 & 1399 \\ 15 & 3 & 785 & 3166 \end{bmatrix}$$

B. General Discussion

We use weighted F1 score along with accuracy because of the imbalance in class distributions which is prevalent across the elements. The confusion matrix of element 13 ("space") reveals the problem in XGBoost model, which also exists across all models. Class weights might be a good strategy, though in LSTM it does not show clear advantage. On the one hand, the situation might be improved through more precise assignments of class weights, e.g. 1/(number of class-j examples in the training data). On the other hand, data augmentation such as bootstrapping minor classes might help. Because of the demand on the integrity of context to judge about the sentiment, cropping is not suitable in this case. Changing the key feature words might also worth trying.

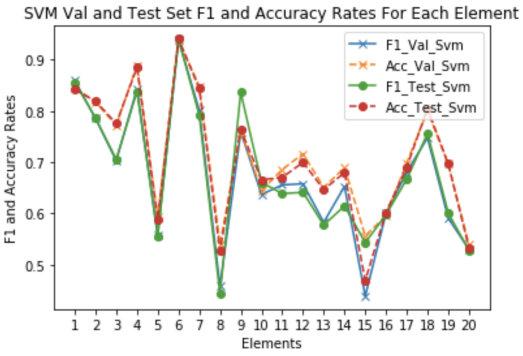

Fig. 4: SVM Weighted F1 Scores and Test Accuracies across 20 elements

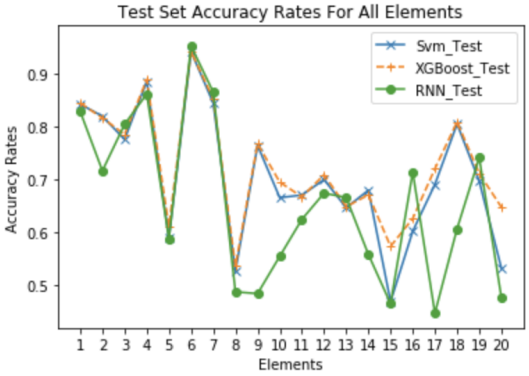

Fig. 5: Test set accuracy across 20 elements with LSTM, SVM and XGBoost

VI. Conclusions and Future Works

In general, XGBoost yield better results in terms of F1 scores and test accuracies. Our models improve from baseline model partially due to better preprocessing, and partially due to better-tuned hyperparameters. Apart from the aspects mentioned in the Discussion section, this task can be improved in the following ways: 1.) collect data with higher label quality (some examples are difficult to classify even for human beings); 2.)improve the quality of language models with contextual representation, e.g. BERT; 3.) Moreover, we might benefit from applying attention mechanism for long input texts.

VII. Contributions

The team members contribute equally to the project. Suofei was responsible for the training of baseline model, data preprocessing, and the construction and training of LSTM model. Eziz contributed to the construction and training of SVM and XGBoost models. Two members both worked on poster making and report writing.

VIII. Code

The code can be found at:

https://github.com/suofeif/CS229-Project.

References

[1] H. Peng, E. Cambria, and A. Hussain, "A review of sentiment analysis research in chinese language," *Cognitive Computation*, vol. 9, pp. 423–435, 2017.

[2] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends® in Information Retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008.

[3] E. Boiy and M. Moens, "A machine learning approach to sentiment analysis in multilingual web texts," *Information Retrieval*, vol. 12, no. 5, pp. 526–558, 2009.

[4] R. P. . S. H. Manning, C. D., *Introduction to information retrieval*. Cambridge University Press, 2008, ch. 13.

[5] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," *European Conference on Machine Learning*, vol. 1398, pp. 137–142, 2005.

[6] L. L. . V. S. Pang, B., "Thumbs up?: sentiment classification using machine learning techniques," *In Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, vol. 10, pp. 79–86, 2002.

[7] E. M. T. H. T. J. D. H. Ehrentraut, C., "Detecting hospital-acquired infections: A document classification approach using support vector machines and gradient tree boosting." *Health informatics journal*, vol. 24, no. 1, pp. 24–42, 2018.

[8] C. T and G. C. XGBoost, "Xgboost: A scalable tree boosting system," 2016. [Online]. Available: http://dmlc.cs.washington.edu/data/pdf/XGBoostArxiv.pdf

[9] H. M. Wang, Y. and L. Zhao, "Attention-based lstm for aspect-level sentiment classification," *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 606–615, 2016.

[10] C. E. Peng, H. and A. Hussain, "A review of sentiment analysis research in chinese language," *Cognitive Computation*, vol. 9, no. 4, pp. 423–435, 2017.

[11] Ai challenger 2018. [Online]. Available: https://challenger.ai/competition/fsauor2018

[12] W. Gong. (2017) Chinese word vectors. [Online]. Available: https://primer.ai/blog/Chinese-Word-Vectors/

[13] Ai challenger 2018 baseline. [Online]. Available: https://github.com/AIChallenger/AI_Challenger_2018

[14] C. Olah. (2015) Understanding lstm networks. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/