A Deep Learning Architecture for Analysis of Airline Reviews Natural Language Processing

Final Project of Introduction of Text Mining

Leonard Tsai	Ethan Chu	Brian Hong	Justin Shih	Bradley Chen
Department of Information Management	Department of Information Management	Department of Information Management	Department of Information Management	Department of Information Management
National Taiwan University	National Taiwan University	National Taiwan University	National Taiwan University	National Taiwan University
b10705010 @ntu.edu.tw	b10705043 @ntu.edu.tw	b10705027 @ntu.edu.tw	b10705015 @ntu.edu.tw	b10705017 @ntu.edu.tw

ABSTRACT

Skytrax airline reviews from 2016-2019 include ratings and written comments on six services offered on airplanes, including Seat Comfort, Cabin Service, Food Quality, Ground Service, Entertainment, and Value for Money. While the written comments reflect the personal experiences and opinions of customers, the ratings, which range from 1 to 5, are subjective and may not accurately reflect the objective reality of the services. As a result, analysis of the written comments in the reviews may be more useful in understanding customers' experiences and sentiments.

Through text mining, we can design a model to automatically process a huge number of reviews and find out what part of the service should be improved in those negative reviews. The procedures are as follow:

Firstly, extract training data (40,000) from the preprocessed data (64,095) and the rest is testing data. Then, use training data to train the six RNN models and create the six BOWs relative to each service provided on airlines, such as “Seat Comfort”, “Cabin Service”, “Food Quality”, “Ground Service”, “Entertainment”, and “Value for Money.” Moreover, airline reviews from Skytrax not only comprise the review texts also comprise the scores on six services.

- RNN models (for each service): Consider reviews with service score between 1 to 2 as negative reviews; reviews with service score between 3 to 5 as positive reviews. Therefore, the RNN model is trained to differentiate whether the review is negative or positive depending on each service.

- Skip-gram-like (SGL) Method with BOW (for each service, take “Seat Comfort” as an example): Recall the negative reviews from training data used in RNN models. Search for the adjectives around some keywords in those reviews. Specifically, the keywords might be “seat”, “uncomfortable”, “sit”, etc. Because those reviews are already negative, the found adjectives are much likely negative as well. After the completion of the dictionary of adjectives, add some synonyms of those adjectives to the bag of words (BOW), so that the BOW is not too deficient.

Lastly, if the testing data is outputted as a negative review (for each service, take “Seat Comfort” as

an example), find the same words between the negative review and the BOW relative to its service. For instance, the words in both the review and the BOW might be “hard”, “upright”, etc. Therefore, we are enabled to determine if any review is negative. If so, the trained RNN models and the SGL with BOW will output which service the airline company should improve and enhance.

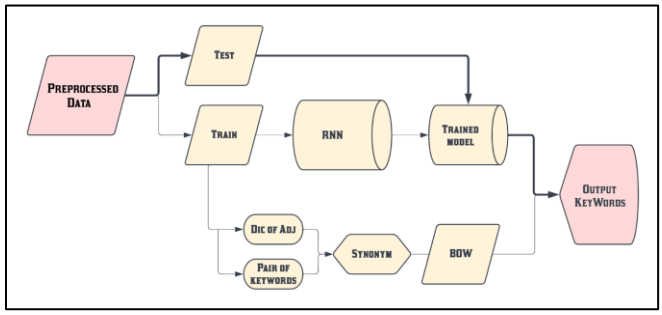

Figure 1: The structure of this project

We have successfully detected some keywords in certain reviews and those words truly reflect the essence of the review. However, for some example, our model outputs quite irrelevant words.

CCS CONCEPTS

• Computing methodologies → Natural language processing

KEYWORDS

Deep learning, Natural Language Processing (NLP), Recurrent Neural Networks (RNN), Skip-gram-like method, Airline reviews.

ACM Reference format:

Leonard Tsai, Ethan Chu, Brian Hong, Justin Shih, Bradley Chen. 2023. A Deep Learning Architecture for Analysis of Airline Reviews Natural Language Processing. Final Project of Introduction of Text Mining, NTU, Taipei, Taiwan

1 Introduction

Skytrax airline reviews from 2016-2019 contain ratings and written comments on six services offered on airplanes, including Seat Comfort, Cabin Service, Food Quality, Ground Service, Entertainment, and Value for Money. The written comments reflect customers' experiences and opinions, while the ratings are subjective and may not accurately reflect the objective reality of the services. Text mining can be used to automatically analyze a large number of reviews and identify areas for improvement in negative reviews. The process involves training six recurrent neural network (RNN) models to classify reviews as positive or negative based on the service ratings, and using the Skip-gram-like method with bag-of-words to identify relevant keywords in negative reviews. The trained RNN models and bag-of-words can then be used to determine which services the airline should improve based on negative reviews.

2 Related work

2.1 RNN

2.1.1 What is RNNs? Recurrent neural networks, known as RNN, is a kind of neural network of deep learning. When activated, RNN will selectively pass input data across sequence steps while processing sequential data one element at the same time. Compared to different kinds of neural networks like CNN or DNN, RNN is capable of memoring. According to the above, it has a hidden state inside its networks. Although it cannot memorize much information, it can record the last output. Hence, every time when RNN has to output something, it will first refer to the information in the hidden state. This allows RNN to learn the correspondence with the previous text.

2.1.2 Problem of traditional RNNs? However, traditional RNN has a severe problem that will have a gradient vanishing problem (often appears in long-term sequence learning processes) while processing the data. Therefore, people come up with two solutions: gated recurrent units and long short-term memory. The idea behind these two solutions is to use a set of gates in order to regulate the value that flows into each hidden state. So the gradient will be refrained from nearly to zero and the whole gradient vanishing problem will no longer bother us.

2.1.3 Why do we use RNNs? While dealing with NLP tasks, RNN can capture long-term linguistic dependencies and improve classification performance. Because of the characteristic that it can be effective when dealing with NLP tasks, we use RNN to help us with the comments that customers give.

In our real life, RNN can be used in many ways. For example: helping caption the image, predicting the prices of stocks in a particular month or year, and the most important one, natural language processing. In our case, it will help us to deal with the comment so that we can classify or analyze these comments.

2.2 Find the BOW

2.2.1 Skip-gram-like (SGL) Skip-gram in word vector aims to predict the words surrounding the key term by acquiring words' embeddings. That's quite similar to our case in which we expect to find the "main flaws" surrounding the specific terms, but simpler. For example, if we want to discover the flaws about the SEATS, it is quite likely that those adjectives describing the seat appear near the word 'seat.' Therefore, we first need a function to find words that get paired with the specific terms. Embedding is not needed in our case [4].

2.2.2 Dictionary of negative adjectives The 'Pairing' function now finds pairs, [seat, ____] as the output. However, the words are still too extensive and messy. We need to decrease the size of the pair list to make it more precise. But how? When customers talk badly about the seat, it is "negative adjectives" that frequently are used. Therefore, we get a dictionary of adjectives through those train reviews which are rated '1' on the seat so that those adjectives will more probably to be negative.

2.2.3 Get the BOW & Add synonyms to reference By matching the "Dictionary of Adj "and "Pairs of Keywords", we finally get the bag of words about certain terms. The last part with predicting the test data, in the seat case, is to find words that are in 2 spaces before and after the keywords 'seat', match them to the BOW, and then output the results if exist. But during the last matching process, we add 'synonyms' to the keywords reference. It is because when customers review about the seat, circumstances are that they don't use the word 'seat' but words close to 'seat'. By adding those synonyms, we hope to maximize all possible predicted outputs.

3 Proposed Architecture: Airline Reviews natural Language Processing (ARNLP)

3.1 Pre-preprocessing: convert "window-1252" into "UTF-8"

In the verified data that we got from the original dataset, there are still some punctuation marks that we have trouble reading. Thus, we first encode those punctuation marks into the format of windows-1252 and then decode them into the format of UTF-8. Besides, we use the parameter 'ignore' to avoid unnecessarily changing the format of a punctuation mark with correct format.

3.2 Train test data split

To construct the 6 models, we also remove some data with the vacancies in scores respectively for each part of service from the data. From the data produced respectively for each model, we pick the first 40000 records as the train data and the rest as the test data. Especially for the part of ground service, because there are only 37581 records in the data after removing vacant data, we pick the first 30000 records as the train data and the rest as the test data, instead.

3.3 RNN model

3.3.1 Structure of RNN First, we need to create a word index which contains the order of the frequency of each word appearing in all the remarks to turn each word into a number. Thus, we use functions *fit_on_texts*, *word_index*, and *texts_to_sequences* in library Tokenizer from *keras*. Besides, we take the most common words among the top 20000 in the remarks to set up the feature dimensions for each tokenizer model. For each remark, we also restrict its length to 150 words through the function *pad_sequences*.

Below is the summary of the framework of the RNN model. From the data produced through the above process, we use this structure to predict whether the remark is positive for the service or not.

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, None, 16)          320000

 simple_rnn_2 (SimpleRNN)    (None, 16)                528

 dense_2 (Dense)             (None, 1)                 17

=================================================================
Total params: 320,545
Trainable params: 320,545
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: The summary of the RNN model we set

3.3.2 Training Process From the test of accuracy of the RNN model (See 5.1 part), we decide to set the parameter epochs as 5 times and batch_size as 512. For every record, we set the service scores at 1 to 2 as 0, meaning as a negative review, and scores at 3 to 5 as 1, meaning as a positive review. Thus, after trained by the training data, each model will be able to predict whether a remark is positive at a service with a certain accuracy.

3.4 Skip-gram-like

3.4.1 Structure of SGL:

Function *text_preprocessing_*custom is responsible for preprocessing of reviews, stopwords, lemmatization, etc.

Function *reate_unique_word_to_dict* (sample): Input is a document of all preprocessed reviews appended. First use set(sample) to dismiss all repeated words. Then create a *unique_word_dict_containing_text* dictionary to store all unique words.

We pair two unique words within a fixed range in form [__, __], and then find all pairs that contain certain terms (‘seat’ in the seat case). Finally, we are enabled to find words surrounding the keywords.

output:

['seat', 'old'], ['seat', 'wear'], ['seat', 'old'], ['seat', 'food'], ['seat', 'two'], ['seat', 'wear'], ['seat', 'food'], ['seat', 'tightest'], ['seat', 'space'], ['seat', 'experience'], ['seat', 'space'], ['seat', 'apart'], ['seat', 'tightest'], ['seat', 'experience'], ['seat', 'apart'], ['seat', 'board'], ['seat', 'pass'], ['seat', 'plan'], ['seat', 'pass']

3.4.2 Establishment of dictionary of negative adjectives:

We take those data being rated 1 on the specific aspect as train data in each case. Use Tokenizer from *tensorflow. keras. preprocessing. text* to split the reviews. After processing, we get the key & rank of each word, then store those terms with higher rank of frequency in *keywords_seat*. Library *nltk.corpus* enables us to download the

dictionary of words of each part of speech. We use the adjective dictionary in this project. After running a loop to detect whether each word is labeled as 'JJ' in *nltk*, we get a dictionary of adjectives only. And because our train data are all negative reviews, those adjectives in our dictionary are mainly negative too by this method.

output:

['next', 'last', 'uncomfortable', 'extra', 'old', 'bad', 'poor', 'american', 'terrible', 'small', 'horrible', 'different', 'able', 'low', 'several', 'whole', 'hard', 'entire', 'additional', 'direct', 'possible', 'open', 'wrong', 'total', 'original', 'expensive', 'regular', 'ridiculous', 'empty', 'complete', 'single', 'impossible', 'unable', 'mechanical', 'unprofessional', 'unhelpful'.....

3.4.3 Development of BOW: Now we have pairs of words with certain terms, and a dictionary of negative adjectives, all we have to do is match them and get the BOW.

BOW:

['next', 'last', 'uncomfortable', 'extra', 'old', 'bad', 'poor', 'american', 'terrible', 'small', 'horrible', 'different', 'able', 'low', 'several', 'whole', 'hard', 'entire', 'additional', 'direct', 'possible', 'open', 'wrong', 'total', 'original', 'expensive', 'regular', 'ridiculous', 'empty', 'complete', 'single', 'impossible', 'unable', 'mechanical', 'unprofessional', 'unhelpful', 'normal', 'final', 'unacceptable', 'tiny', 'young', 'miserable', 'live', 'third', 'ready', 'unpleasant', 'aware', 'important', 'angry', 'inedible', 'swiss', 'upright', 'true', 'actual', 'pathetic', 'unhappy', 'certain', 'horrendous', 'unbelievable'.....

3.5.4 Synonyms: Wordnet from *nltk.corpus* has *wordnet.synsets* to let us acquire synonyms of words. By adding those synonyms of certain terms to the reference in the output process, we maximize the prediction number of output.

3.5 Output (Through Our Search Function)

We continue the example we use above. That is, an example about a seat. This "search" function is used to output the final data. First of all, the function will find the words which are before and after 'seat'. Then, we will apply the prediction function to predict this seat-related comment's rating. If the prediction tells us that the rating is low, we will search for the neighbor adjective of importance. After all, if that neighbor adjective is matched with those in BOW. We will output the adjective.

4 System outcomes

There will be 2 examples to show the results of our models, and one of them is positive and the other is negative. Due to the length, we will just pick up some parts of the remark here.

4.1 Example 1 (Positive):

4.1.1 Remark: Flew Business Class to and from Perth to Johannesburg on an A340-300. Very ***comfortable relatively flatbed seats, excellent*** crews and service and good choice of quality meals. Only gripe on the A340-300 is that the cabin temperature on these night flights was set way too high and was exceptionally uncomfortable with crew resistant to lowering it and with one staff member claiming it was a defect of the A340' s! The Business Class service on the 738' s between Johannesburg and Cape Town was good but the aircraft are showing their age and if anyone looks at floor level you 'll notice grubbiness and that the IFE units are coming apart - not a good look.

4.1.2 Score for Seat Comfort: 5

4.1.3 Prediction about the need to improve on seat: No

4.1.4 Result from Our Model: No need to improve on 'Seat Comfort.'

4.2 Example 2 (Negative):

4.2.1 Remark: A decent internal flight from Beijing to Chengdu, the flight ran on time and the service desk was helpful. Beijing Capital is a lovely airport from which to fly. I was surprised to receive food on the flight, but it was ***inedible***, so I would not recommend trying it. The seats were comfortable. The problem was the constant barrage of announcements, many of which were superfluous and unnecessary which made it impossible to truly feel at ease. Otherwise I would fly Air China again.

4.2.2 Score for Seat Comfort: 2

4.2.3 Prediction about the need to improve on seat: Yes

4.2.4 Problem for Seat Comfort Caught by Our Model: {'inedible'}

5 Evaluation

5.1 RNN Accuracy

Below are the line charts of loss and accuracy for each service. In order not to overfit, we set epochs as 5 for each model. As epochs are as 5, for each chart, we can observe that the accuracy rate is always higher than the loss rate and more stable than other times of epochs. Thus, as the statement in part 3.3.2, we will use epochs as 5 and batch_size as 512 to train our RNN model.

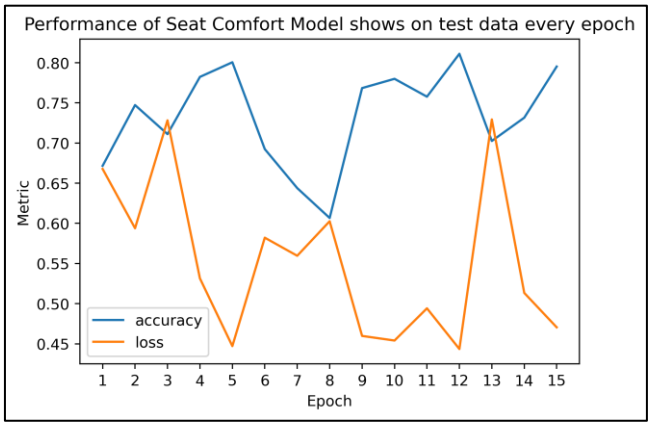

Figure 3: Accuracy loss chart of seat comfort prediction

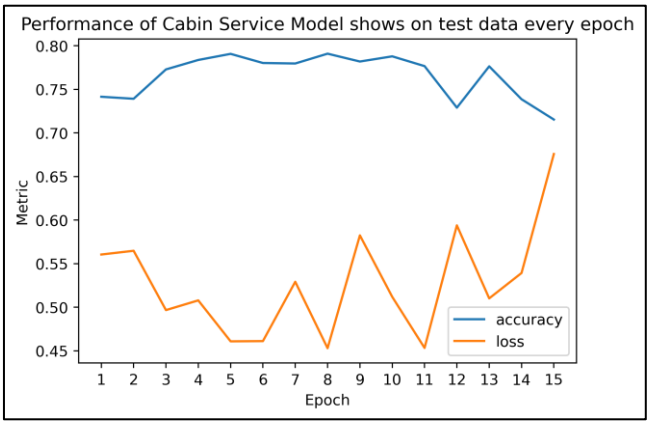

Figure 4: Accuracy loss chart of cabin service prediction

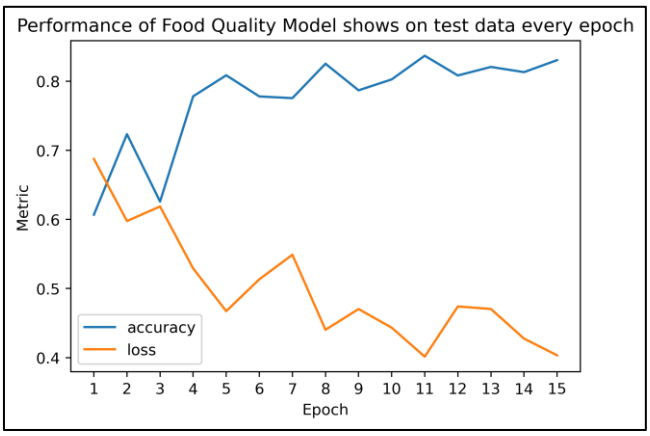

Figure 5: Accuracy loss chart of food quality prediction

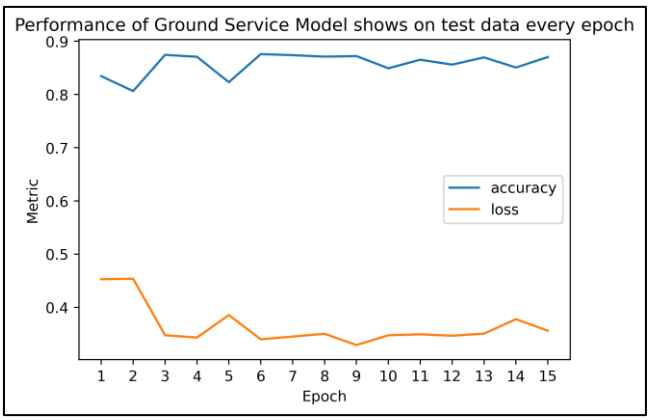

Figure 6: Accuracy loss chart of ground service prediction

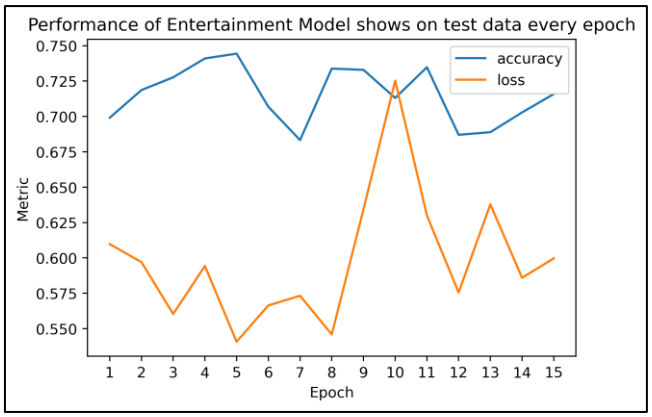

Figure 7: Accuracy loss chart of entertainment prediction

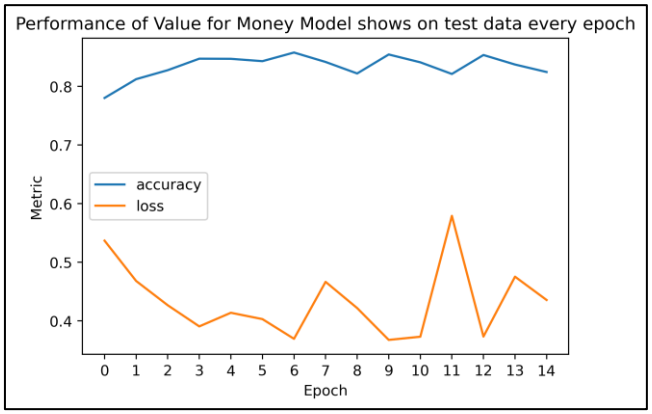

Figure 8: Accuracy loss chart of value for money prediction

5.2 Output

From the examples in part 4, we can find out that our models actually produce some expected result. However, there are still some problems that need to be solved. For example, there are some words that have different meanings for different services, which will have the wrong results. The solution of this problem isn't implemented in this project, but it will be mentioned in part 6 later.

Besides, in the word index storing the rank of the frequency of each word appearing in the whole remarks, there are many meaningless words like the airline company's name or the aircraft types. Due to the high rank of those words, the models may still be affected by this.

Last, sometimes, people will leave a remark that is different from the scores they gave, which will also largely affect the judgment of our models and increase the rate of wrong predictions.

6 Future Possible Improvements

Though the model we designed indeed outputs the flaws of each aspect in an airline company, there are common outputs in all six aspects. For example, we found aircraft types and customers' emotional words. Also, there were complaints of seat comforts in the output of the food services part. So, we can find a way to do segmentations to detect the keywords in customers' reviews and separate them into the corresponding aspects of services before we put the reviews into our model. Therefore, we could prevent outputting common or wrong problems.

7 Conclusion

Customers' written reviews are critical to companies; however, boundless reviews with countless words relating to different services of the companies are often collected, which makes it hard for managers to manually find out the essence of the opinions. Through text mining, we designed a model to automatically process a huge number of reviews and find out what part of the service should be improved in those negative reviews. In this model, we first used training data to train RNN models for the six services, to differentiate whether the review is positive or negative.

We believe the biggest novelty of our work lies in the application of the Skip-gram-like (SGL) Method with bag of words (BOW) on the negative reviews from the training data used in RNN models. Searching for the adjectives around the keywords in those reviews. Since they are negative reviews, there must be negative

adjectives that can reflect airline companies' flaws. After completing the dictionary of negative adjectives, we add some synonyms of those adjectives to enrich BOW, making it more abundant.

Lastly, we collected the output of negative reviews and got keywords that match companies' defects in different aspects. Although there are common words and unnecessary words in all aspects, we believe that it could be improved by the addition of segmentation in the future. In short, our work can process a huge number of reviews and extract the keywords of the written comments to help companies succeed in finding what part of services that need to be improved right away. It can be applied to business use and reduce the waste of manpower and resources.

ACKNOWLEDGMENTS

We would like to express our sincere gratitude to our supervisor, Prof. Chien-Chin Chen, for his teaching and advices throughout this project. His wisdom, expertise and passion have been invaluable in helping us overcome various of challenges in this that arose during the course of this project. His willingness to offer time and resources to ensure the success of this project is greatly appreciated and has not gone unnoticed. We are deeply grateful for the mentorship and support, and we feel fortunate to have had the opportunity to work with such talented and dedicated professor. Thank you for everything.

REFERENCES

[1] Faizan Ahmad et al. (2020, Feb.). A Deep Learning Architecture for Psychometric Natural Language Processing. Emory University and Georgia Tech. https://dl.acm.org/doi/fullHtml/10.1145/3365211

[2] Carolina Bento. (2022, May.). Recurrent Neural Networks Explained with a Real Life Example and Python Code. https://towardsdatascience.com/recurrent-neural-networks-explained-with-a-real-life-example-and-python-code-e8403a45f5de

[3] Ajitesh Kumar. (2022) Neural Network Types & Real-life Examples. https://vitalflux.com/deep-neural-network-examples-from-real-life/amp/

[4] Angel Das. (2022, Nov) Generating Word Embeddings from Text Data using Skip-Gram Algorithm and Deep Learning in Python. https://towardsdatascience.com/generating-word-embeddings-from-text-data-using-skip-gram-algorithm-and-deep-learning-in-python-a8873b225ab6

[5] Chien-Chin Chen. (2022, Nov.). Word Vector, [Course presentation]. Department of Information Management, National Taiwan University.

[6] Chien-Chin Chen. (2022, Nov.). Deep Learning, [Course presentation]. Department of Information Management, National Taiwan University.