Contents

1 Introduction **2**

2 Application **2**
2.1 Advantages 2
2.2 Modules 3
2.3 Sentiment Analysis Applications 4

3 Methodology **4**
3.1 Data Pre-Processing 4
3.2 Model Building 4
3.3 Training 5
3.3.1 Data Selection 5
3.3.2 Labeling Criterion 5
3.3.3 Optimization 7
3.3.4 Evaluation 8
3.4 Prediction 8

4 Approach **8**
4.1 Data preprocessing 8
4.1.1 Words Vectors 8
4.1.2 Segmentation length distribution 9
4.1.3 Padding 9
4.1.4 Words Embedding 9
4.2 Modeling 9
4.2.1 Deep Neural Networks 9
4.2.2 Convolutional Neural Network 9
4.2.3 Recurrent neural network 9

5 Implementation **11**
5.1 Model structure 11
5.2 Hyper parameter 12
5.2.1 Model 12

6 Costs/ Requirement **13**

7 Results **13**
7.1 Training procedure 13
7.2 Classification results 14
7.2.1 Test on Dataset 1 14
7.2.2 Test on Dataset 2 15
7.3 Split Comment Test 15
7.3.1 Test on Dataset 1 15
7.3.2 Test on Dataset 2 15

8 Evaluation **15**

9 Conclusion and future work **16**

Product Comments Sentiment Analysis

Chen Qi

Oct 2020

1 Introduction

Plenty of unstructured text data was saved when people are online shopping, such as Customer service consultation before and after sales, product comments, or comments from other social media source. If these text data could be used for convert into insight review from consumer, find potential requirements, figure out key points of Attribution conversion/non-conversion issues then customize operation strategy based on different categories customer, which can improve both on sales performance and market share.

Sentiment analysis is an useful tool on business intelligence analysis problem. It can help companies better understand all aspect of product image whether it is positive or negative. Therefore, company will be able to locate market position and identify brand advantages in a more fast and detailed method, contribute to promotion and sale spots.

We proposed 2 implementable methods for sentiment analysis on product comment, data source from TIANMAO and manual labeling. These methods have been proved effectively and efficiently on general product comment sentiment analysis.

2 Application

DITING aims to become a new business intelligence software which can increase purchase rate from AI people to PL people, find the improvement/transformation key points based on more detailed operations system.

2.1 Advantages

There are four main DITING's advantages. Firstly, it realized comprehensive analysis for pre-sale, after-sales conversations and product comment. Secondly, DITING can find the key factors in the process of purchase. Thirdly, final analysis results will be more objectivity because it own multi-features, such as

Figure 1: Comment words cloud

consultation rate, conversion rate and refund rate. Finally, DITING generate a complete solutions include consumer insight to implementation method.

2.2 Modules

DITING can be divided into 4 modules.

1. Introduction page optimization
2. Customer Services include method optimization and management system.
3. labeling system
4. Comments analysis

intelligence recommendation, multi dimension operations, customer asset analysis, label management, customer customized selection and operations track.

2.3 Sentiment Analysis Applications

The labeling library was buit in labeling system. such as people segmentation, product function category, brand and product sentiment labeling. However, positive and negative sub-comments was mixed together on the same comment sometimes, it is important to make semantic segmentation and labeling short comments for more accuracy analysis. That is means each sub-comments has it own label for next step analysis.

After received the labeled comments, our analysis tool will count the proportion of each dimension on negative and positive then provide analysis reports.

3 Methodology

The whole process is been separated into 4 parts, pre-processing, model building, training and predictions.

Raw comment will be encoded in pre-processing step and be embedding as input for the neural network model. After training the model it will be able to make classification to target value.

3.1 Data Pre-Processing

This step aims to convert the raw string type comments into numbers value to make sure computer can identify these data with words relationship information and make classification. The following shows the processing procedure in detail.

1. convert negative label into 0 and 1
2. words segmentation by 'jieba' library
3. create words vector by CountVectorizer
4. words padding to constant length
5. words embedding

3.2 Model Building

This step aims to building classification model architectures to make predictions with high accuracy on comment.

1. Recurrent Neural Network (RNN)

- Long Short Term Memory (LSTM)
- Bidirectional LSTM
- Gated Recurrent Unit(GRU)

3.3 Training

3.3.1 Data Selection

Average negative comment only take 14 percentage from all comments, which is unbalanced dataset. However, One of main targets is that figure out negative reasons and provide effective solutions. Based on this, classification model should have strong ability on negative comments detection with high accuracy. Therefore, comments which be collected for training will perform likely a uniform distribution. The Negative and Positive percentile will be adjusted respond to test accuracy.

3.3.2 Labeling Criterion

The main purpose for product comments can be classified into four situations. However, these four situations usually shows in one comment be combined in different ways.

1. Product Feedback
2. Purchase experiences
3. Expectations
4. The status after purchase

The difficult parts for comments sentiment analysis is that it has many special ways of expressing emotions or describing situations. The most confused parts be summary as following list.

1. Comments without any emotion
2. Negative and positive comments combined in the same comment
    (a) Purchased path
    (b) Different brand
3. Comparative comments
4. Purchased cost

Three classification - Negative, Positive and Neural		
comments	id	Negative label
comments	id	
comments	id	Positive label
comments	id	
comments	id	Neural label
comments	id	

Table 1: Labeling Criterion Second Version

(a) Time

(b) Price

Data labeling will be processed along with basic rules shown as following steps, which aims to improve the precision of sentiment analysis gradually.

1. Three classification - Negative, Positive and Neural
   - comments only be expressed in negative or positive way will be labeled in negative or positive.
   - comments own both negative and positive aspect will be labeled in negative.
   - comments without any emotion or not related to purchased commodity will be labeled in negative.

2. Two types of labeling

   Firstly, comments will be labeling into Available and Unavailable data. Secondly, only available data will be passed for next labeling step. Secondly, available comments will be labeled into four categories, Negative, Positive, Neural (or Nonsense) and Combine.
   - comments which is completely irrelevant to the purchased product will be labeled into Unavailable, otherwise Available.
   - comments only be expressed in negative or positive way will be labeled in negative or positive.
   - comments own both negative and positive aspect will be labeled into combine.
   - comments without any emotion or just describe the situation will be labeled into neural.

Two types of labeling			
comments	id	Available	Negative
comments	id		Positive
comments	id		Combine
comments	id	Unavailable	Neural

Table 2: Labeling Criterion First Version

Two types of labeling			
split comment$_1$	id	original comment	Negative
split comment$_2$	id		Positive
split comment$_3$	id		Neural
split comment$_4$	id		Meaningless

Table 3: Labeling Criterion Third Version

3. Short comments labels

   Long comments will be split into short comments based on certain length after Jieba segmentation. Then sentiment analysis of each short comments. This method can figure out the problem that negative and positive comments combined in one comment, additionally, it will output each objection sentiment label.

   - split original comment by specific length
   - group by continuous positive classification sentences together
   - manual labeling short sentences

3.3.3 Optimization

Future optimization will be implement in two aspect, data and model. The specific implementation is shown as following list.

1. Data
   - Increase total number of training data
   - Modify proportion of data with different labels
2. Model
   - More complex model structure
   - Fine-tuning Hyper-parameters

3.3.4 Evaluation

During training process, model will be modified based on following results. Selected model should own following abilities:

1. High accuracy both on model and negative detection
2. Validation loss
3. Coverage rate
4. Strong ability of the negative detection

3.4 Prediction

Model will be tested based on product categories, because negative comments takes different proportions in different product categories.

The best performance model will be decided by two indicators.

- TN / TN + FN
- TN / TN + FP

4 Approach

4.1 Data preprocessing

4.1.1 Words Vectors

Original corpus of Jieba cannot recognize new vocabulary and lead to wrong words segmentation. For this reason, new vocabulary should be added and result in right words segmentation. Negative corpus, blacklist corpus and commodity vocabulary were been introduced to enrich current corpus. Then flatten the whole vocabulary, it will use 1 to represent the vocabulary appear on current comment and 0 means not show up. So each comment has its own sentence vector, which is one-dimension vectors. Use these sentence vectors as input to the traditional machine learning algorithms and made classifications. However, this words vector is sparse, each comments only occupied 1% of the whole vocabulary list. Therefore, we introduce the words Embedding processing method, which generate dense vectors for calculation more effectively.

4.1.2 Segmentation length distribution

4.1.3 Padding

4.1.4 Words Embedding

Gensim is been selected for training chinese segmentation embedding training. Each vocabulary own 300 dimensions vector. such as ¿¿¿¿¿.

It maps vocabulary into a high dimension space, each vocabulary has its own location. at the same time, words embedding can calculations the similarity between different words, which is the Euclidean distance between two words.

4.2 Modeling

Recent years deep learning networks has been developed dramatically, both on application and technically aspect. such as

Attention based mechanisms – novel attention based recurrent networks

Hierarchical attention network(HAN)

deep reinforcement learning algorithms

bayesian time series modeling

Gaussian Mixture Model

Deep reinforcement learning with recurrent neural network attention based models

Historical attempts to using deep learning algorithms on sentiment analysis have been successful. Empirically, RNN performs well on natural language processing task. Because RNN can keep the information or relationships between vocabularies, words order was been taken into consideration for the whole calculations.

4.2.1 Deep Neural Networks

4.2.2 Convolutional Neural Network

4.2.3 Recurrent neural network

1. **Long Short Term Memory**

   Long short-term memory (LSTM) is one of the recurrent neural network (RNN) architecture used in the field of deep learning. It can keep words orders and word embedding information for calculation also in the long sentence. Therefore, LSTM structure is often used in natural language processing.

2. **Bidirectional Recurrent Neural Networks**

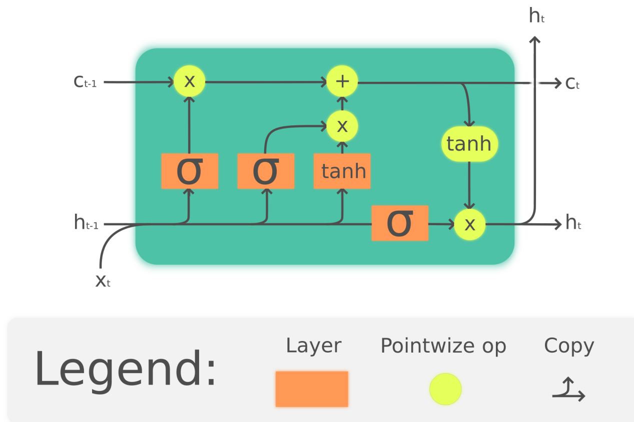

Figure 2: LSTM cell

Bidirectional Recurrent Neural Networks (BRNN) connect two hidden layers of opposite directions to the same output. BRNN layer output double units parameters.

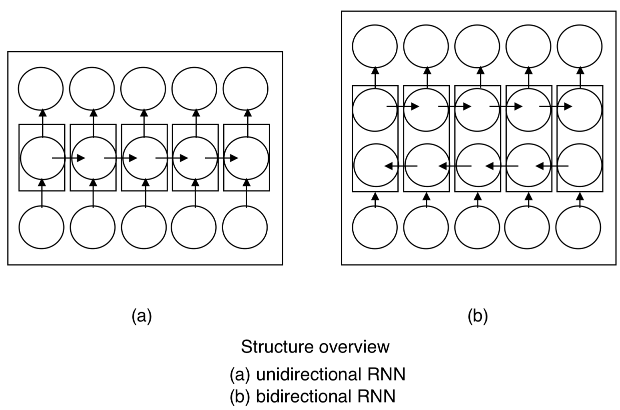

Figure 3: BRNN mechanism

3. **Gated Recurrent Unit**

   The GRU has the similar mechanism with long short-term memory but own fewer parameters than LSTM, it lacks an output gate. So the training time is shorter than the LSTM. GRUSs can perform better on specfic smaller and less frequent dataset.

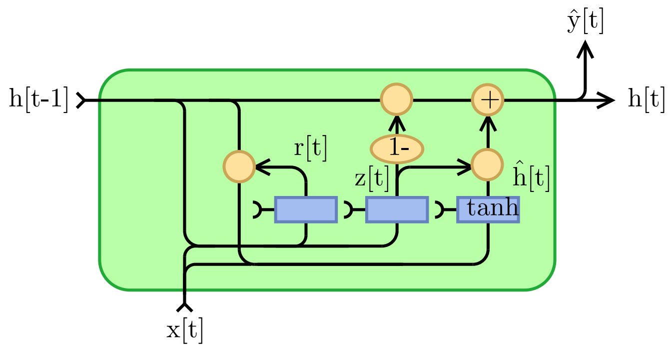

Figure 4: GRUs mechanism

5 Implementation

The LSTM was selected for for comment sentiment analysis.

5.1 Model structure

Table 4 present the best performance LSTM model structure and hyperparameters values based on the determined random seed setting.

Layer (type)	**Output Shape**	**Param**
embedding (Embedding)	(None, 87, 300)	87
bidirectional (Bidirectional)	(None, 87, 512)	1140736
$\text{lstm}_1(LSTM)$	(None, 87, 256)	787456
$\text{lstm}_2(LSTM)$	(None, 128)	197120
dropout (Dropout)	(None, 128)	0
dense (Dense)	(None, 64)	8256
$\text{dense}_1(Dense)$	(None, 1)	87
Total params: 17,133,633		
Trainable params: 2,133,633		

Table 4: Hyperparameters for final Model Structure

5.2 Hyper parameter

Table 6 describes the hyperpearameters for pretraining of stack LSTM models.

Hyperparam	**Values**
num words	50,000
max tokens	87
embedding dim	300 * 50,000
Batch size	128
Dropout	0.2
Learning Rate Decay	Linear
Optimizer	Adam (Adaptive Moment Estimation)
Loss	Binary Crossentropy
Activation	sigmoid & relu

Table 5: Hyperparameters for pretrainiing stack LSTM models

5.2.1 Model

Finetuning hyperparameters for different LSTM models are given in Table 6.

Hyperparam	**Model 1**	**Model 2**	**Model 3**	**Model 4**
Total params	15,525,697	16,477,249	16,944,705	17,133,633
Peak Learning Rate	1e-6	1e-6	1e-9	1e-8
Max Epochs	8	10	15	14

Table 6: Hyperparameters for pretraining stack LSTM models

Model Structure	
Input layer	50,000 * 300
First Bidirectional LSTM layer	256
Second LSTM layer	256
Third LSTM layer	128
Forth Dropout layer	0.2
Fifth Dense layer	64
Output layer	1
Optimizer	Adam (Adaptive Moment Estimation)
Loss	binary crossentropy
Batch size	128

Table 7: Hardware and Software requirements list

6 Costs/ Requirement

For the purpose of the high accuracy classification results, an amount of accurate labeled training and test data is necessary and play an important role for the future application. Implementation process was stated in Labeling Criterion part. Additionally, corpus related to comments should be added regularly.

Deep learning running environment require GPU to accelerate the calculations because of amount of hyper-parameters. In addition to this, whole procedure require python libraries and certain environment to dealing with data. Hardware and software requirements shown on Table 1 and Table 2.

Hardware and Software List	
Hardware Name or Software Name	version
Graphics Processing Unit(GPU)	NVIDIA GeForce RTX 2060 SUPER
Jupyter Notebook	6.0.3
Python	3.8.0
CUDA	V10.0
cuDNN	cuDNN64_7.dll

Table 8: Hardware and Software requirements list

Library List	
Library Name	version
TensorFlow	2.3.0
Keras	2.4.0
Numpy	1.18.5
pandas	1.0.5
sklearn	0.23.2
genism	3.8.3

Table 9: Python libraries requirements list

7 Results

7.1 Training procedure

During the training process, evaluation points will be key to determine whether the model preform good or not.

7.2 Classification results

There are 18037 labeled data as first training version. Negative data is 8826, positive data is 2870. some comments include both positive and negative, or do not express any emotion was labeled on neural is 6341. Then 10,000 more data was input as current training version. Negative data is 8826, positive data is 2870. some comments include both positive and negative, or do not express any emotion was labeled on neural is 6341.

Model	**Model Accuracy**	**Test Accuracy**	**TN/(TN + FN)**	**TN/RN**
Model 1	85	79.54	88.27	73.9
Model 2	79.36	79.81	81.55	84.83
Model 3	86.01	86.32	96.6	64.03
Model 4	91.31	90.8	97.48	75.81

Table 10: Development results for various configurations of Stack LSTM models

7.2.1 Test on Dataset 1

Firstly, I tried to classify all comment into 3 different categories. However, 90% predictions belong to neural. Two of the reasons lead to this result are unstructured data and lack of amount of training data. So I tried to combine neural and negative data together as negative then make binary classification. Results shown on Table 3.

Each comment length longer than 12 (Negative: 15167,Positive: 2870)	
Score Name	Results
Training Data Count	12625
Test Data Count	5412
Training epoch	10
Batch size	128
Model Accuracy	93.18%
Test Accuracy	88.51%
TN/(TN + FN)	91.34%
TN/RN	95.36

Table 11: DataSet count 18037 test results

Model Structure	
Input layer	50,000 * 300
First Bidirectional LSTM layer	256
Second LSTM layer	256
Third LSTM layer	128
Forth Dropout layer	0.2
Fifth Dense layer	64
Output layer	1
Optimizer	Adam (Adaptive Moment Estimation)
Loss	binary crossentropy
Batch size	128

Table 12: Hardware and Software requirements list

Model	**Accuracy**	**TN / TN + FN**	**TN / TN + FP**
Training Data Count	12625		
Test Data Count	5412		
Training epoch	10		
Batch size	128		
Model Accuracy	93.18%		
Test Accuracy	88.51%		
TN/(TN + FN)	91.34%		
TN/RN	95.36		

Table 13: DataSet count 18037 test results

Model	**Data**	**Batch Size**	**Step**	**Time**
Training Data Count	12625			
Test Data Count	5412			
Training epoch	10			
Batch size	128			
Model Accuracy	93.18%			
Test Accuracy	88.51%			
TN/(TN + FN)	91.34%			
TN/RN	95.36			

Table 14: DataSet count 18037 test results

7.2.2 Test on Dataset 2

7.3 Split Comment Test

7.3.1 Test on Dataset 1

7.3.2 Test on Dataset 2

8 Evaluation

LSTM tends to become forgetful in specific cases. there is no way to give more importance to some of the input words compared to other while translating the sentence.

9 Conclusion and future work

introduce Attention mechanism