Project Final Report:
Data mining on E-commerce platform

Group 1

1. Problem to Solve

Online shops often face difficulties in managing and optimizing their business activities from their business data.. The main issues include:

- **Product Improvement:** Adjusting and enhancing product quality based on customer feedback and shops' data.
- **Understanding Customer Needs:** Recognizing customer trends and demands to optimize business strategies.

Manual analysis can be time-wasting and inefficient. Therefore, we propose a system to process and analyze shops' data, to help the shops utilize insight from their product to improve sales performance.

2. Data

- **Data Source:** We collected data from 11 sample Fashion shops on Shopee by implementing a web scraping technique. The data includes: 3 tables which is Shop, Product and Customer reviews.
- **Data Size:** Our dataset currently stands at **24710** raw customer reviews after scraping 11 shops on 124 products, each product will be collected 200-500 reviews from 1-5 star ratings based on the number of reviews on its product, encompassing a diverse range of products and customer opinions. We are not currently applying to scrape images or videos in review yet, also respond to the shop on customer reviews and shop contact information as well. But there will be developed in the future for further purposes in analyzing and functioning to support customers from the product owner.
- **Data Problem :**
    - Scraping data is not automated but still has to do it by hand. We'll run javascript on the console page URL to get html.content and send it to the Flask server and scrape it. Each page gives us max 6 reviews and kind of takes time to scrape huge amounts of data.
    - Customer account (name): some names are defined as "t*****s".It is hard to summarize which customer comment most, don't figure out how we got that issues => solution: Create a customer ID unique for each customer, it will work to replace the customer name if this "t*****s" has not been generated by shopee
    - Data type: some of integer column has "k" in their data such as

Total_Rat	Total_Rat	Total_Rev	Total_sold	Res
61,8k	4.8	2,6k	6,4k	
61,8k	4.8	2,6k	6,4k	
61,8k	4.8	2,6k	6,4k	
61,8k	4.8	2,6k	6,4k	

2.2 Data type wrong for integer columns

lead to hard to visualize and calculated => Solution: each k stands for *1000, 6,4k is shortcut for 6400, so apply this calculated on code to change data type and it format

- When scraping an empty review, we have a problem in that the review information will be replaced as these texts:

h	p*****5	hữu ích?báo cáo
h	c*****5	1báo cáo
h	maiemmaie	hữu ích?báo cáo
h	5ohzlzvknq	phản hồi của Người BánDạ shop cảm ơn góp ý của bạ
h	pntuyen191	hữu ích?báo cáo
h	haluan2001	hữu ích?báo cáo
h	s55556	hữu ích?báo cáo
h	ng_anh73.q	hữu ích?báo cáo
h	lththutrinh1	hữu ích?báo cáo
h	thngnmai	hữu ích?báo cáo
h	tquangloc6(	hữu ích?báo cáo

2.3 Text return if there is an empty reviews scraped

- And if it only contains videos, no text reviews it will end up like this :

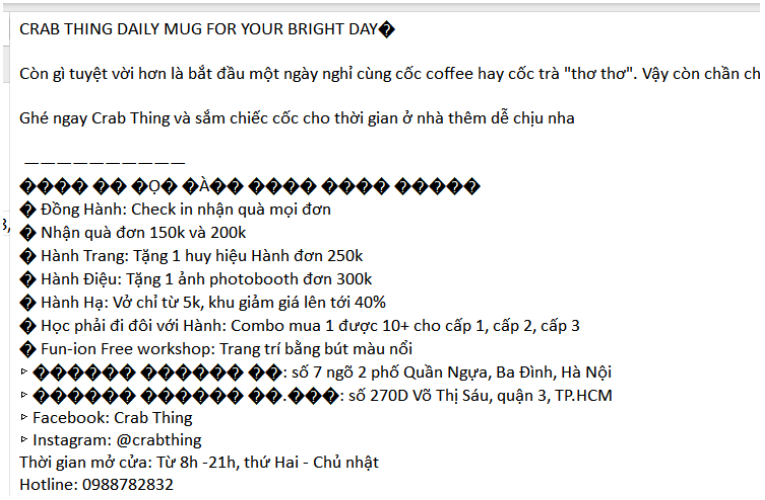
ng	Ten_khac	Thong_tin_Revie	S
ại	99minhqua	0:08	
ại	lngvn787	0:39	
ại	phamttvy	0:04	
ại	kjgs_ko8h	0:11	
ại	vinacoen	0:04	

2.4 Reviews is video duration if empty text in reviews scraped

- Reviews contain spam words, for advertisements, and other topics :

Bsjs akd q jx ẹc kx. Snc na x sk cndns nx.do b xó c	
Còn lại đều ổn	
3dwgvvpvxi	yyhhhhhhhhhjjjigghhhhhhhhhvfftyyhhggfgtyh
bihungha	1báo cáo
phamquang	Ok

2.5 Reviews contain random words



CRAB THING DAILY MUG FOR YOUR BRIGHT DAY

Còn gì tuyệt vời hơn là bắt đầu một ngày nghỉ cùng cốc coffee hay cốc trà "thơ thơ". Vậy còn chần ch

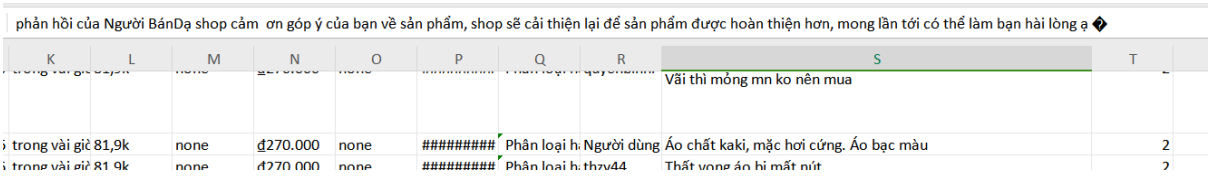
Ghé ngay Crab Thing và sắm chiếc cốc cho thời gian ở nhà thêm dễ chịu nha

——————————

Đồng Hành: Check in nhận quà mọi đơn
Nhận quà đơn 150k và 200k
Hành Trang: Tặng 1 huy hiệu Hành đơn 250k
Hành Điệu: Tặng 1 ảnh photobooth đơn 300k
Hành Hạ: Vở chỉ từ 5k, khu giảm giá lên tới 40%
Học phải đi đôi với Hành: Combo mua 1 được 10+ cho cấp 1, cấp 2, cấp 3
Fun-ion Free workshop: Trang trí bằng bút màu nổi
▹ : số 7 ngõ 2 phố Quần Ngựa, Ba Đình, Hà Nội
▹ : số 270D Võ Thị Sáu, quận 3, TP.HCM
▹ Facebook: Crab Thing
▹ Instagram: @crabthing
Thời gian mở cửa: Từ 8h -21h, thứ Hai - Chủ nhật
Hotline: 0988782832

2.6 Advertisement in reviews

- Reviews contain responses of the shop to customer reviews accidentally. It contains only shop responses so we can't get anything from it, if there are reviews that come along, there will be ideas for it in the future, like automated responses after classifier reviews type…



phản hồi của Người BánDạ shop cảm ơn góp ý của bạn về sản phẩm, shop sẽ cải thiện lại để sản phẩm được hoàn thiện hơn, mong lần tới có thể làm bạn hài lòng ạ

K	L	M	N	O	P	Q	R	S	T
								Vãi thì mỏng mn ko nên mua	
trong vài giờ	81,9k	none	₫270.000	none	#########	Phân loại h	Người dùng	Áo chất kaki, mặc hơi cứng. Áo bạc màu	2
trong vài giờ	81,9k	none	₫270.000	none	#########	Phân loại h	thzv44	Thất vọng áo bị mất nút	2

2.7 Responses of the shop in customer reviews

- Duplicate reviews, customerID : both will be drop without any troubles, because customerID is generated, even though they duplicated but remains columns still have their information, just the id is duplicate, it's more about code than scrape tools.

₫350.000 - ₫199.000 -	43% giảm	#########	Phân loại hà	nguyenchinl	Chất liệu: kà	2	15064
₫350.000 - ₫199.000 -	43% giảm	#########	Phân loại hà	nguyenchinl	Chất liệu: kà	2	1

2.8 Duplicated with customerID = 1

- Reviews have errors in some formats when instead of going down the line, the words are stuck together.

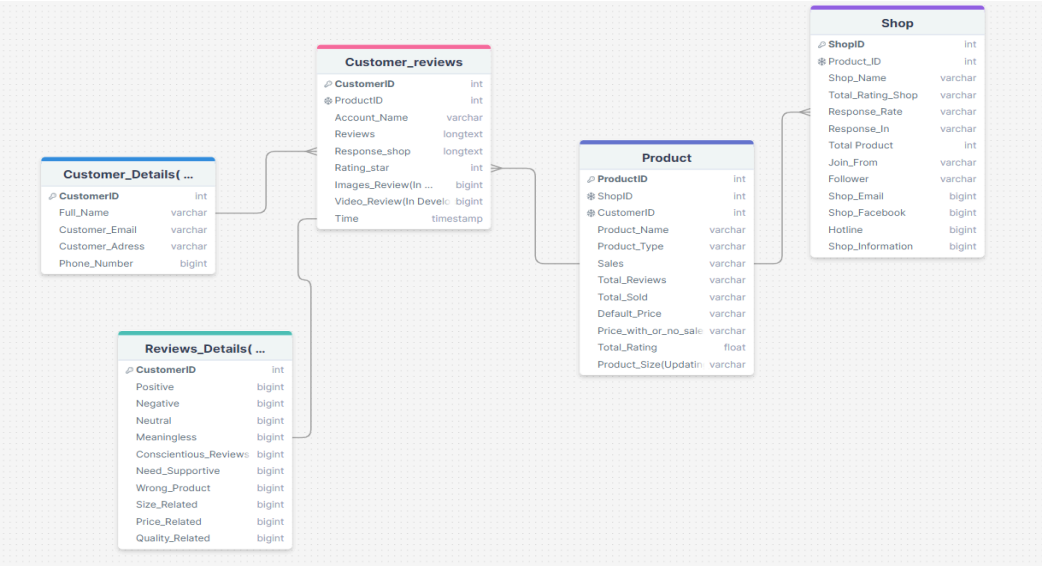
235	Chất liệu: vảiMàu sắc: đenLý do mình quyết định cho 3 sao là vì chất vải, chỉ có CHẤT VẢI thôi. Đầu tiên là vải
236	Áo mặc hơi ngứa vì chất vải , giũ áo thì nhiều vụn bay đầy ra sàn , còn lại thì ổn
237	Chất liệu: Ni ngứaChất liệu Ni , mặc vào ngứa , có tặng áo thun cũng thường , cân nhắc mua hàng
	Shop giao đúng sz
	Chất lượng sản phẩm đúng với giá
238	áo hơi nhiều chỉ thừa
239	Chất liệu: vải dờMàu sắc: xámĐúng với mô tả: đặt màu xám, giao màu đenSản phẩm thì tệ, cách nói chuyện của s

2.9 Words are stuck together.

3. Solution Implementation

3.1. Data Collection

- Data is collected from e-commerce platforms through web scraping, including:
- **Shop**: ShopID, Shop name, Shop rating, Response_Rate, Total product, Time joined, Follower, Shop information.
- **Product**: ProductID, Product Name, Type, Total Sold, Price, Sales, Total Rating, Size
- **Customer reviews**: CustomerID, Account Name, Reviews, Rating Star, Time.



3.1 Data diagram about the relationship between tables on scraped data

Some attributes are in developing, to figure out suitable ways to scrape them. Therefore, they are still in processing, not official

3.2. Data Processing (Tuan Anh)

- Collected data is cleaned and processed before analysis:
- Generated ShopID, ProductID, and CustomerID unique based on its name

Follower	Gia_Goc	Gia_Sau_	Discount	Time	Mat_Hang	Ten_khac	Thong_tin	Sao_danh	Customer_ID
81,9k	none	₫270.000	none						1
81,9k	none	₫270.000	none	#######	Phân loại	l*****1	Chất liệu:	5	2
81,9k	none	₫270.000	none	#######	Phân loại	n*****a	Áo 2 lớp đ	5	3
81,9k	none	₫270.000	none	#######	Phân loại	psuyfq133	Không nói	5	4
							Áo khoác		

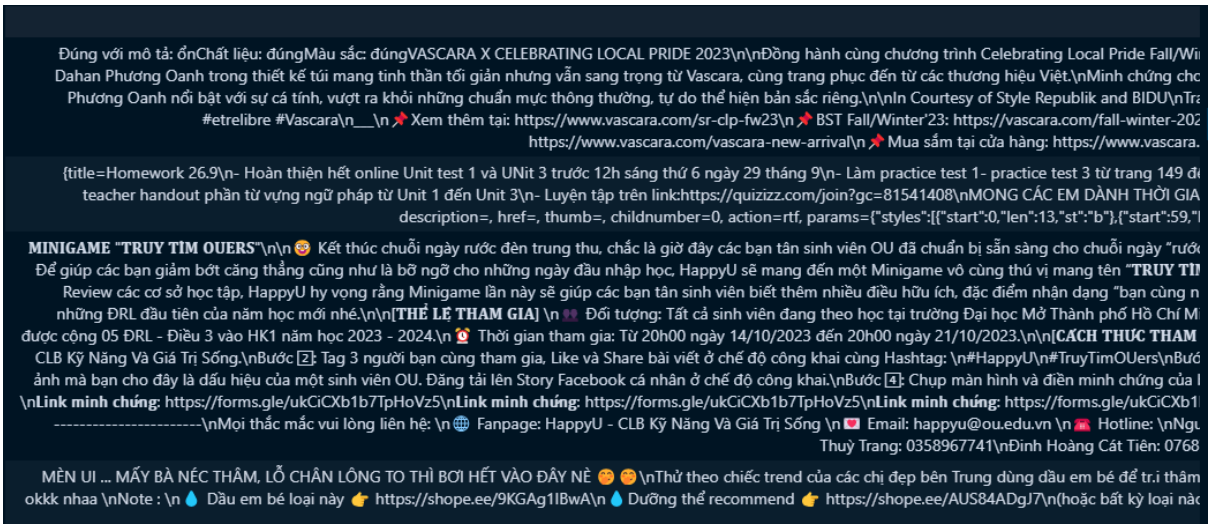
- Create a new column named `normalized_comment` to contain processed data from `raw_comment`
- Then, we will process data that has words stuck together by adding space between them.

	raw_comment	label	normalized_comment
0	Chất liệu: tốt, đẹp vải cũng sịnMàu sắc: beĐún...	1	Chất liệu : tốt , đẹp vải cũng sịn Màu sắc : b...
1	Áo 2 lớp đường may khá chắc chắn lên form cũng...	1	Áo 2 lớp đường may khá chắc chắn lên form cũng...
2	Không nói gì nhiều mng cứ xem ảnh và video nhé...	1	Không nói gì nhiều mng cứ xem ảnh và video nhé...
3	Áo khoác đẹp, không có chỉ thừa. \nVải kaki 2 ...	1	Áo khoác đẹp , không có chỉ thừa . \nVải kaki ...
4	Ưng nhất là giao hàng siêu nhanh, minh đặt hôm...	1	Ưng nhất là giao hàng siêu nhanh , minh đặt hô...

- In processing text data, texts may have different Unicode encodings such as Unicode-8 and Unicode-16. Thus, similar words may not have the same meaning, so we will bring them into a single form using unicode.normalize() from Python's standard package `unicodedata`.

	raw_comment	label	normalized_comment
0	Chất liệu: tốt, đẹp vải cũng sịnMàu sắc: beĐún...	1	chất liệu : tốt , đẹp vải cũng sịn m...
1	Áo 2 lớp đường may khá chắc chắn lên form cũng...	1	áo 2 lớp đường may khá chắc chắn le...
2	Không nói gì nhiều mng cứ xem ảnh và video nhé...	1	không nói gì nhiều mng cứ xem ảnh và ...
3	Áo khoác đẹp, không có chỉ thừa. \nVải kaki 2 ...	1	áo khoác đẹp , không có chỉ thừa . \nv...
4	Ưng nhất là giao hàng siêu nhanh, minh đặt hôm...	1	ưng nhất là giao hàng siêu nhanh , minh ...

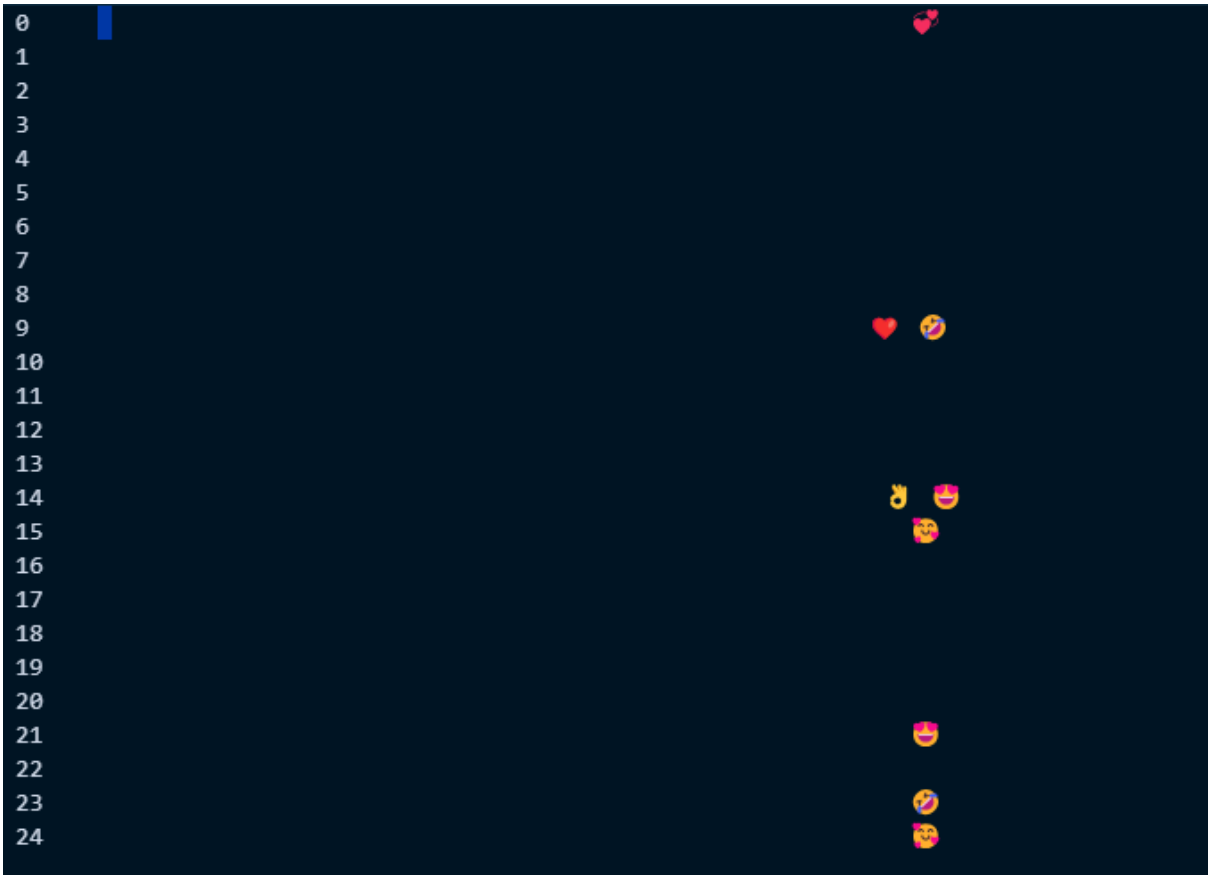
- Product comments will sometimes contain URLs inserted by sellers to help customers click to see other items. They are noise samples that we need to remove from the dataset.



- After filtering comments that contain URLs because they tend to be noisy, we will check comments where CAPITAL letters account for more than 50% of the comment length. These comments are also likely to be advertisements, because the seller wants them to. Make this comment stand out from the rest.

	raw_comment	contain_adv
3281	SUPERCALIFRAGILISTICEXPIALIDOCIOUS\nMỌI THỨ KHÁ OK. CHẤT LIỆU VẢI, KHẢ NĂNG CO DÃN. ẢNH MÈO CUTE, CHỈ CÓ CỔ ÁO HƠI CHẬT VÀ BÓ 1 TÍ NÊN 4 SAO NHE	True
5305	BTh	True
10484	K	True
11192	F	True
12414	GIAO SAI MÀU RỒI ĐẶT CỔ ÁO MÀU VÀNG MÀ GIAO NGUYÊN CÁI ÁO TRẮNG,THẤY TỘI SHIPPER ĐI GIAO NÊN NHẬN LUÔN KHÔNG ĐỔI	True
14515	H	True

- Since it doesn't seem to be noise data, we won't discard these data.
- Comments include emojis. These emojis are also a valuable source of data so we can distinguish positive comments from negative comments. We will use a python package called `emojis` to separate emojis from comments.



- Then, we will remove punctuation and special characters.

1	Áo 2 lớp đường may khá chắc chắn lên form cũng đẹp. Mk m67 mặc size L đẹp lắm nha. Giao hàng còn rất là nhanh, shop uy tín và thân thiện. Mọi người nên mua thử nhé, đảm bảo chất lượng	1	áo lớp đường may khá chắc chắn lên form cũng đẹp mk m mặc size l đẹp lắm nha giao hàng còn rất là nhanh shop uy tín và thân thiện mọi người nên mua thử nhé đảm bảo chất lượng
2	Không nói gì nhiều mng cứ xem ảnh và video nhé. Áo màu đẹp vs hoạ tiết ổn. Mng mặc nên phối thêm áo trong ấm hơn nhé chứ áo này là không đủ ấm đâu :))	1	không nói gì nhiều mng cứ xem ảnh và video nhé áo màu đẹp vs hoạ tiết ổn mng mặc nên phối thêm áo trong ấm hơn nhé chứ áo này là không đủ ấm đâu
3	Áo khoác đẹp, không có chỉ thừa. \nVải kaki 2 lớp mềm mỏng. Áo này chỉ mặc khoác ngoài mùa thu thôi chứ mùa đông lạnh :)) \nSẽ ủng hộ shop	1	áo khoác đẹp không có chỉ thừa vải kaki lớp mềm mỏng áo này chỉ mặc khoác ngoài mùa thu thôi chứ mùa đông lạnh sẽ ủng hộ shop
4	Ưng nhất là giao hàng siêu nhanh, minh đặt hôm 22 mà giờ đã nhận được rồi.\nÁo mỏng vừa phải, vải kaki khá ổn áp.\nMình cao 1m50 nặng 45kg, chọn size M qua mông nên rất ưng (lúc đầu sợ ngắn quá).\nVới giá tiền này nên mua nha mng.	1	ưng nhất là giao hàng siêu nhanh minh đặt hôm mà giờ đã nhận được rồi áo mỏng vừa phải vải kaki khá ổn áp mình cao m nặng kg chọn size m qua mông nên rất ưng lúc đầu sợ ngắn quá với giá tiền này nên mua nha mng

- Comments will usually always have acronyms, these words also contribute to the meaning of the sentence so we must standardize them into a standard form so that the machine can learn and understand. We will standardize some basic abbreviations found in the file `modules/dependencies/abbreviate.txt`. This file contains basic abbreviations that young people often use in comments, and we can add them over time.

dules > dependencies > abbreviate.txt

```
sp,sản phẩm
sd,sử dụng
sph,sản phẩm
ths,thanks
tks,thanks
thk,thanks
thank you,thanks
thabs,thanks
thương,thương
thw,thương
đep,đẹp
gh,giao hàng
góm,gớm
gê,ghê
dỏm,dởm
nsc,nói chuyện
nc,nói chuyện
```

abbreviate.txt

```
# xây dựng dictionary cho các từ viết tắt
abbreviate = Utils.buildDictionaryFromFile("./modules/dependencies/abbreviate.txt")

# test
abbreviate['okela']
```

✓ 0.0s

```
'ok'
```

Test standardize words with abbreviations

- In the dataset, there are duplicated words such as `*chờiiiiiii ơiiiiiii, xinhhhhhhh quá, đẹp xỉuuuuuuuuuuuu*`, we need to normalize these words to *"chời ơi, xinh quá, đẹp xỉu*`.

```
reviews['normalized_comment'] = reviews['normalized_comment'].apply(lambda cmt: Processor.removeDuplicateLetters(cmt))
# Test
test_comment = 'okkkkkkkkkkkkkkkkkkkkk chờiiiiiiiiii ơiiiiiiii xinhhhhhhhhhhh quá đẹpppppppppp xỉuuuuuuu'
result = Processor.removeDuplicateLetters(test_comment)
print("Original comment:", test_comment)
print("Processed comment:", result)
```

✓ 0.1s

```
Original comment: okkkkkkkkkkkkkkkkkkkkk chờiiiiiiiiii ơiiiiiiii xinhhhhhhhhhhh quá đẹpppppppppp xỉuuuuuuu
Processed comment: ok chời ơi xinh quá đẹp xỉu
```

- As mentioned above in the Data Problem section, some comments contain meaningless words, so we will proceed to delete them. In addition, there are comments using languages other than Vietnamese such as English, Chinese, Korean, Japanese, etc. These comments may be noise samples that cause the model to reduce performance, so we will also eliminate them. leave them. The simplest way to do it is to build a dictionary containing single words of Vietnamese. For each comment, if the number of words not found in this dictionary is greater

than the number of words found in the dictionary, then the possibility is a noise sample. However, there are still a few English words that we need to keep such as shipper, shop,... so we will use the `enchant` package to check whether a word is an English word or not.

- We see that there is a large difference between the two groups positive and negative, so our training data set will be equal 0.8 * min(size(positive), size(negative))*2. It is the smallest number of labels in the `reviews` data set. Then separate the reviews into two groups based on labels. Shuffle and reset the index for reviews_positive and then randomly sample from reviews_positive. Select samples from `reviews_positive` based on the random index taken and combine with `reviews_negative` to produce a balanced dataset between the 2 classes.

```
normalize_reviews = pd.concat([reviews_negative, reviews_positive2], axis=0)
normalize_reviews = normalize_reviews.reset_index(drop=True)

Processor.printAfterProcess(normalize_reviews)
normalize_reviews.head()
```

```
Shape: (12394, 4)
label
0    6197
1    6197
Name: count, dtype: int64
```

- Finally, split the dataset into train and test with a ratio 80:20.

3.3. Data Analysis

- Review Content Analysis: Utilizing natural language processing (NLP) techniques to analyze review content, identifying product strengths and weaknesses. This includes extracting key phrases, topics, and common themes from reviews.
- Sentiment Analysis: Training a sentiment analysis model using machine learning or deep learning algorithms to classify customer reviews as positive, negative, or neutral. The model is trained on a labeled dataset with pre-annotated sentiments to improve accuracy.
- Identifying Specific Issues: Conducting a detailed analysis of negative reviews to identify specific product issues such as quality, delivery, or customer service. This helps in pinpointing areas that need immediate attention.

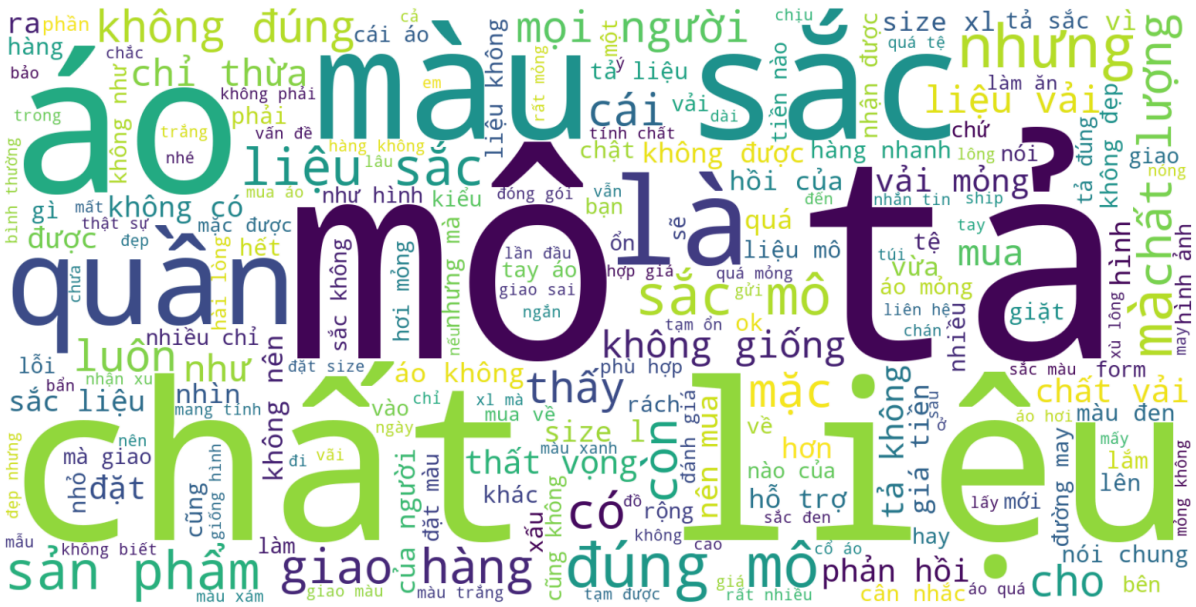
3.4. Review Filtering

- Applying machine learning algorithms to detect and filter out fake or unreliable reviews. This involves:
- Identifying Fake Review Patterns: Using characteristics like review frequency, language used, and review timing to identify fake reviews. For example, a high number of reviews in a short period or reviews with overly positive/negative language can indicate suspicious activity.
- Applying the Model: Utilizing trained models to automatically filter out fake reviews from the dataset, ensuring the analysis is based on genuine customer feedback.

3.5. Report Visualization

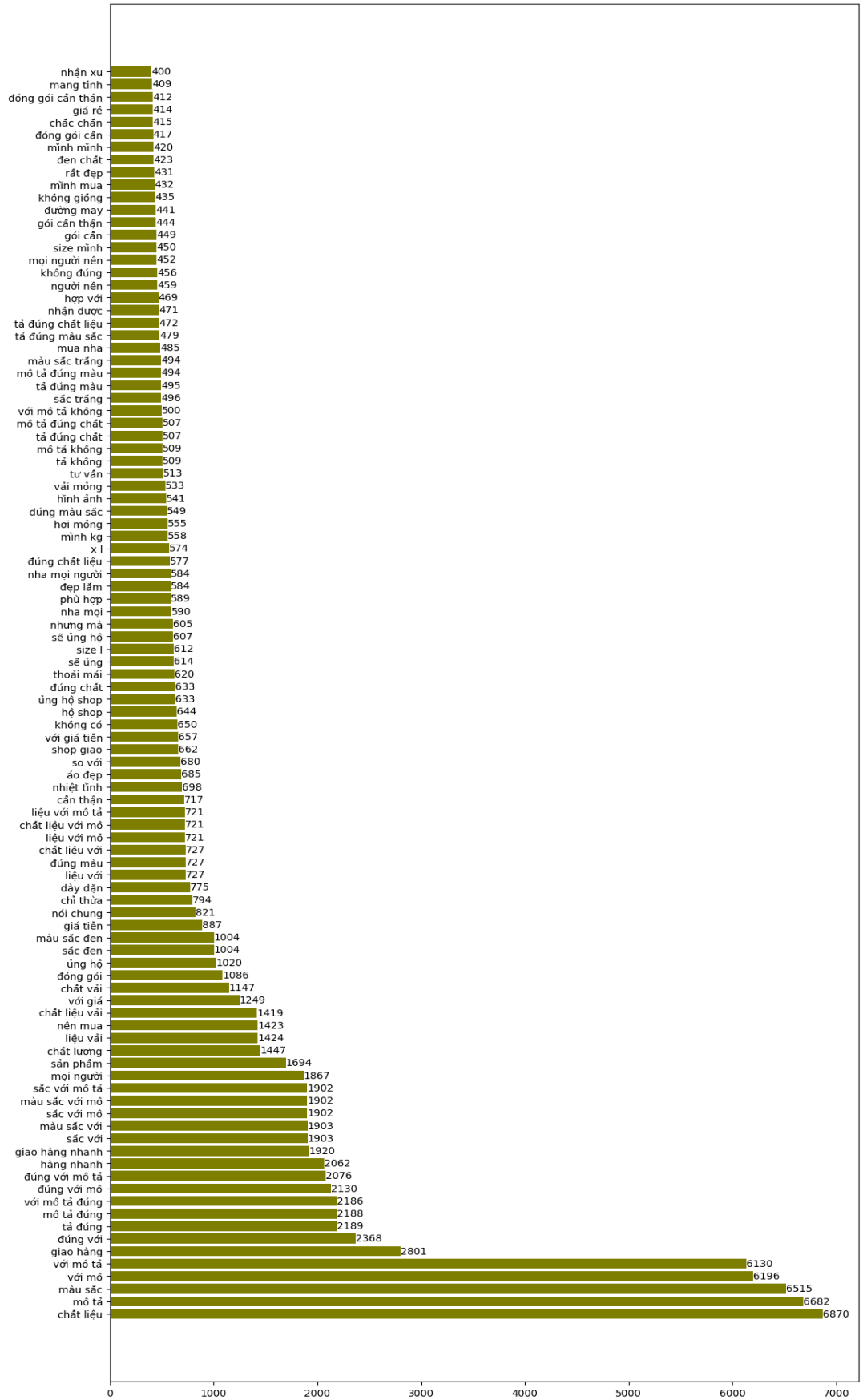
- Creating visual reports to help shops recognize product trends and issues. This includes:

    - Rating Charts: Displaying charts of the number of reviews over time, categorized by star ratings. This helps in identifying trends and patterns in customer satisfaction.
    - Keyword Analysis: Displaying popular keywords in positive and negative reviews using word clouds and frequency analysis. This highlights common praises and complaints.
    - Improvement Suggestions: Providing specific suggestions to improve products and services based on review analysis. For instance, if delivery time is a common issue, the shop can focus on improving logistics.
    - With the help of WordCloud, we can easily get an overview of text data, showing us which words overlap across classes. In addition, it helps us clearly list important compound words. The compound words will have a frequency that appears close to each other so they will have the same font-size - typically here we see the words such as: *`sản phẩm, mô tả, màu sắc, chất liệu,...`*



3.5.2 Vocabulary often appears in the negative class

- To see in detail the difference between the frequency with which words in each class appear. One of the ways when encountering this problem we will apply the methods one by one is **Bag of words** and then **TF-IDF**, then I divide it into [m, n] and frequency statistics to see which words are in these paragraphs.



3.5.3 100 compound words with the highest frequency

- Through the Two Directions barplot, we can compare the distribution of emotion labels (usually positive and negative) between two different dimensions. As shown below, these overlapping words are mostly tilted to one side, this helps the model avoid confusion when having to make decisions when encountering these words.

3.4 Overlap of words between classes

4. Feature engineering

- **Tokenization**: The text data is tokenized using Pyvi's ViTokenizer and Keras's Tokenizer. ViTokenizer splits the text into tokens specifically for Vietnamese language processing, while Keras's converts the tokenized text into sequences of numbers. This dual-step tokenization is essential for converting raw text into a format that can be fed into a machine learning model.

```
# Tokenize
if tokenize:
    df['content'] = df['content'].apply(lambda x: ViTokenizer.tokenize(x))
return df
```

- **Padding**: Ensuring all sequences have the same length by adding special padding tokens. Padding is crucial because models like LSTM and RNN require inputs of the same length for

```
X_train = sq.pad_sequences(X_train, maxlen=maxlen)
print('X_train shape:', X_train.shape)
X_test = sq.pad_sequences(X_test, maxlen=maxlen)
print('X_test shape:', X_test.shape)
X_val = sq.pad_sequences(X_val, maxlen=maxlen)
print('X_val shape:', X_val.shape)
```

batch processing.

- **Embedding**: Using pre-trained word vectors from cc.vi.300.vec from https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.vi.300.vec.gz and Keras's embedding layer to map words into fixed-length vectors (300 dimensions). This embedding layer is initialized with pre-trained embeddings which are key for transforming words into numerical vectors that the model can learn from and optimize.

```
EMBEDDING_FILE =  '/content/drive/MyDrive/DBM/cc.vi.300.vec'

def load_embeddings(filename):
    embeddings = {}
    with open(filename) as f:
        for line in f:
            values = line.rstrip().split(' ')
            word = values[0]
            vector = np.asarray(values[1:], dtype='float32')
            embeddings[word] = vector
    return embeddings

embeddings = load_embeddings(EMBEDDING_FILE)
```

```
from keras.preprocessing import text, sequence
```

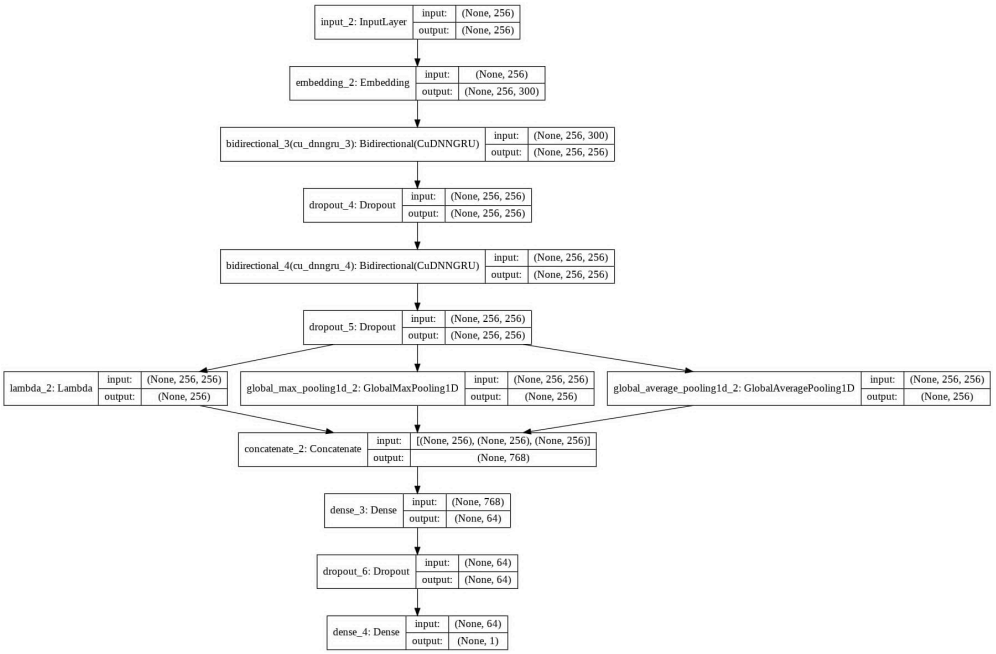
```
def filter_embeddings(embeddings, word_index, vocab_size, dim=300):
    embedding_matrix = np.zeros([vocab_size, dim])
    for word, i in word_index.items():
        if i >= vocab_size:
            continue
        vector = embeddings.get(word)
        if vector is not None:
            embedding_matrix[i] = vector
    return embedding_matrix

embedding_size = 300
embedding_matrix = filter_embeddings(embeddings, tokenizer.word_index,
                                     vocab_size, embedding_size)
print('OOV: {}'.format(len(set(tokenizer.word_index) - set(embeddings))))
```

5. Modeling and evaluation

5.1. Self-Attention RNN model:

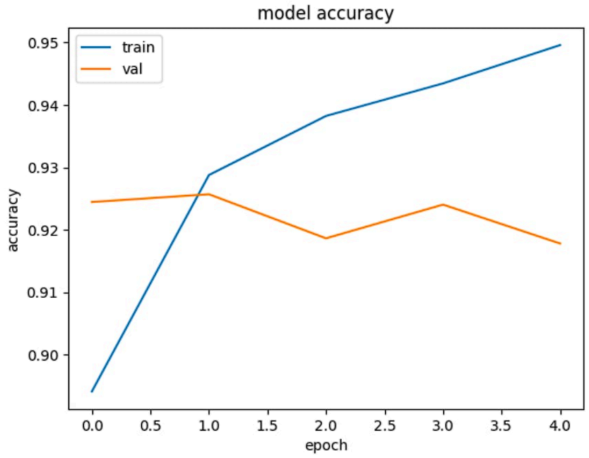
- **The self-attention RNN model** is a neural network that uses the **self-attention mechanism** to enhance the processing capabilities of **RNNs**. The self-attention mechanism allows the model to focus on different important parts of the input sequence at each time step, rather than relying solely on the hidden state as in traditional RNNs.This helps the model better handle long sequences and complex relationships in the data.

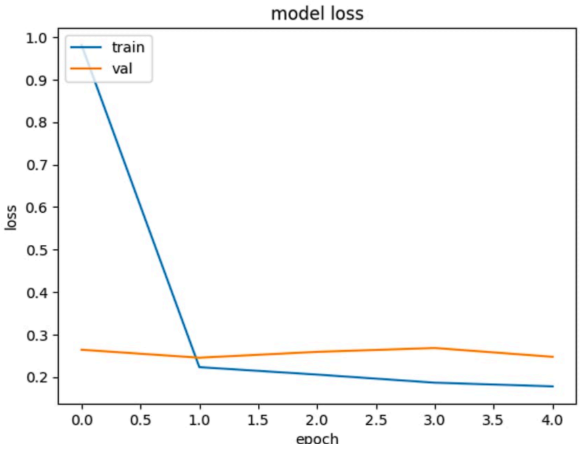


- **RNNs** are a type of neural network designed to process sequential data such as text, audio, or time series signals. **RNNs** use a hidden state to store information from previous steps in the sequence, allowing the model to have a "memory" of what has happened before. However, **traditional RNNs** face difficulties when processing long sequences due to the "vanishing gradient" and "exploding gradient" problems. These issues reduce the model's ability to learn and remember information from distant parts of the sequence.
- **The self-attention mechanism** is a method that helps the model focus on important parts of the sequential data, regardless of their distance in the sequence. This is done by calculating a set of attention weights based on the relationships between different parts of the sequence.

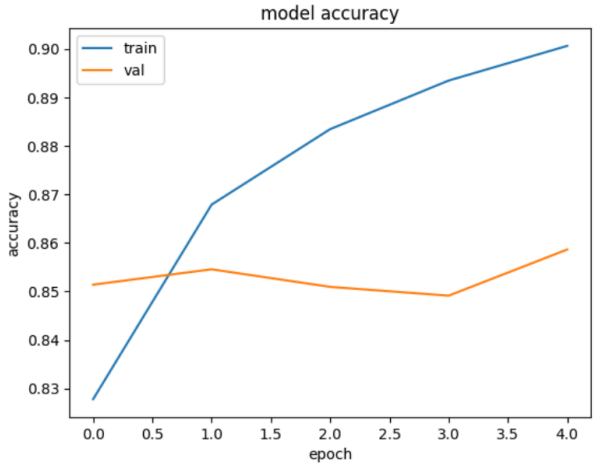
5.1.2.Evaluation of SARNN

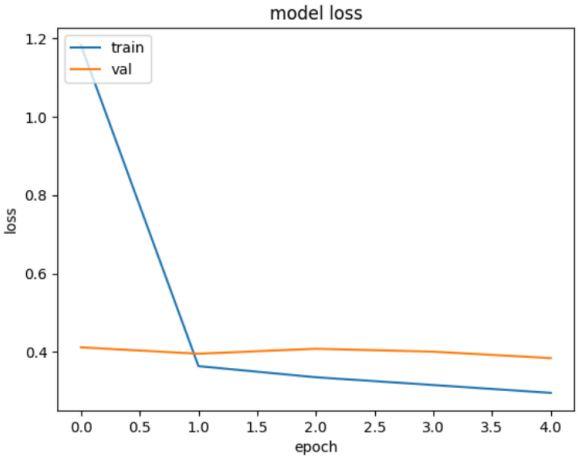
On 12k-data:





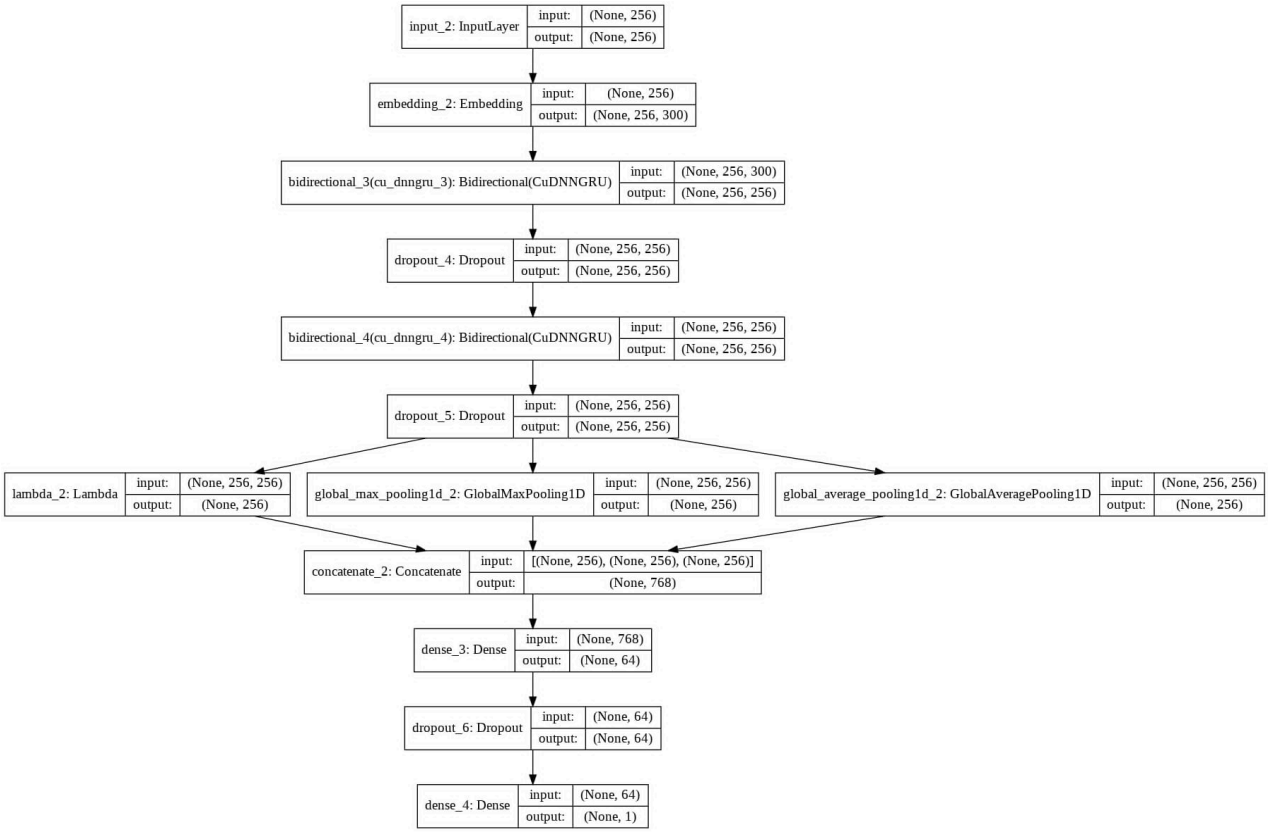
On 24k-data:





5.2 Self-Attention LSTM model:

- **Self-Attention LSTM** is a model that combines the self-attention mechanism with **LSTM (Long Short-Term Memory)** to enhance the processing capabilities of **LSTM.** This combination allows the model to focus on important parts of the input sequence, overcoming the limitations of **traditional LSTM.**

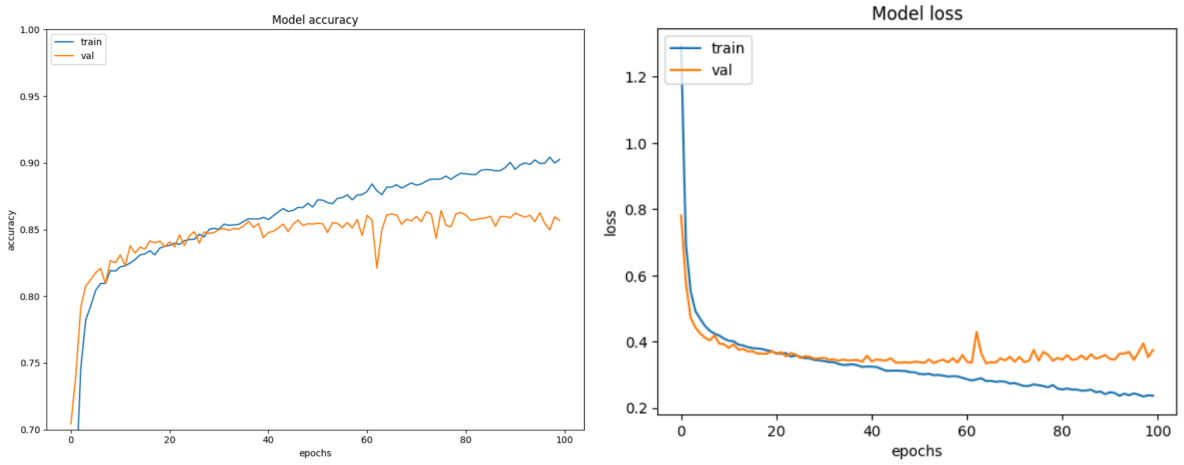


Benefits of Self-Attention LSTM

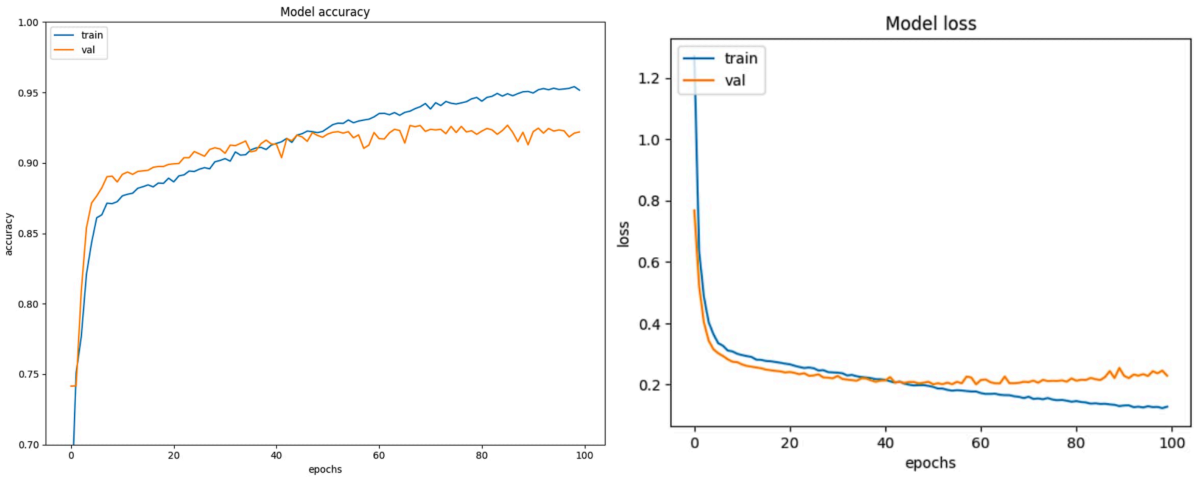
- Ability to process long sequences: The self-attention mechanism helps the model focus on important parts of the sequence, mitigating the vanishing gradient problem.
- Learning complex relationships: The model can learn complex relationships in the data without being limited by the distance between elements in the sequence.
- Explainability: Attention weights can help explain the model's decisions, showing which parts of the input sequence the model considered most important.

5.2.2.Evaluation of SA-LSTM

On 24k-data:



On 12k-data:



5.3.Evaluate SARNN and SA-LSTM on 12k dataset and 24k dataset

There are several reasons that can explain why the larger dataset (24k) has lower accuracy compared to the smaller dataset (12k) when training the SARNN model:

- If the larger dataset contains **more noise or irrelevant data** compared to the smaller dataset, this can lead to poor model performance. Data quality is crucial, and unclean data can degrade the model's performance.
- When increasing the dataset size, it may be **necessary to adjust the model parameters**, such as the learning rate, the number of epochs, or the model architecture (number of layers, number of units in each layer, etc.). If not adjusted properly, the model's performance can be affected.
- A larger dataset **requires more training time**. If the training process is not long enough, the model might not learn adequately from the larger dataset.

6. Challenging and Limitations

- Several security problems in the crawling step. We crawled the data from shopee.vn, which uses cloudflare and antibot technologies so we need to investigate some methods to bypass these security layers.
- The type of our data is text and we lack experience in handling text data. Most of our previous projects were about image data so this project’s processing steps and technologies are new to us, especially the models. During the project, we needed to learn new technologies in order to process the text data (tokenize, padding, encoding, etc.) so we spent an amount of time investigating and learning new technologies. This problem slowed us down at first, but after we got used to the technologies, we were able to handle the data from scratch.
- Not able to apply Power BI on the processed data. The main reason is that the team lacks experience in using Power BI and advanced tools like this. In addition, the data is complicated and the crawling step was more complicated than expected so the time for investigating and

using Power BI was shortened. Therefore, we didn't use Power BI for our data. However, we investigated the tool and our data and slightly doubt that Power BI is currently not appropriate for our data.

7. Future Development

- In the future, we will try to do more research to make the data cleaner, thereby improving model performance.
- Apply more functions in scrape tools for more data fields like customer information, and shop information like email or pages on social media. We can send the shop's page url on other social media to the customers who love the product and give positive reviews to provide them with more information and videos about shop products like a marketing campaign.
- Take the shop's response to each customer review to develop an auto-generated response for the shop to support customer issues. It will first base on the reviews, categorize it and use the shop's response on these reviews for model generation to learn how to respond and send it back to the shop or even automate replies when a new review or claim appears. It also works when shops don't have time to reply to each review but have to keep their response rate and support their potential customer.
- Develop tools to scrape videos and images from reviews, mostly negative ones for checking if it's true or not to decide support or not and determine what problems on the product to improve it later on. Save those images or videos if it's good feedback to introduce them to people who having an interest on our product.
- Make the crawling process automatic. Currently, there are still some manual actions in the crawling step which slows us down. Therefore, we want to develop an automatic crawling system, which will crawl the data automatically based on some config. Currently, there is no clear plan for this, but there are some promising ideas like Puppeteer and Chrome Extension.
- Modularize existing processing methods in order to make them reusable. We have all the preprocessing and feature engineering methods in a Jupyter Notebook which are hard to be reused later. Therefore, we plan to have these methods modularized into some separated classes in order to be reused in different projects.

8. Conclusion

- The final result of the project is we understand more about the reviews data on Shopee. This result matches our expectations when we started working on the project. We succeeded in all the steps of the project such as crawling, preprocessing, feature engineering, using models to extract the complex features. Each step gives us some insights about the data and makes the data more useful. We can see there are more further steps we can take in order to take advantage of the data, however we consider the current result as OK for a 9-week project. In addition, during the project, we learned more about data mining concepts and technologies, which is our main goal when starting the project. In short, the project result matches our expectations: understanding the reviews data on Shopee and learning new things about the data mining field.
- After this project, we learned many lessons about the data mining field.The first thing is that the quality of data is important. We processed the data from scratch, went through all the steps

like crawling, preprocessing, processing, feature engineering, extract complex features using different models. Therefore, we understand that the process to improve the data is the art of using different kinds of techniques to make the data useful. The second lesson is the techniques we used to process the data. When working with the data, we need to process it through many steps. Each step uses different techniques in order to make the data useful and help us to understand the data. Going through the project, we learned to use some advanced techniques such as crawling tons of data from scratch using python, flask, bs4, javascript and our crawling strategy, handling data problems like abbreviations, duplicated data, null values, applying sentiment analysis models “RNN” to understand the complex features of the data. In short, via the project, we understand the problems of the data and learn to use different techniques to handle them. Last but not least, we learned about how to present our works. This lesson is about showing what we learned from the data and presenting the insights of the data to the others. After processing the large amount of data, we want to share the knowledge about our data using some common techniques such as visualizations, statistics and slides. We consider this lesson as important as the two above because presenting our works is proof that we understand our data. To conclude, after the project, we learned some important lessons like the process to improve the data from scratch, the techniques to handle the data problems and presenting our works to the others.