



Deep Learning: Using Sentiment Classification with Natural Language Processing (NLP) to Handle Customer Feedbacks/Reviews for Online Food Business

PhD. Nguyen Hung Cuong¹ and PhD. Nguyen Duc Huy²

¹ The University of Transport Technology, Hanoi, Vietnam

² The State Council for Professorship

¹ E-mail: cuongnh@utt.edu.vn, ² E-mail: huynd@moet.edu.vn

Abstract

The online food business has a lot of potential but is also challenging due to the diversity in customers' needs and tastes. Therefore, collecting customer feedback helps businesses better understand products and customers to make accurate business decisions. However, feedback from food is often very diverse and difficult to identify due to the variety of foods, colors, shapes and feelings of consumers. However, in order to gather feedback and identify attitudes, views of customers for food products online is very difficult and costly due to the huge amount of data. Deep Learning with the ability to self-learn and self-handle complex problems can become an effective, costly and easy solution for handling customer feedback on products at e-commerce food websites. This article focuses on introducing Natural Language Processing in Deep Learning to handle huge customer feedbacks through a set of actual data from e-commerce food website, through which readers can understand and apply algorithms into your own research or business issues.

Keywords: Deep learning, Natural Language Processing, e-commerce, customer feedback.

1. INTRODUCTION

Today, with the dramatic development of technology revolution 4.0 and the explosion of the Internet has changed every aspect of human society and economy. Through the Internet people have created and exchanged a huge amount of information ever. These is become an important information gathering channel for businesses in the age 4.0, especially e-commerce businesses. One of the most important information that every business need to collect is customer feedback about its products and services. Collecting customer feedback is always necessary to help businesses understand more about their products and services, especially knowing the sentiment of customers.

However, e-commerce websites often have a huge amount of feedback/review. Moreover, feedback from food is often very diverse and difficult to identify due to the variety of foods, colors, shapes and feelings of consumers. Due to the complexity and large amount of data, handling customer feedback/review on online food processed by hand to know the sentiment of customers is very costly and time consuming. Natural Language Processing (NLP) is a branch of artificial intelligence that deals with the interaction between computers and humans using the natural language. (Michael J. Garbade, 2018) Sentiment classification with NPL in Deep Learning can be a good solution for complex data processing of this type. Using Deep Learning will help online food business websites handle customer feedback easily, quickly and economically compared to previous traditional methods.

The screenshot displays the Amazon product page for '10 (6 oz.) Filet Mignon Steaks' by 'Feed The Party'. The product is marked as 'Amazon's Choice for "filet"'. It features a large image of a cooked steak and a grid of smaller images showing different views. Below the main image, there is a section titled 'About the product' with a list of features: 6 oz USDA Filet Mignon Steaks (Individually Sealed), Custom Packaging* - Dry Ice Cooler Shipped Right To Your Door, Finest USDA Inspected Beef, Hand Cut, Thick and Juicy, and Hand Cut in Louisville, Kentucky Since 1946. To the right, the 'Showing 1-8 of 106 reviews' section is visible, showing two reviews. The first review is by 'Daniel T S' from April 16, 2017, with a 5-star rating and a 'Verified Purchase' badge. The review text describes the steaks as 'Nice small steaks. Tasty and tender. Some of them are a little odd shaped but cook up great. I pan fried them in butter to seal them and baked them for 5 minutes at 400 degrees. Perfect Medium rare and tender.' and includes a warning: 'My wife stuck them under the broiler and cooked them well done to shoe leather tough, dry and more like jerky. Don't do it.' The second review is by '-jdh' from December 5, 2017, also with a 5-star rating and 'Verified Purchase' badge. The review text says: 'Can't go wrong with these filets. Perfectly trimmed, and a steal at this price. Had a problem with FedEx delivery on my first order, Feed the Party's customer service took care of it immediately. Could not be happier. HIGHLY recommended'.

Figure 1: Customer feedbacks about a product.

Source: www.amazon.com/10-oz-Filet-Mignon-Steaks/

This article studies the use of Sentiment Classification with Natural Language Processing on **Gated Recurrent Unit (GRU)** algorithm in **Deep Learning** to handle customer feedback/review, in Python programming language and programming library: Numpy, Pandas, Matplotlib, Scikit-learn, Tensor flow through a processing reality Amazone's Data.

2. GATED RECURRENT UNITS AND THE DATASET

2.1. Artificial Intelligence, Machine Learning and Deep Learning

Artificial intelligence can be interpreted as an intelligence represented by any artificial system, programmed by humans with the goal of enabling computers to automate intelligent behaviors like humans. (Smola & Vishwanathan, 2008)

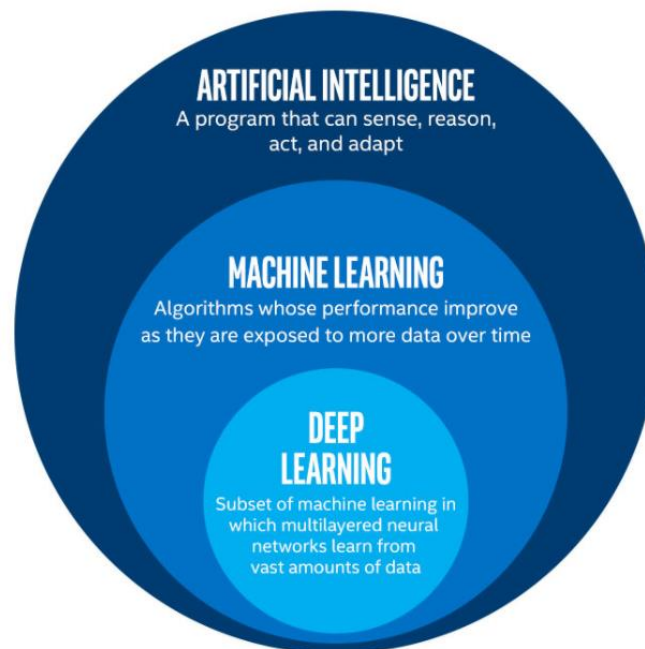


Figure 1: Cousins of AI

Source: <https://towardsdatascience.com/cousins-of-artificial-intelligence-dda4edc27b55>

Machine learning is a field of computer science, in which algorithms or systems are designed to learn from collected data, often for the purpose of predicting results from input data that cannot be seen (the amount of data is so large that users often can't see it all). (Shalev-Shwartz & Ben-David, 2014)

Machine Learning is divided into 3 categories:

- Supervised Machine Learning
- Unsupervised Machine Learning
- Reinforcement Machine Learning

This article is focused on using Deep learning.

Deep learning is part of machine learning methods based on artificial neural networks. Can understand Deep Learning is Neural Network with many Hidden Layer. A basic ANN can have several Hidden Layers, but with Deep Learning, that number can reach hundreds, thousands or millions of layers.

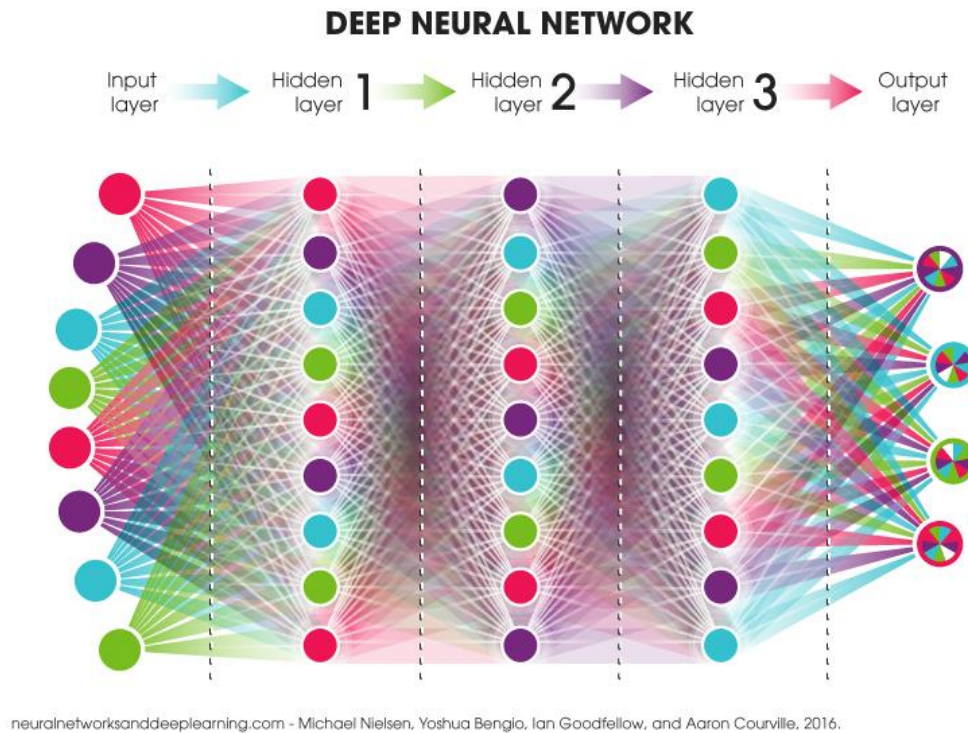


Figure 3: Deep neural network

Deep learning focuses on solving problems related to artificial neural networks to apply to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation... Deep learning architectures include such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks.

2.2. Recurrent Neural Network

A recurrent neural network (RNN) is a class of artificial neural network where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. (*wikipedia.org, 2019*) Put simply, it is an artificial neural network that contains loops inside it. Recurrent Neural Network is very suitable for handling input sequences. (*Shai Shalev-Shwartz & Shai Ben-David, 2014*)

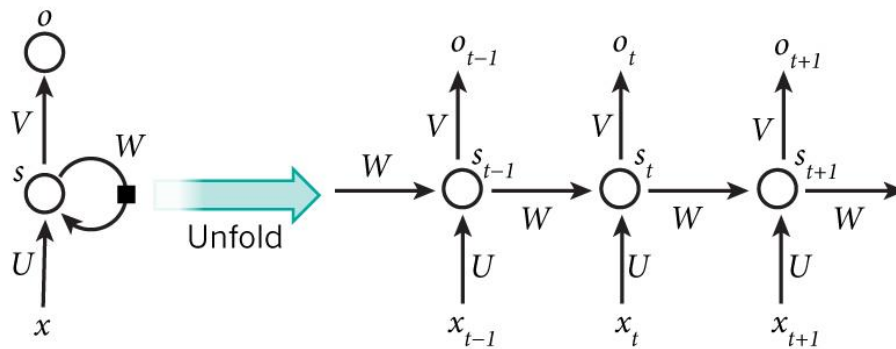


Figure 4: A recurrent neural network and the unfolding

Source: Nature

- The calculation of the Recurrent Neural Network is as follows:
- x_t is the input at step t .
- s_t is the hidden state at step t . It is the memory of the network. s_t is calculated based on both the previous hidden states and the input at that step: $s_t = f(U \cdot x_t + W \cdot s_{t-1})$. Function f is usually a non-linear function like hyperbolic (\tanh) or [ReLU](#).
- o_t is the output at step t .

2.3. Gated Recurrent Unit

Gated recurrent unit (GRU) aiming to solve the vanishing gradient problem which comes with a standard recurrent neural network. (Kyunghyun Cho & et al., 2014) GRU uses the so called, update gate and reset gate. The Sigma notation above represent those gates: which allows a GRU to carry forward information over many time periods in order to influence a future time period. In other words, the value is stored in memory for a certain amount of time and at a critical point pulling that value out and using it with the current state to update at a future date. (Georgios Drakos, 2019)

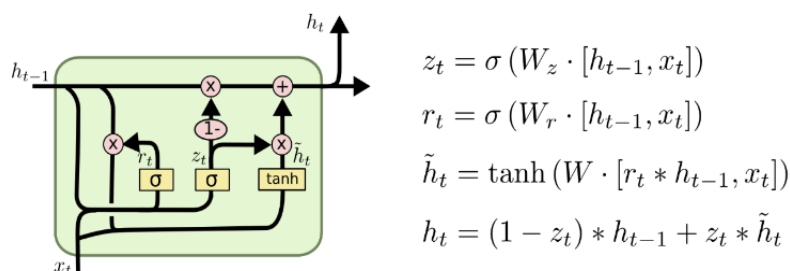


Figure 5: A Gated recurrent unit

Source: towardsdatascience.com/what-is-a-recurrent-nns-and-gated-recurrent-unit-gru

A common GRU unit component:

z_i : update gate vector

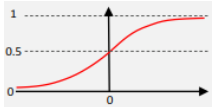
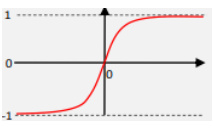
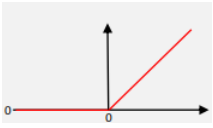
r_i : Reset gate vector

h_i : output vector

x_i : input vector

σ : The original is a sigmoid function.

Table 1: Activation functions

Activation functions	Formula	Derivative	Graph	Results domain
Sigmoid (logistic function)	$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}}$	$\frac{\partial a(x)}{\partial x} = \partial(x)(1 - \partial a(x))$		(0,1)
Tanh (hyperbolic tangent)	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$\frac{\partial \tanh(x)}{\partial x} = \frac{1}{(e^x + e^{-x})^2}$		(-1,1)
ReLU (rectified linear unit)	$\text{relu}(x) = \max(0, x)$	$\frac{\partial \text{relu}(x)}{\partial x} = \begin{cases} 0, & \text{if } x \leq 0 \\ 1, & \text{if } x > 0 \end{cases}$		(0,∞)

Source: (Shalev-Shwartz & Ben-David, 2014)

2.4. Natural Language Processing.

Natural Language Processing (NLP) is a branch of artificial intelligence that deals with the interaction between computers and humans using the natural language. The ultimate objective of NLP is to read, decipher, understand, and make sense of the human languages in a manner that is valuable. Most NLP techniques rely on machine learning to derive meaning from human languages. (Michael J. Garbade, 2018)

NLP entails applying algorithms to identify and extract the natural language rules such that the unstructured language data is converted into a form that computers can understand. When the text has been provided, the computer will utilize algorithms to extract meaning associated with every sentence and collect the essential data from them. (Michael J. Garbade, 2018)

NLP and text analytics are used together for many applications, including:

- Investigative discovery. Identify patterns and clues in emails or written reports to help detect and solve crimes.

- Subject-matter expertise. Classify content into meaningful topics so you can take action and discover trends.
- Social media analytics. Track awareness and sentiment about specific topics and identify key influencers.

Sentiment Classification

Sentiment analysis (also known as opinion mining or emotion AI) refers to the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine. (wikipedia, 2019)

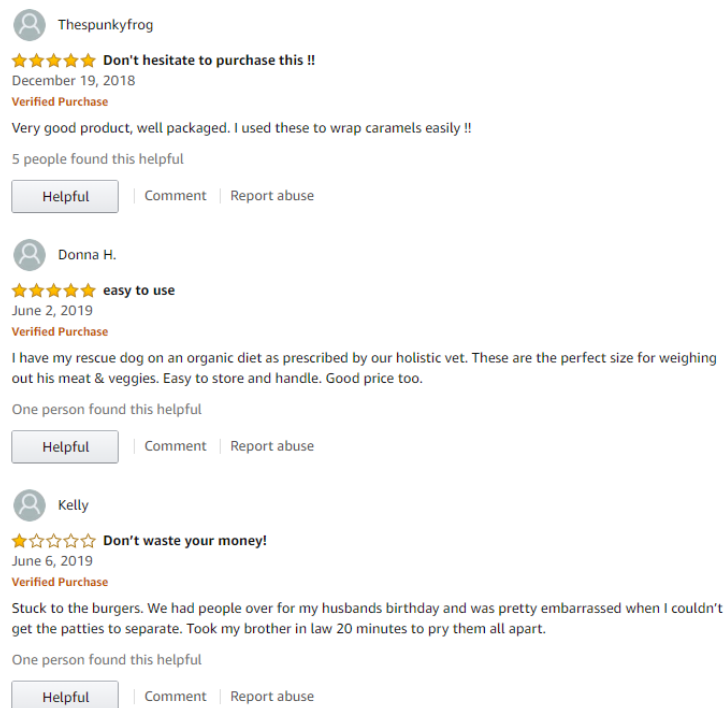


Figure 6: review and rating of customers.

Source: www.amazon.com/Restaurant-Grade-Hamburger-Avant-Non-Stick-Food-Grade/



2.5. The Dataset

This paper uses an Amazon's food feedbacks or reviews dataset revolving around the reviews written by customers. The data span a period of more than 10 years, including all 568454 reviews up to October 2012. This is the real dataset so the Text data is written in natural language. With a large sample, natural language, the dataset is very suitable to use NLP analysis in Deep Learning.

Data fields include:

- Productid: Unique identifier for the product
- Userid: Unique identifier for the user
- ProfileName: Profile name of the user
- HelpfulnessNumerator: Number of users who found the review helpful
- HelpfulnessDenominator: Number of users who indicated whether they found the review helpful or not
- Score: Positive Ordinal Integer variable for the product score granted by the customer from 1 Worst, to 5 Best.
- Time: Timestamp for the review
- Summary: Brief summary of the review
- Text: Text of the review

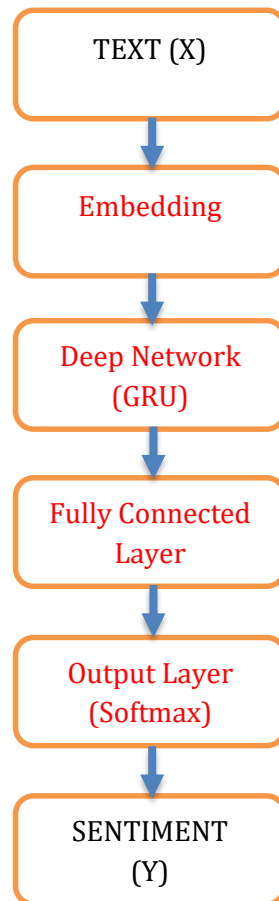
Source: <https://www.kaggle.com/snap/amazon-fine-food-reviews>

Sentiment classification is the task of looking at a piece of text and telling if someone likes or dislikes the thing they're talking about. (Nabi, 2018)

The input is a piece of text and the output is the sentiment which we want to predict, such as the rating or score of a review.

2.6. Deep Learning Architecture

Deep learning text classification model architectures generally consist of the following components connected in sequence:



Embedding Layer

Word Embedding is a representation of text where words that have the same meaning have a similar representation. In other word, it represents words in a coordinate system where related words, based on a corpus of relationships, are placed closer together.

Deep Network

Deep network takes the sequence of embedding vectors as input and converts them to a compressed representation. The compressed representation effectively captures all the information in the sequence of words in the text.

Fully Connected Layer

The fully connected layer takes the deep representation from the GRU and transforms it into the final output classes or class scores. This component is comprised of fully connected layers along with batch normalization and optionally dropout layers for regularization.

Output Layer

Based on the problem at hand, this layer can have Softmax for multi classification output.

3. BUILDING MODEL

Using Jupyter programming editor software to build model:

1. Import Libraries and The Dataset

1.1. Import Libraries

```
[1]: import numpy as np
import pandas as pd
import sqlite3

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set()

import math
import warnings
import itertools
warnings.filterwarnings('ignore')
```

1.2. Import The Dataset

```
[2]: con = sqlite3.connect('database.sqlite')
data = pd.read_sql_query("SELECT * from Reviews", con)
data.head()
```

[2]:	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVESNK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCHO	ABXLMWJ0XXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A3958ORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient L...
4	5	B006K2ZZ7K	A1UQRSCLF8W1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

```
[3]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 568454 entries, 0 to 568453
Data columns (total 10 columns):
Id                568454 non-null int64
ProductId         568454 non-null object
UserId           568454 non-null object
ProfileName       568454 non-null object
HelpfulnessNumerator  568454 non-null int64
HelpfulnessDenominator  568454 non-null int64
Score             568454 non-null int64
Time             568454 non-null int64
Summary          568454 non-null object
Text             568454 non-null object
dtypes: int64(5), object(5)
memory usage: 43.4+ MB
```

2. Preprocessing Data

2.1. Label encoding

```
[4]: from sklearn.preprocessing import LabelEncoder
labelencoder_X = LabelEncoder()
data['Score'] = labelencoder_X.fit_transform(data['Score'])
data.head()
```

[4]:	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A35GXH7AUHU8GW	delmartian	1	1	4	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	0	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	3	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	1	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	4	1350777600	Great taffy	Great taffy at a great price. There was a wid...

2.2. Processing text values ¶

```
[5]: # Processing text: RAW data
print("Text to seq process")
from tensorflow.keras.preprocessing.text import Tokenizer
raw_text = np.hstack([data['Summary'].str.lower(), data['Text'].str.lower()])

print(" Fitting tokenizer...")
tok_raw = Tokenizer()
tok_raw.fit_on_texts(raw_text)
print(" Transforming text to seq...")

data["seq_Summary"] = tok_raw.texts_to_sequences(data['Summary'].str.lower())
data["seq_Text"] = tok_raw.texts_to_sequences(data['Text'].str.lower())
data.head(5)
```

```
Text to seq process
Fitting tokenizer...
Transforming text to seq...
```

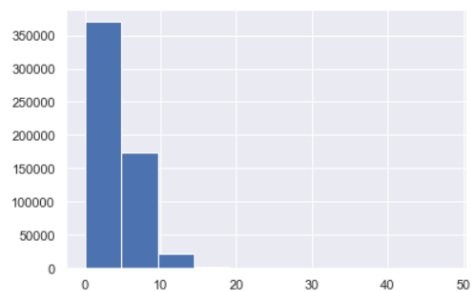
[5]:	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text	seq_Summary	seq_Text
0	1	B001E4KFG0	A35GXH7AUHU8GW	delmartian	1	1	4	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...	[27, 162, 85, 47]	[2, 18, 128, 332, 7, 1, 4549, 511, 85, 47, 212, ...]
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	0	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...	[16, 23, 1359]	[34, 384, 2269, 23, 5400, 1964, 1057, 1, 1057, ...]
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	3	1219017600	"Delight" says it all	This is a confection that has been around a fe...	[1740, 520, 6, 41]	[9, 8, 4, 7084, 14, 53, 88, 281, 4, 177, 9574, ...]
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	1	1307923200	Cough Medicine	If you are looking for the secret ingredient i...	[2805, 1878]	[40, 19, 20, 256, 11, 1, 2470, 584, 12, 23609, ...]
4	5	B006K2ZZ7K	A1UQRSLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	4	1350777600	Great taffy	Great taffy at a great price. There was a wid...	[29, 3311]	[29, 3311, 35, 4, 29, 76, 81, 21, 4, 2144, 200, ...]

```
[6]: #SEQUENCES VARIABLES ANALYSIS
max_seq_Summary = np.max([np.max(data.seq_Summary.apply(lambda x: len(x)))])
max_seq_Text = np.max([np.max(data.seq_Text.apply(lambda x: len(x)))])
print("max Summary seq "+str(max_seq_Summary))
print("max Text seq "+str(max_seq_Text))
```

```
max Summary seq 48
max Text seq 3507
```

```
[7]: data.seq_Summary.apply(lambda x: len(x)).hist()
```

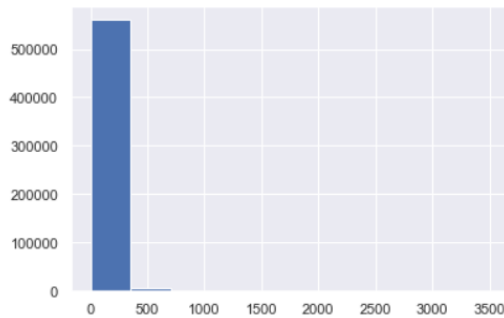
```
[7]: <matplotlib.axes._subplots.AxesSubplot at 0x1bb6ceaa90>
```





```
[8]: data.seq_Text.apply(lambda x: len(x)).hist()
```

```
[8]: <matplotlib.axes._subplots.AxesSubplot at 0x1bb062250b8>
```



```
[10]: #EMBEDDINGS MAX VALUE
#Base on the histograms, we select the next lengths
MAX_seq_Summary = 12
MAX_seq_Text = 200
MAX_TEXT = np.max([np.max(data.seq_Summary.max()), np.max(data.seq_Text.max())]+2
#MAX_Division = np.max([data['Division Name'].max()])+1
#MAX_Department = np.max([data['Department Name'].max()])+1
#MAX_Class = np.max([data['Class Name'].max()])+1
```

```
[11]: data["Score"].unique()
```

```
[11]: array([4, 0, 3, 1, 2], dtype=int64)
```

2.3. Slipting The Dataset

```
[12]: #EXTRACT DEVELOPMENT TEST
dtrain, dvalid = train_test_split(data, random_state=123, train_size=0.8)
print(dtrain.shape)
print(dvalid.shape)

(454763, 12)
(113691, 12)
```

2.4. Keras data definition

```
[13]: from tensorflow.keras.preprocessing.sequence import pad_sequences

def get_keras_data(dataset):
    X = {
        'Summary': pad_sequences(dataset.seq_Summary, maxlen=MAX_seq_Summary)
        , 'Text': pad_sequences(dataset.seq_Text, maxlen=MAX_seq_Text)
    }
    return X

X_train = get_keras_data(dtrain)
X_valid = get_keras_data(dvalid)
```

```
[14]: import tensorflow as tf
y_train = tf.keras.utils.to_categorical(dtrain['Score'], 5)
y_valid = tf.keras.utils.to_categorical(dvalid['Score'], 5)
```



3. Buiding the Model

3.1. Definiting The Molde

```
[15]: from tensorflow.keras.layers import Input, Dropout, Dense
from tensorflow.keras.layers import Activation, concatenate, GRU, Embedding
from tensorflow.keras.models import Model
from tensorflow.keras.callbacks import Callback, EarlyStopping
from tensorflow.keras import backend as K

def get_model():
    # Params
    dr_r = 0.2

    #Inputs
    Summary = Input(shape=[X_train["Summary"].shape[1]], name="Summary")
    Text = Input(shape=[X_train["Text"].shape[1]], name="Text")

    #Embeddings Layers
    emb_Summary = Embedding(MAX_TEXT, 20)(Summary)
    emb_Text = Embedding(MAX_TEXT, 80)(Text)

    #rnn Layer

    rnn_layer1 = GRU(200, activation='relu')(emb_Text)
    rnn_layer2 = GRU(100, activation='relu')(emb_Summary)

    main_l = concatenate([ rnn_layer1, rnn_layer2])

    main_l = Dropout(dr_r) (Dense(200, activation='relu')(main_l))
    main_l = Dropout(dr_r) (Dense(100, activation='relu')(main_l))
    main_l = Dropout(dr_r) (Dense(50)(main_l))

    #output
    output = Dense(5, activation="softmax")(main_l)

    #model
    model = Model([Summary, Text], output)
    model.compile(loss='categorical_crossentropy', optimizer="adam", metrics=['accuracy'])

    return model

model = get_model()
model.summary()
```

```
Model: "model"
```

Layer (type)	Output Shape	Param #	Connected to
Text (InputLayer)	[(None, 200)]	0	
Summary (InputLayer)	[(None, 12)]	0	
embedding_1 (Embedding)	(None, 200, 80)	11154160	Text[0][0]
embedding (Embedding)	(None, 12, 20)	2788540	Summary[0][0]
unified_gru (UnifiedGRU)	(None, 200)	169200	embedding_1[0][0]
unified_gru_1 (UnifiedGRU)	(None, 100)	36600	embedding[0][0]
concatenate (Concatenate)	(None, 300)	0	unified_gru[0][0] unified_gru_1[0][0]
dense (Dense)	(None, 200)	60200	concatenate[0][0]
dropout (Dropout)	(None, 200)	0	dense[0][0]
dense_1 (Dense)	(None, 100)	20100	dropout[0][0]
dropout_1 (Dropout)	(None, 100)	0	dense_1[0][0]
dense_2 (Dense)	(None, 500)	50500	dropout_1[0][0]
dropout_2 (Dropout)	(None, 500)	0	dense_2[0][0]
dense_3 (Dense)	(None, 5)	2505	dropout_2[0][0]

```

Total params: 14,281,805
Trainable params: 14,281,805
Non-trainable params: 0
    
```

The network has 7 neural layers, 14,281,805 params.

3.2. Fitting The Model

```
[16]: BATCH_SIZE = 1000
epochs = 5

# Define early stopping callback
my_callback = [EarlyStopping(monitor='val_accuracy', patience=1, verbose = 1)]
model = get_model()

[17]: model.fit(X_train, y_train, epochs=epochs, batch_size=BATCH_SIZE
              , validation_data=(X_valid, y_valid), callbacks = my_callback
              , verbose=1)

Epoch 1/5
454763/454763 [=====] - 830s 2ms/sample - loss: 0.7871 - accuracy: 0.7045 - val_loss: 0.6318 - val_accuracy: 0.7562
Epoch 2/5
454763/454763 [=====] - 826s 2ms/sample - loss: 0.5601 - accuracy: 0.7861 - val_loss: 0.5544 - val_accuracy: 0.7890
Epoch 3/5
454763/454763 [=====] - 827s 2ms/sample - loss: 0.4628 - accuracy: 0.8276 - val_loss: 0.5397 - val_accuracy: 0.8020
Epoch 4/5
454763/454763 [=====] - 830s 2ms/sample - loss: 0.3914 - accuracy: 0.8573 - val_loss: 0.5501 - val_accuracy: 0.8130
Epoch 5/5
454763/454763 [=====] - 829s 2ms/sample - loss: 0.3320 - accuracy: 0.8808 - val_loss: 0.5909 - val_accuracy: 0.8128
Epoch 00005: early stopping
[17]: <tensorflow.python.keras.callbacks.History at 0x1bb066e4f98>
```

The accuracy of the Model after 5 epochs training is 88.08%. However, with the test dataset, val_accuracy is only 81.21%.



4. DICUSSION

In order to assess the predictability and application of the forecast Model in fact the confusion matrix is determined:

4. Evaluaeting The Model

```
[18]: # compute predictions
y_pred = model.predict(X_valid)
y_pred = np.array(y_pred)

[19]: from sklearn.metrics import classification_report, confusion_matrix

y_valid_class = np.argmax(y_valid, axis=1)
y_pred_class = np.argmax(y_pred, axis=1)

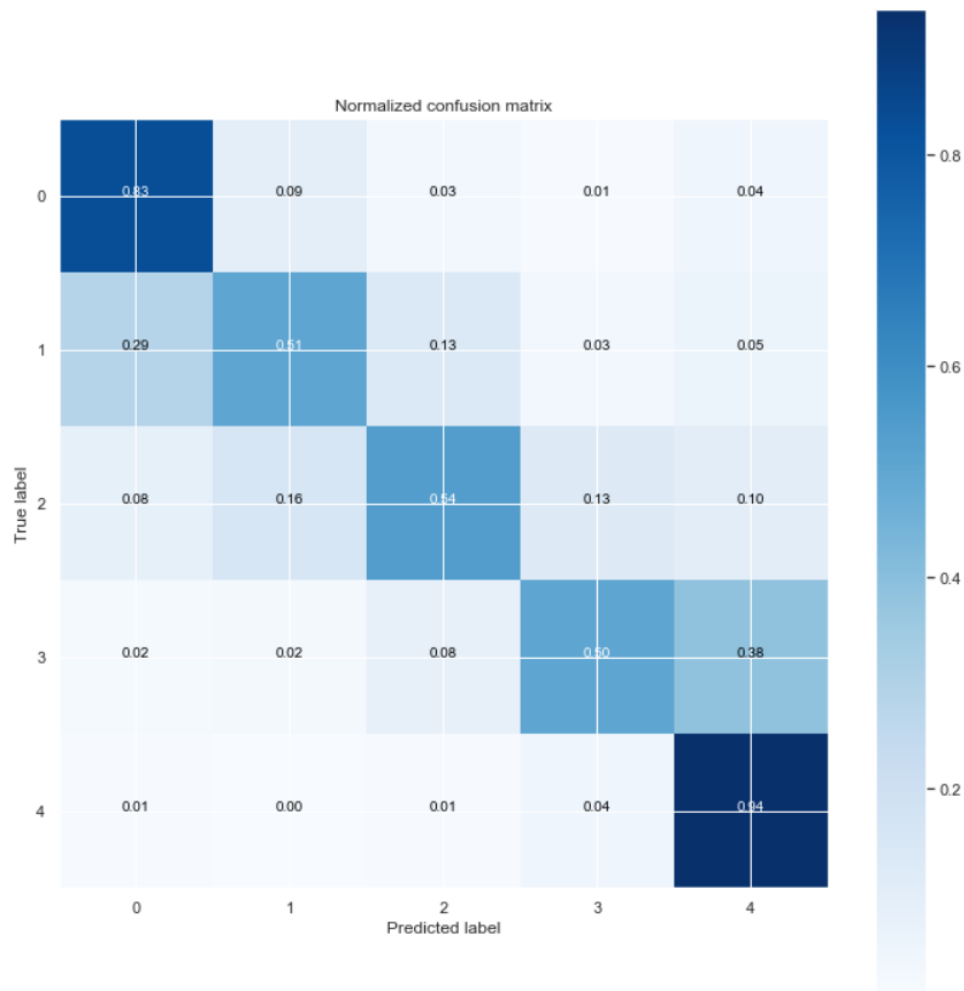
[21]: def plot_confusion_matrix(cm, title='Confusion matrix', cmap=plt.cm.Blues):
cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
plt.figure(figsize=(10,10))
plt.imshow(cm, interpolation='nearest', cmap=cmap)
plt.title(title)
plt.colorbar()
fmt = '.2f'
thresh = cm.max() / 2.
for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
    plt.text(j, i, format(cm[i, j], fmt),
             horizontalalignment="center",
             color="white" if cm[i, j] > thresh else "black")

plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.tight_layout()

[22]: # show the confusion matrix of our predictions

plt.figure()
plot_confusion_matrix(cm, title='Normalized confusion matrix')
plt.show()

<Figure size 432x288 with 0 Axes>
```



Scoring satisfaction research of the Dataset has is: 0-very poor, 1- poor, 2-fair, 3-good, 4-very good.
 The Confusion matrix shows:

- The model that accurately predicts the 4-very good feedbacks is 94%, the worst false prediction to 3-good feedbacks is only 4%. The cause of high accuracy in prediction because the language of this customer feedback is often clear and recognizable.
- The most inaccurate predicting model is the 3-good feedbacks of only 50%, the worst false prediction to 4-very good feedbacks is 38%. The reason of high inaccuracy is the confusion in natural language between 3-good and 4-very good feedbacks of the customers.
- The second accurate predicting model is the 1-very poor feedbacks of 83%, the worst false prediction to 1-poor feedbacks is 9%. The cause of high accuracy like predicting 4-very good feedbacks, language of this customer feedback is often clear and recognizable.
- The 2-fair feedbacks prediction has an accurate prediction of 54%, the worst false prediction to 1-poor feedbacks is 16%. The 1-poor feedbacks prediction has an accurate



prediction of 51%, the worst false prediction to 0-very poor feedbacks is 29%. The cause of false predictions is due to the low number of samples, the easy to confuse the two consecutive feelings and the complexity of using natural language when expressing their satisfaction.

The model has an average accuracy of 81.1% which shows that this is a good forecasting model. The forecast errors only occur in 2 pairs of consecutive satisfaction because it is difficult to clearly identify even for the person making the review. The model has very little false predictions between negative and positive feedbacks and vice versa. Therefore, this is a model with high applicability in practice. Furthermore, continuing to collect customer feedback will increase the database, which will make the model training more accurate. The application of artificial intelligence in handling customer feedback will help the food website: reduce costs and labor, keeping up with customers' needs; make the right business marketing decisions for each product; improve business efficiency. Thereby improving competitiveness and expanding the market of businesses in a sustainable way.

5. CONCLUSION

In the context of the ongoing industrial revolution 4.0, it has promoted the application of scientific and technological advances in all aspects of business. The online food business has a lot of potential but is also challenging due to the diversity in customers' needs and tastes. The use of Artificial Intelligence helps businesses keep up with the trend and increase customer satisfaction thanks to its outstanding and quick processing capabilities. This article focuses on introducing GRU algorithms in Natural Language Processing of Deep Learning to classify feedbacks of online customers. This model can also be expanded to predict what customers are satisfied with and not satisfied with each garment product (design, color or material ...), from which the business has correct adjustments and timely. When customers are heard and satisfied with the website, they will increase their competitiveness, reduce business risks and maximize profits.



REFERENCES

1. Alex Smola, & S.V.N. Vishwanathan. (2008). *INTRODUCTION TO MACHINE LEARNING*. Cambridge University Press.
2. Edouard Duchesnayt, & Tommy Löfsted. (2018). *Statistics and Machine Learning in Python - Release 0.2*. <ftp://ftp.cea.fr>.
3. Frank Rosenblatt. (1961). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Washington DC: Spartan Books.
4. Georgios Drakos. (2019). *What is a Recurrent Neural Networks (RNNS) and Gated Recurrent Unit (GRUS)*. <https://towardsdatascience.com/what-is-a-recurrent-nns-and-gated-recurrent-unit-grus>
5. Guido van Rossum, & et al. (2018). *The Python Language Reference - Release 3.7.1*. Python Software Foundation.
6. Jürgen Schmidhuber. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85-117.
7. Kyunghyun Cho, & et al. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. [arxiv.org](https://arxiv.org/abs/1406.2661).
8. Michael J. Garbade. (2018). *A Simple Introduction to Natural Language Processing*. <https://becominghuman.ai/a-simple-introduction-to-natural-language-processing-ea66a1747b32>
9. NumPy community. (2018). *NumPy Reference - Release 1.15.4*. [SciPy.org](https://docs.scipy.org/doc/numPy).
10. Scikit-learn developers. (2018). *scikit-learn user guide - Release 0.20.1*. scikit-learn.org.
11. Sepp Hochreiter, & Jürgen Schmidhuber. (1997). *Long Short-Term Memory* (Volume 9, Issue 8). Massachusetts Institute of Technology.
12. Shai Shalev-Shwartz, & Shai Ben-David. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.
13. Shashi Sathyanarayana. (2014). *A Gentle Introduction to Backpropagation*. Numeric Insight, Inc Whitepaper.
14. Wes McKinney & et al. (2018). *pandas: powerful Python data analysis toolkit - Release 0.23.4*. PyData Development Team.