ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA HỆ THỐNG THÔNG TIN

PROJECT REPORT

DETECTING SPAM REVIEWS ON VIETNAMESE E-COMMERCE PLATFORMS

Lecturer: **Đỗ Duy Thanh**

Made by: **Lê Nguyễn Lam Lâm**

Trần Anh Thy

Ho Chi Minh city, July 2023

Abtract

This project focuses on developing an AI system to detect and eliminate spam reviews on Vietnamese e-commerce platforms. By using advanced natural language processing and machine learning techniques, the system will analyze the content of user reviews, distinguishing between genuine and spam reviews. The goal is to ensure the authenticity and reliability of reviews, enhancing customer experiences and fostering a trustworthy e-commerce environment in Vietnam.

Introduction

With the exponential growth of e-commerce platforms in Vietnam, ensuring the authenticity and reliability of user reviews has become a critical challenge. The presence of spam reviews not only misleads potential customers but also hampers the growth and reputation of online businesses.

This project aims to develop an Artificial Intelligence (AI) system to detect and filter out spam reviews on Vietnamese e-commerce platforms.

The proposed system leverages natural language processing (NLP) techniques and machine learning algorithms to analyze the textual content of user reviews. The dataset will be preprocessed to extract relevant features such as sentiment, keyword frequency, and linguistic patterns.

The developed AI system will be implemented as a scalable and efficient solution that can be seamlessly integrated into existing e-commerce platforms.

By experimenting with various machine learning architectures for NLP, such as CNNs, LSTMs, and other ML structures, CNNs yielded the best performance metrics. Therefore, we have chosen CNNs as the main model for this project.

The successful implementation of this project will significantly contribute to maintaining the integrity and trustworthiness of Vietnamese e-commerce platforms. By automatically identifying and eliminating spam reviews, online shoppers will have access to more reliable and informative feedback, leading to improved customer experiences and fostering a healthier e-commerce ecosystem in Vietnam.

Dataset

The dataset we obtained is publicly available and attached on paper by Thanh Son Luu and his team. you can find it with this name: "Detecting Spam Reviews on Vietnamese E-Commerce Websites. In Intelligent Information and Database Systems.".

Original dataset comprises two tasks. The first task determines whether the reviews are spam or not spam (Task 1), and the second task indicates the types of spam reviews (Task 2). But in this project we just use first task that filter spam or not.

NO-SPAM (0): Reviews labeled with this label are regular reviews, true to the product's reality. Reviews like these provide helpful information for buyers to get an overview of the product before deciding whether to buy it or not.

SPAM (1): Reviews labeled with this label are reviews that are entirely or partially untrue about products sold on e-commerce sites. Reviews like these often make it easier to sell products or hurt the sales and reputation of stores and provide inaccurate or unhelpful information.

Star	Comment	Label
5	Chất lượng sản phẩm tuyệt vời, má	0
1	quá mỏng, keo dính tay quá nhiều, quá mệt công	0
5	Shop nay giao hàng nhanh xỉu 😂 mua quá nhiu lần r hay dc tặng thêm hehe se ung hộ thêm nhiu lần	1
1	Giao hàng không đúng mẫu khi đặt. Đặt hàng 1 đằng thì shop giao hàng 1 kiểu.Mua lần đầu cũng là lần cuối	0

Step by step

Step	Content
1.	Data cleaning
2.	Data preprocessing
3.	Model building & Evaluation
4.	Compare
5.	Deploy

Data clearning

Data cleaning is the initial stage of any machine learning project and is one of the most critical processes in data analysis. It is a critical step in ensuring that the dataset is devoid of incorrect or erroneous data. It can be done manually with data wrangling tools, or it can be completed automatically with a computer program. Data cleaning entails a slew of procedures that, once done, make the data ready for analysis. Given its significance in numerous fields, there is a growing interest in the development of efficient and effective data cleaning frameworks.

Some data cleaning tasks that we do:

- Drop rating, drop Spam label (just use necessary data).
- Dropduplicate values and rows.
- Split data to use

Before

1	Rating	Comment	Label	SpamLabel
2	5	Chất lượng sản phẩm tuyệt vời, má	0	0
3	5	Jdjd is a waste of my time and place to ask questions about assignments and I will be rewarde	1	3
4	5	Dhnxncbcncncncncnncncncnncncncncncncncnncncncjjnjn	1	3
5	5	giao hàng nhanh, đóng gói cẩn thận, sản phẩm nói chung là ok.	0	0
6	3	Bạt mỏng vs lại không có túi đi kèm để cất khi không dùng tới. Tạm được, phù hợp vs gía tiền	0	0
7	1	Đo size đúng theo hướng dẫn của shop nhưng vẫn bị trật. K biết có đổi được không?	0	0
8	5	Nói chung ổn dễ dùng, ko sắc nét đc như máy bt chắc do in nhiệt, phục vụ in dán đơn tiện. Sh	0	0
9	4	hàng đúng y hình, chưa dùng nên chưa biết chất lượng thế nào, nhưng giao hàng lâu quá!	1	2

After

```
[12] X_train #X_train is data for review (comment)
```

	Comment
0	Chất lượng sản phẩm tuyệt vời, má
1	Jdjd is a waste of my time and place to ask qu...
2	Dhnxncbcncncncncnncncncnncncncncncncncnncncncj...
3	giao hàng nhanh, đóng gói cẩn thận, sản phẩm n...
4	Bạt mỏng vs lại không có túi đi kèm để cất khi...
...	...

```
y_train #y_train is labels
```

	Label
0	0
1	1
2	1
3	0
4	0
...	...

Data preprocessing

Purpose of data preprocessing is preparing the text data for analysis by converting it into a format that can be easily understood by machine learning algorithms.

Improve the accuracy and effectiveness of the NLP classification model.

Some skill that we do:

- Tokenizer
- Lower casing
- Stop words removal
- Steamming

- Lemmatization

Prepocessing function:

```
[ ] def filter_stop_words(train_sentences, stop_words):
        new_sent = [word for word in train_sentences.split() if word not in stop_words]
        train_sentences = ' '.join(new_sent)
        return train_sentences


    def deEmojify(text):
        regrex_pattern = re.compile(pattern = "["
            u"\U0001F600-\U0001F64F"  # emoticons
            u"\U0001F300-\U0001F5FF"  # symbols & pictographs
            u"\U0001F680-\U0001F6FF"  # transport & map symbols
            u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                               "]+", flags = re.UNICODE)
        return regrex_pattern.sub(r'',text)


    def preprocess(text, tokenized = True, lowercased = True):
        text = ViTokenizer.tokenize(text) if tokenized else text
        text = filter_stop_words(text, stopwords)
        text = deEmojify(text)
        text = text.lower() if lowercased else text

        text = text.strip()
        text = re.compile('<.*?>').sub('', text)
        text = re.sub('\s+', ' ', text)
        text = re.sub(r'\[[0-9]*\]',' ',text)
        text = re.sub(r'[^\w\s]', '', text)
        text = re.sub(r'\d',' ',text)
        text = re.sub(r'\s+',' ',text)
        return text
```

Result:

```
text_01 = input("Enter some text: ")
text_02 = preprocess(text_01)
print("Before preprocess: ", text_02)

Enter some text: Thầy Đỗ Duy Thanh dạy môn Thương mại điện tử ở trường Công nghệ thông tin @@ 3 😀😀
Before preprocess:  thầy đỗ_duy_thanh_dạy môn thương_mại điện_tử trường công_nghệ thông_tin
```

Data preprocess feature function:

```
[23] def pre_process_features(X, y, tokenized = True, lowercased = True):
         X = np.array(X)
         y = np.array(y)
         X = [preprocess(str(p), tokenized = tokenized, lowercased = lowercased) for p in list(X)]
         for idx, ele in enumerate(X):
             if not ele:
                 X = np.delete(X, idx)
                 y = np.delete(y, idx)
         return X, y
```

```
# train_X trở thành mảng
train_X, train_y= pre_process_features(X_train['Comment'], y_train['Label'], tokenized=True, lowercased = True)
dev_X, dev_y= pre_process_features(X_dev['Comment'], y_dev['Label'], tokenized=True, lowercased = True)
test_X, test_y = pre_process_features(X_test['Comment'], y_test['Label'], tokenized=True, lowercased = True)
```

Result

```
[27] train_X

array(['chất_lượng sản_phẩm tuyệt_vời má',
       'jdjd is a waste of my time and place to ask questions about assignments and i will be rew
       'dhnxncbcncncncncnncncncnncncncncncncncnncncncjjnjn', ...,
       'sh tư_vấn nhiệt_tình giao hàng nhanh_chóng sản_phẩm chất_lượng ',
       'hài_lòng giây đẹp đi êm chân giá rẻ đóng_gói cẩn_thận shipper vui_vẻ',
       'chất_lượng sản_phẩm tuyệt_vời giao hàng đóng_gói sản_phẩm '],
      dtype='<U1047')
```

```
train_X.shape

(14273,)
```

Word embedding in NLP is an important term that is used for representing words for text analysis in the form of real-valued vectors. It is an advancement in NLP that has improved the ability of computers to understand text-based content in a better way. It is considered one of the most significant breakthroughs of deep learning for solving challenging natural language processing problems.

Embbedding features and exucute

```
def embedding_featues(X, y, tokenizer, is_one_hot_label=True, number_class1=2):
    X = tokenizer.texts_to_sequences(X)
    X = pad_sequences(X, maxlen=sequence_length)
    if is_one_hot_label:
        y = to_categorical(y, num_classes=number_class1)
    return X, y
```

```
train_X, train_yy = embedding_featues(train_X, train_y, tokenizer)
dev_X, dev_yy = embedding_featues(dev_X, dev_y, tokenizer)
test_X, test_yy = embedding_featues(test_X, test_y, tokenizer, is_one_hot_label=False)
```

Result

Before

```
train_X
```

```
array(['chất_lượng sản_phẩm tuyệt_vời má',
       'jdjd is a waste of my time and place to ask questions about assignments and i
       'dhnxncbcncncncncnncncncnncncncncncncncnncncncjjnjn', ...,
       'sh tư_vấn nhiệt_tình giao hàng nhanh_chóng sản_phẩm_chất_lượng ',
       'hài_lòng giầy đẹp đi êm chân giá rẻ đóng_gói cẩn_thận shipper vui_vẻ',
       'chất_lượng sản_phẩm tuyệt_vời giao hàng đóng_gói sản_phẩm '],
      dtype='<U1047')
```

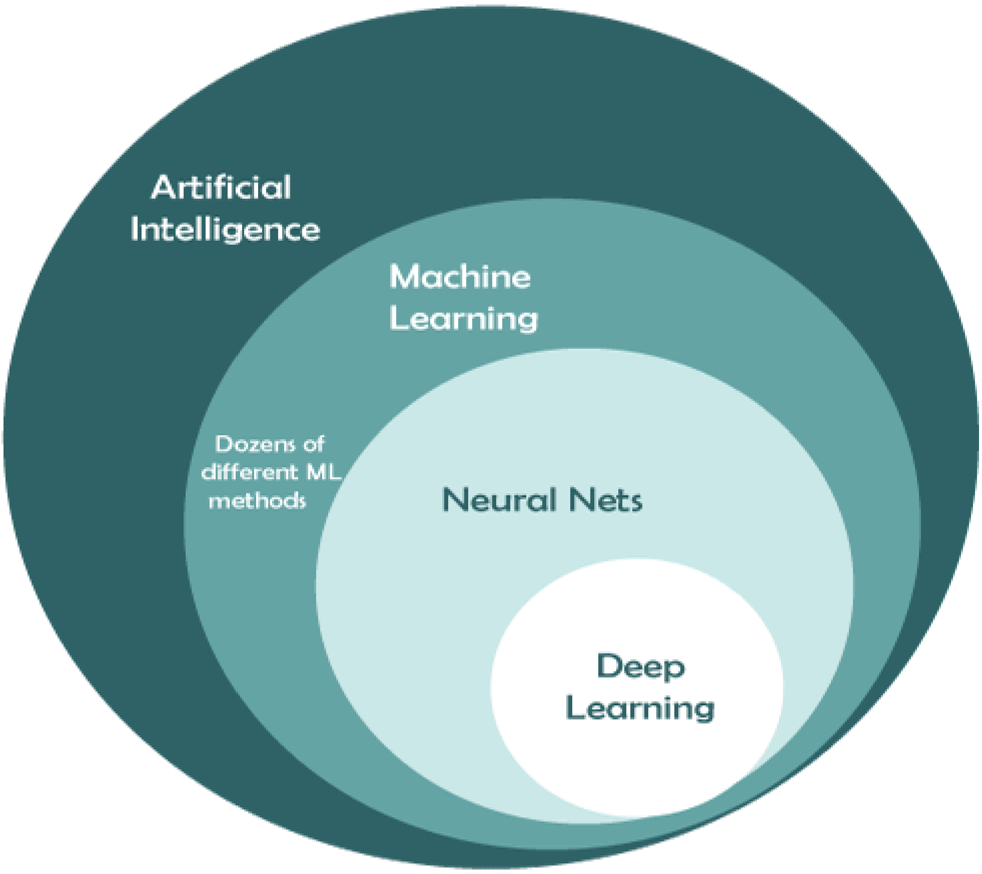
After

```
train_X
```

```
array([[   0,    0,    0, ...,    5,   49, 1936],
       [   0,    0,    0, ...,  671, 4707, 4708],
       [   0,    0,    0, ...,    0,    0, 6455],
       ...,
       [   0,    0,    0, ...,    1,  111,  150],
       [   0,    0,    0, ...,   15,  116,  655],
       [   0,    0,    0, ...,    1,    7,    5]], dtype=int32)
```

Model building & Evaluation

Model building and Evaluation involves the creation and testing of machine learning models to perform tasks such as sentiment analysis, language translation, text classification, and more.



we choose 2 deep learning model to focus: cnns and lstrms:

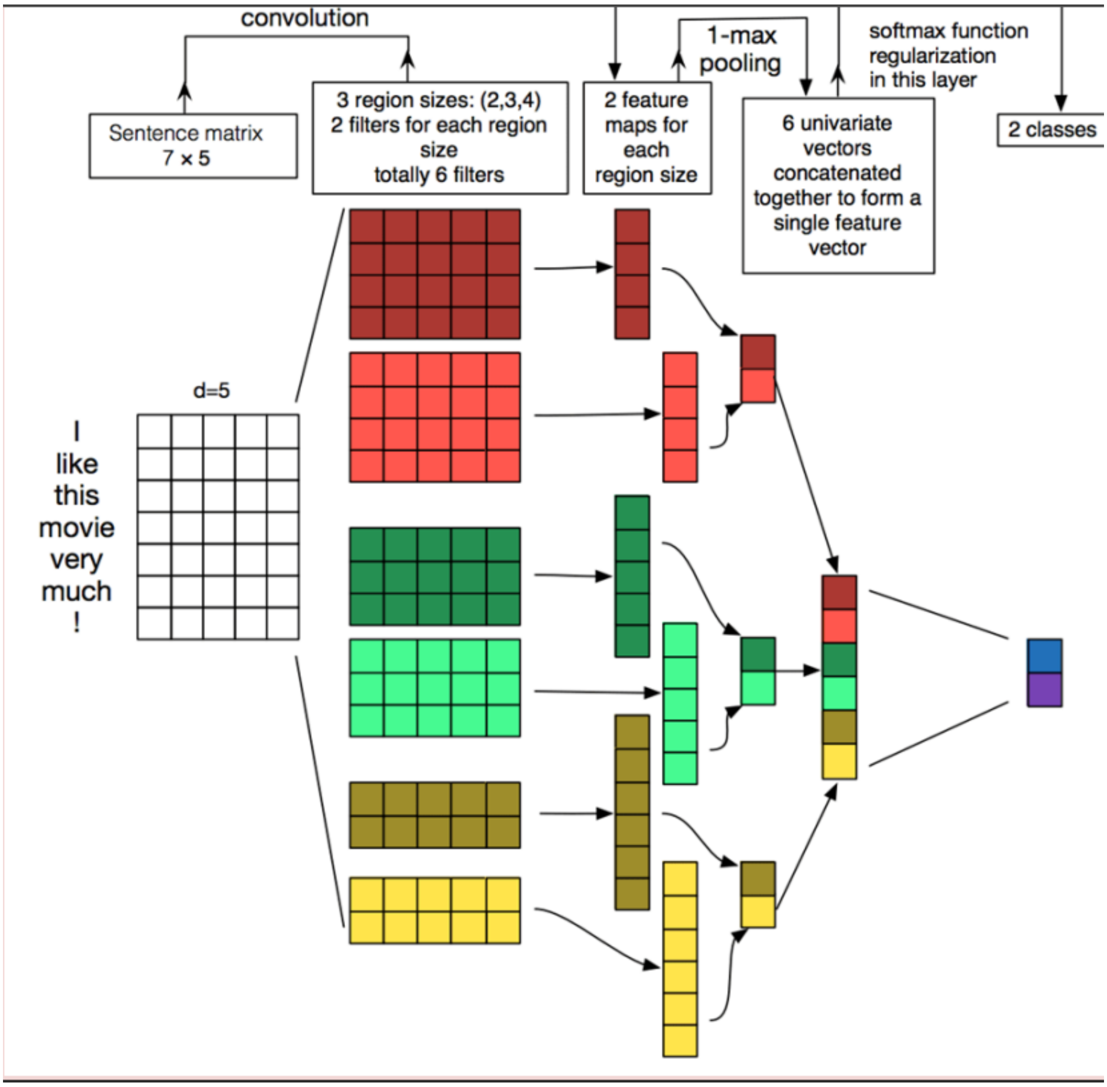
CNNs - (Convolutional Neural Networks)

Basic functionality of convolution neural network resembles to the visual cortex\of the animal brain. In text classification task convolution neural network gives promising result. Criteria for text classification is similar to image classification only difference is that instead of pixel values we have matrix of word vectors.

The first layer in convolution neural network is embedding layer which maps vocabulary word indices to low dimensional vectors. After transformation of all the words to vectors these are than fed to the convolution layer.

What do the textCNN work

Filters of different sizes and shapes are defined. The shape of the filters use in the proposed model is (2, 3, 4). Total number of filter is 2*3=6. The filters will roll over the original sentence matrix thus reducing it to low dimensional matrix. Instead of training our own embedding make all the sentence matrices of same size and shape.



Embedding lookup function is used to

get the word embedding of the sentence.

The matrix generated as a result of embedding layer is than padded to equalize all the sentences. The defined filters will than start reducing the matrix and generate convolved features. These convolved features are than further reduced. The output generated as a result of convolved features is than spread over the max pooling layer for further down sampling of output.

CNNs model code:

```
[ ]  # Add layers
     textCNNModel = Sequential()
     textCNNModel.add( layers.Embedding( num_words, embedding_dim, input_length=sequence_length ) )

     textCNNModel.add( layers.Conv1D( 128, 5, activation='relu' ) )
     textCNNModel.add( layers.GlobalMaxPooling1D() )
     textCNNModel.add( layers.Dense( 256, activation='relu' ) )
     textCNNModel.add( layers.Dense(1 , activation='sigmoid' ) )
     textCNNModel.compile( optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'] )
     textCNNModel.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 100, 300)          4643400

 conv1d (Conv1D)             (None, 96, 128)           192128

 global_max_pooling1d (Globa  (None, 128)              0
 lMaxPooling1D)

 dense (Dense)               (None, 256)               33024

 dense_1 (Dense)             (None, 1)                 257

=================================================================
Total params: 4,868,809
Trainable params: 4,868,809
Non-trainable params: 0
_________________________________________________________________
```

Explain CNNs model code:

- The first layer added is an Embedding layer, which maps each word in the input sequence to a high-dimensional vector epresentation.
- The second layer added is a Conv1D layer with 128 filters of size 5, which performs convolutions over the sequence of embedding vectors to extract features. The activation parameter sets the activation function for the layer to ReLU.
- The third layer added is a GlobalMaxPooling1D layer, which applies global max pooling over the output of the previous layer to obtain a fixed-length feature vector.
- The fourth layer added is a Dense layer with 256 units and ReLU activation, which performs a linear transformation on the feature vector.
- The fifth layer added is a Dense layer with a single unit and sigmoid activation, which produces a probability output for binary classification

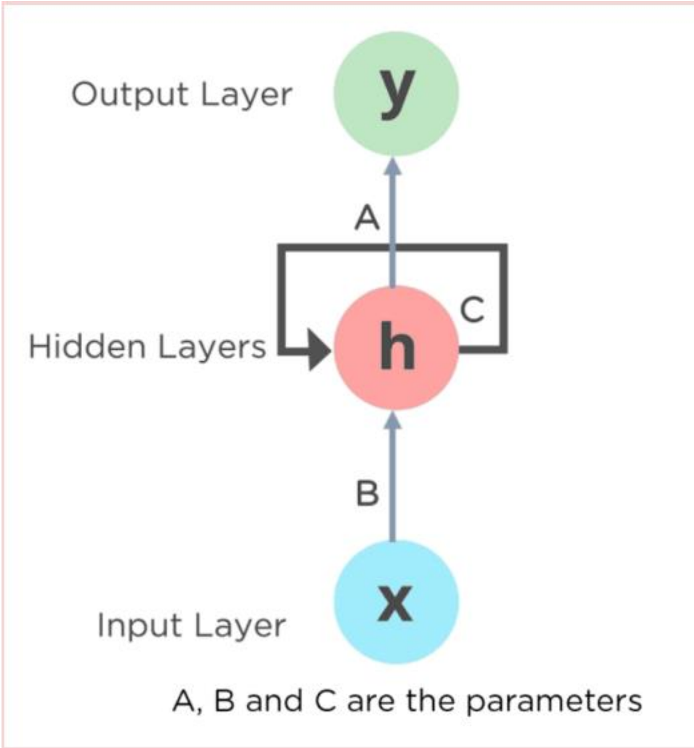
```
Epoch 1/100
56/56 - 29s - loss: 0.5027 - accuracy: 0.7637 - val_loss: 0.4110 - val_accuracy: 0.8244 - 29s/epoch - 520ms/ste
Epoch 2/100
56/56 - 11s - loss: 0.3199 - accuracy: 0.8674 - val_loss: 0.3838 - val_accuracy: 0.8395 - 11s/epoch - 196ms/ste
Epoch 3/100
56/56 - 9s - loss: 0.1946 - accuracy: 0.9236 - val_loss: 0.4764 - val_accuracy: 0.8263 - 9s/epoch - 152ms/step
Epoch 4/100
56/56 - 8s - loss: 0.1016 - accuracy: 0.9641 - val_loss: 0.5498 - val_accuracy: 0.8282 - 8s/epoch - 141ms/step
Epoch 5/100
56/56 - 5s - loss: 0.0548 - accuracy: 0.9813 - val_loss: 0.6522 - val_accuracy: 0.8313 - 5s/epoch - 91ms/step
Epoch 6/100
56/56 - 6s - loss: 0.0379 - accuracy: 0.9872 - val_loss: 0.7898 - val_accuracy: 0.8232 - 6s/epoch - 109ms/step
Epoch 7/100
56/56 - 5s - loss: 0.0276 - accuracy: 0.9894 - val_loss: 0.8220 - val_accuracy: 0.8206 - 5s/epoch - 84ms/step
Epoch 8/100
56/56 - 4s - loss: 0.0232 - accuracy: 0.9920 - val_loss: 0.9160 - val_accuracy: 0.8244 - 4s/epoch - 77ms/step
Epoch 9/100
56/56 - 5s - loss: 0.0198 - accuracy: 0.9926 - val_loss: 0.9658 - val_accuracy: 0.8257 - 5s/epoch - 84ms/step
Epoch 10/100
56/56 - 3s - loss: 0.0165 - accuracy: 0.9938 - val_loss: 1.0706 - val_accuracy: 0.8244 - 3s/epoch - 49ms/step
Epoch 11/100
56/56 - 3s - loss: 0.0165 - accuracy: 0.9943 - val_loss: 0.9968 - val_accuracy: 0.8257 - 3s/epoch - 51ms/step
Epoch 12/100
56/56 - 2s - loss: 0.0163 - accuracy: 0.9943 - val_loss: 1.0541 - val_accuracy: 0.8238 - 2s/epoch - 31ms/step
```

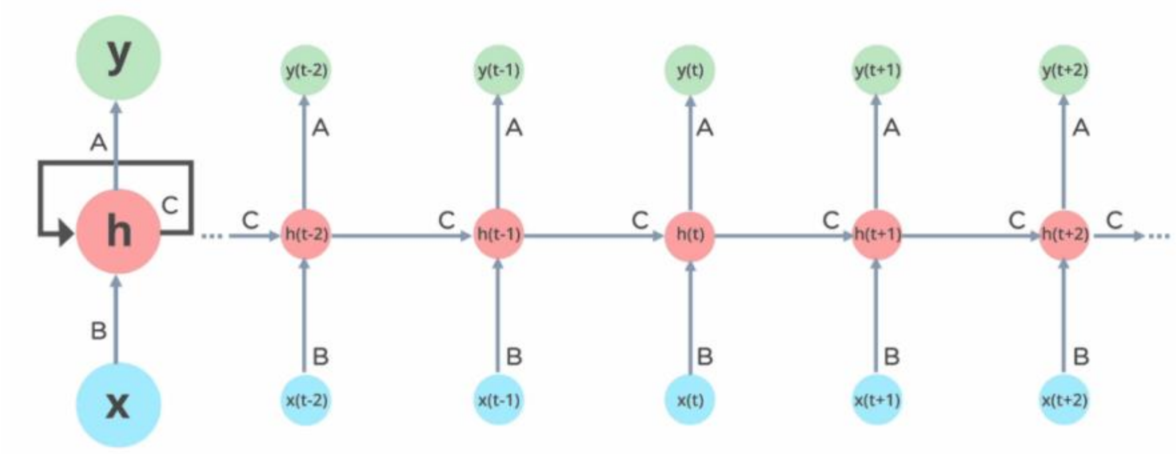
LSTMs - (Long Short-Term Memory Networks)

LSTM (Long Short-Term Memory) is a type of Recurrent Neural Network (RNN) architecture that is commonly used in Natural Language Processing (NLP) tasks

In an LSTM-based NLP classification model, the input text is typically fed into the network as a sequence of word embeddings (vector representations of words) or character embeddings (vector representations of individual characters). The LSTM network processes the input sequence and produces a final output, which is then used to predict the class or label of the input text.

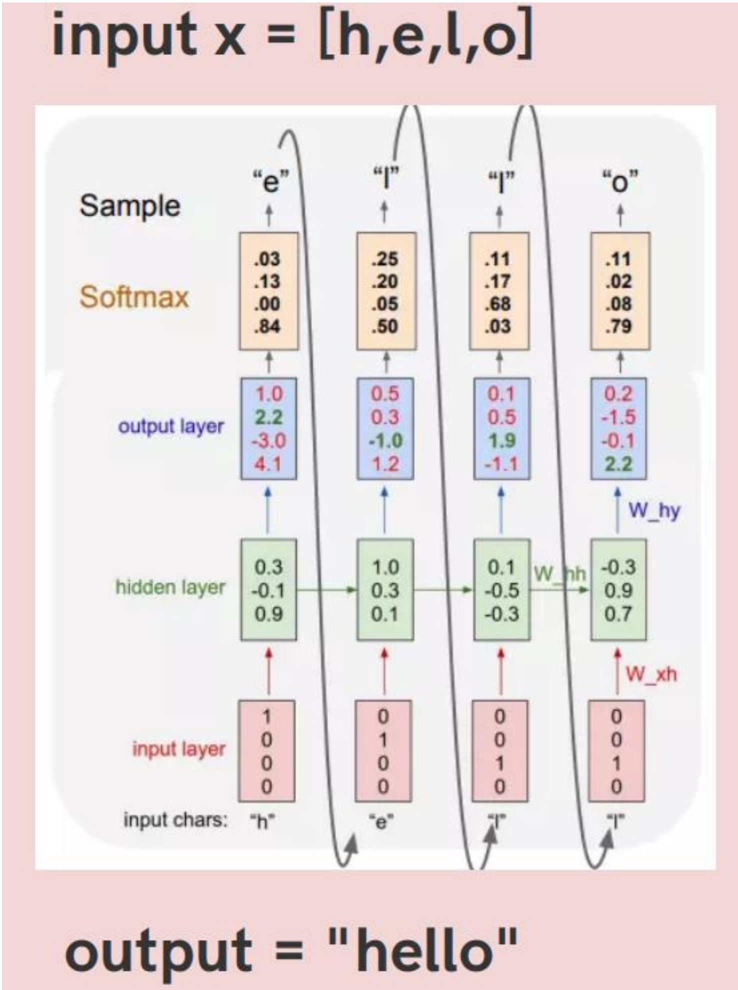
(RNNs) Recurrent Neural Network



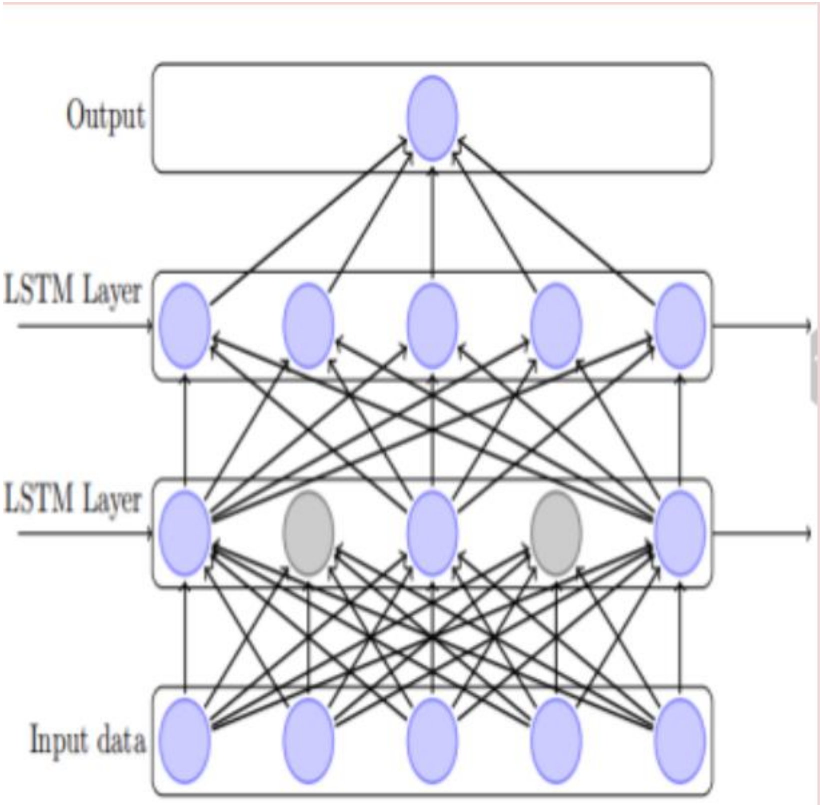


RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.

RNNs see that the first character is "h", so RNNS deduce that the next most probable letter is "e". "E" then becomes the input for the next group of letters, ... and so on until a meaningful word is formed, in this case, the word "hello".



Long Short-Term Memory Networks (LSTMs)
LSTMs work by introducing a special memory cell that is able to selectively remember or forget information over time, depending on the context of the input sequence. At each time step, the LSTM takes as input the current input element (eg. word embedding) and the previous hidden state, and produces a new hidden state and an output.



The key components of an LSTM are:

- Input gate: determines which information to store in the memory cell based on the current input.
- Forget gate: determines which information to discard from the memory cell based on the previous hidden state and current input.
- Output gate: determines which information to output based on the current input and the current memory cell state.
- Memory cell: a special unit that can store and update information over time.

LSTMs model code:

```
# LSTM
lstmModel = Sequential()

lstmModel.add( layers.Embedding( num_words, embedding_dim, input_length=sequence_length ) )
lstmModel.add( layers.SpatialDropout1D(0.2) )

lstmModel.add( layers.LSTM( 100, return_sequences=False ) )
lstmModel.add( layers.Dense( 512, activation="relu" ) )
lstmModel.add( layers.Dense( 1, activation="sigmoid" ) )

lstmModel.compile( optimizer="adam", loss="binary_crossentropy", metrics=['accuracy'])
lstmModel.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 100, 300)          4643400

 spatial_dropout1d (SpatialD  (None, 100, 300)         0
 ropout1D)

 lstm (LSTM)                 (None, 100)               160400

 dense_2 (Dense)             (None, 512)               51712

 dense_3 (Dense)             (None, 1)                 513

=================================================================
Total params: 4,856,025
Trainable params: 4,856,025
Non-trainable params: 0
_________________________________________________________________
```

Explain LSTM code:

- lstmModel.add(layers.Embedding(num_words, embedding_dim, input_length=sequence_length)): adds an embedding layer to the model that will convert the integerencoded input sequences into dense vectors of fixed size.
- lstmModel.add(layers.SpatialDropout1D(0.2)): adds a dropout layer to the model that will randomly drop out 20% of the inputs to prevent overfitting.
- lstmModel.add(layers.LSTM(100, return_sequences=False)): adds a Long Short-Term Memory layer to the model with 100 memory units. The return_sequences=False argument means that the LSTM will only return the last output in the output sequence, rather than the full sequence.
- lstmModel.add(layers.Dense(512, activation="relu")): adds a fully connected layer with 512 units and ReLU activation function.
- lstmModel.add(layers.Dense(1, activation="sigmoid")): adds a final output layer with a single unit and sigmoid activation function, which will output a probability value between 0 and 1 indicating the predicted class.

Execute model:

```
[56] lstmModel.fit( train_X, train_y,
        epochs=50,
        verbose=2,
        validation_data=(dev_X, dev_y),
        batch_size=batch_size)

Epoch 1/50
56/56 - 12s - loss: 0.5079 - accuracy: 0.7657 - val_loss: 0.4234 - val_accuracy: 0.8250 - 12s/epoch - 207ms/step
Epoch 2/50
56/56 - 8s - loss: 0.3279 - accuracy: 0.8622 - val_loss: 0.4055 - val_accuracy: 0.8307 - 8s/epoch - 145ms/step
Epoch 3/50
56/56 - 5s - loss: 0.2402 - accuracy: 0.9057 - val_loss: 0.4825 - val_accuracy: 0.8175 - 5s/epoch - 86ms/step
Epoch 4/50
56/56 - 6s - loss: 0.1812 - accuracy: 0.9296 - val_loss: 0.5502 - val_accuracy: 0.8150 - 6s/epoch - 112ms/step
Epoch 5/50
56/56 - 4s - loss: 0.1408 - accuracy: 0.9465 - val_loss: 0.6090 - val_accuracy: 0.8087 - 4s/epoch - 74ms/step
Epoch 6/50
56/56 - 3s - loss: 0.1169 - accuracy: 0.9557 - val_loss: 0.7012 - val_accuracy: 0.8062 - 3s/epoch - 58ms/step
Epoch 7/50
56/56 - 6s - loss: 0.1057 - accuracy: 0.9592 - val_loss: 0.7778 - val_accuracy: 0.8062 - 6s/epoch - 107ms/step
Epoch 8/50
56/56 - 3s - loss: 0.0980 - accuracy: 0.9606 - val_loss: 0.8413 - val_accuracy: 0.8055 - 3s/epoch - 55ms/step
Epoch 9/50
56/56 - 2s - loss: 0.0858 - accuracy: 0.9654 - val_loss: 0.9110 - val_accuracy: 0.8055 - 2s/epoch - 43ms/step
Epoch 10/50
56/56 - 3s - loss: 0.0772 - accuracy: 0.9685 - val_loss: 0.9598 - val_accuracy: 0.8024 - 3s/epoch - 48ms/step
Epoch 11/50
56/56 - 4s - loss: 0.0726 - accuracy: 0.9702 - val_loss: 0.9668 - val_accuracy: 0.8005 - 4s/epoch - 66ms/step
Epoch 12/50
56/56 - 2s - loss: 0.0717 - accuracy: 0.9711 - val_loss: 1.0244 - val_accuracy: 0.8074 - 2s/epoch - 36ms/step
```

Compare:

The comparison of Thanh Son Luu and his team

Model	Accuracy (%)		F1-macro (%)	
	Task 1	Task 2	Task 1	Task 2
Text-CNN	84.18	83.42	77.89	64.74
LSTM	82.97	83.35	77.24	66.58
GRU	83.50	82.84	77.67	66.51
PhoBERT	**90.01**	**88.93**	**86.89**	**72.17**
BERT4News	86.39	86.20	86.16	62.62

Looking at this table we can probably see that PhoBert - the transformer structure is the best model for detecting spam labels with an accuracy of 90.01 and 86.89 f1 for the first task. on the other hand, CNNs model is the best deep learning for this solution. 84.14 and 77.89 corresponding accurancy and F1 score.

Similar to that, our CNN model performance accuracy more than LSTMs.

Model	Accuracy
textCNNs	0.820015
LSTMs	0.81094

For these reasons, we decided to choose CNNs as the solution for this project

Deployment

After developing the detection model i came up with 2 ways to apply it:

- One way to use this detecting model is to stop spam reviews from appearing on the platform. When people write reviews, the AI system can quickly check the content to see if it's spam. If it is, the system will block it and prevent it from being shown to others. The system can also keep track of people who repeatedly post spam reviews and stop them from posting more.
- Another way to use your detecting model is to gather comments from specific product and analyze them for spam. The system can look at comments on product pages, forums, and social media. It will use special techniques to figure out if a comment is spam or not. If it's spam, the system will remove it and create a report that shows how much spam there is. This report can help you make decisions about whether to continue developing a product or cancel it. If there's a lot of spam comments about a product, it might mean there are problems with it, and you might need to make changes or be more careful.

Collect data, filter and reports

In this approach, detecting model can be use to filter spam datas for dataset of dashboard analyze. We can collect comments from various sources, such as product pages, forums, or social media platforms, and apply NLP techniques to detect spam comments.

This solution supports information analysts about the products the company is selling

Crawl data from Shopee platforms

After we collect the information, we have a set of data to analyze.

take a look at the sample header of the dataset

```
In [8]: df.head(7)
```

Out[8]:

	Unnamed: 0	Username	Rating	Posted At	Product Categories	Review Text	Shop's response	Likes	Page	Class	Probability
0	0	h*****a	5	2021-10-26 10:34	bạn mới nhập coshbm 50 đơn 0đ thực phẩm chức ...	Hàng giao đúng như trong mô tả, đóng gói đẹp, ...	NaN	0	1	0.0	0.000003
1	1	ngocshina	5	2022-04-26 16:29	bạn mới nhập coshbm 50 đơn 0đ thực phẩm chức ...	Chưa dùng nên chưa biết hiệu quả. Giao hàng nh...	NaN	0	1	0.0	0.000009
2	2	t*****9	5	2022-02-16 21:19	bạn mới nhập coshbm 50 đơn 0đ thực phẩm chức ...	Sản phẩm giao hàng rất nhanh, chất lượng tốt c...	NaN	0	1	1.0	1.000000
3	3	hong_minh79	5	2023-02-14 22:21	bạn mới nhập coshbm 50 đơn 0đ thực phẩm chức ...	Công dụng: giải rượu\nMình đã mua lần 2. Uống ...	Rohto Mentholatum Việt Nam xin chào bạn, Rohto...	0	1	0.0	0.087927
4	4	trangtranggggg144	5	2021-10-22 09:32	bạn mới nhập coshbm 50 đơn 0đ thực phẩm chức ...	Hàng ok, đóng gói cẩm thận k bị vỡ. Chất lượng...	NaN	0	1	0.0	0.525038
5	5	thanhluan8688	5	2022-05-04 20:22	bạn mới nhập coshbm 50 đơn 0đ thực phẩm chức ...	Đã mua lần 2, công dụng giải rượu rất tốt\nSẽ ...	NaN	0	1	1.0	1.000000
6	6	getcaimat	5	2023-04-03 20:17	bạn mới nhập coshbm 50 đơn 0đ thực phẩm chức ...	Đối tượng sử dụng: người lớn\nCông dụng: giải ...	NaN	0	2	1.0	0.999985

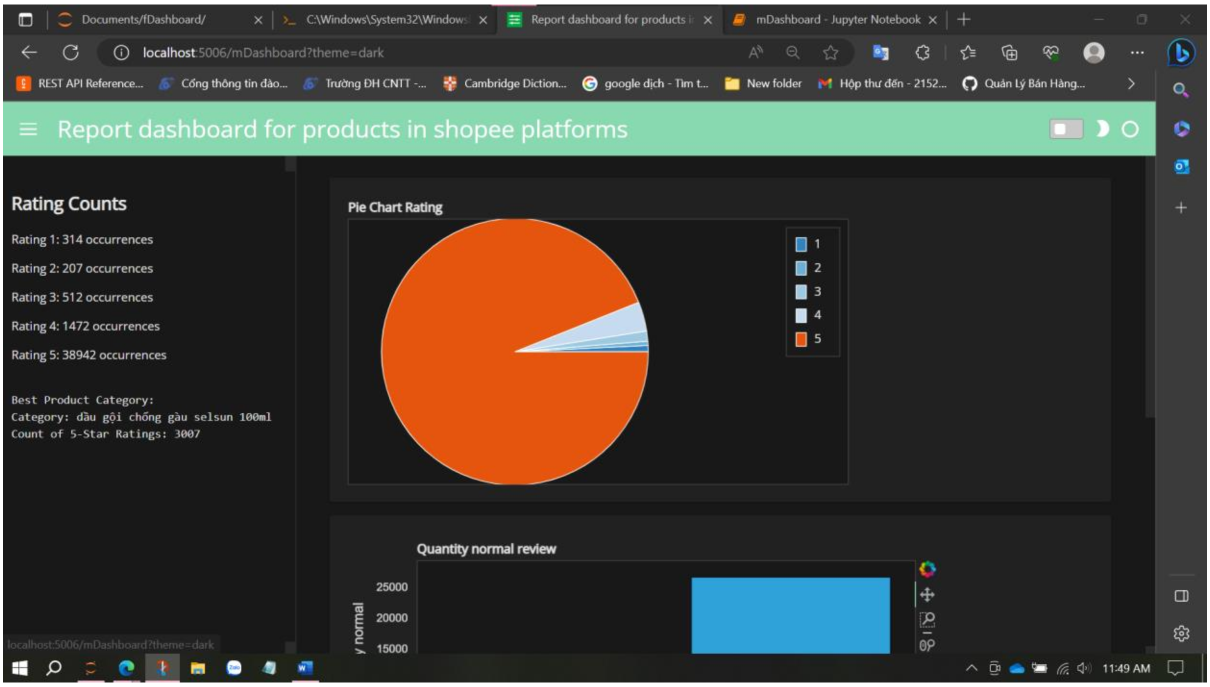
Explain dataset:

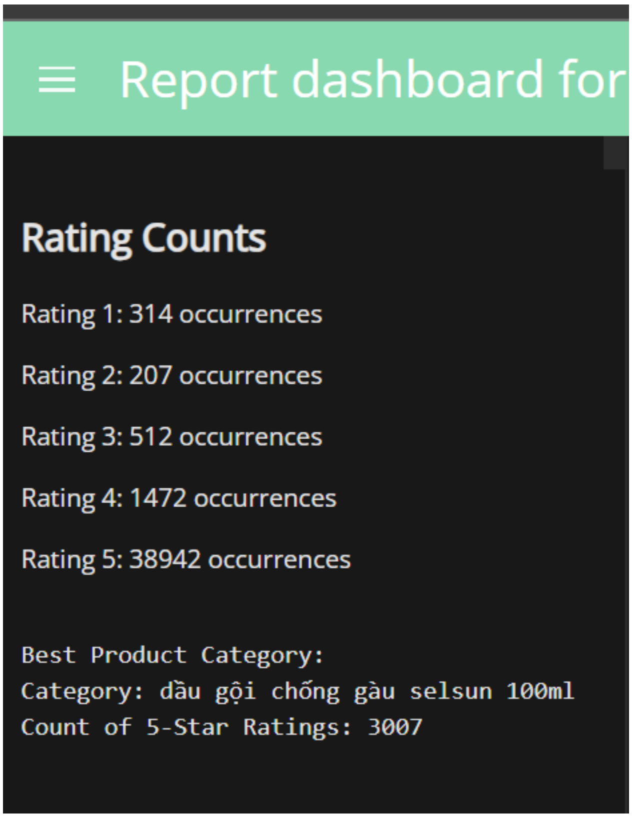
The dataset have 10 columns:

Column name	Explain
Username	This column contains the name of the reviewer
Rating	Just contains a rating from 1 to 5 stars
Posted At	This column contains time of the review publicized
Product Categories	The name of the product for which we have collected information
Review Text	Review content, it can be empty
Shop's Response	Review content reply user review, it can be empty
Like, Page	Just is some information of review
Class	It is label of review contain. Simuilar to base the dataset class 1 is spam and 0 is not
Probability	Probability is a most important value to decide spam or not. if probability is greater than 0.7 then review as spam and label is 1

Dashboard report

Dashboard built by python using jupyter notebook tool and necessary libraries that list is hvplot, pandas, numpy and more.





In the sidebar we have some values about the company's product

Example: 38942 total for 5 stars rating, 1472 for 4 stars rating

The best product with the most interactions is “dầu gội chống gàu selsun 100ml” with 3007 total for 5 stars rating. That product could become the company's flagship product, …

Product with the most spam reviews:: dầu gội chống gàu selsun 100ml

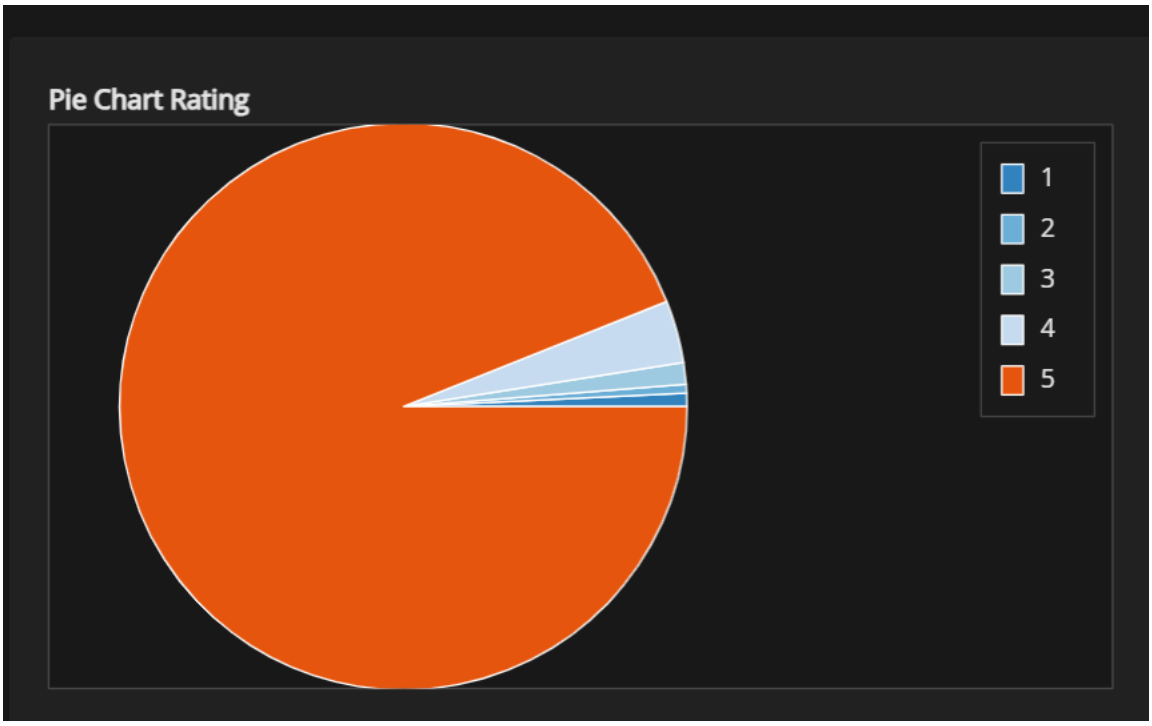
Number of spam reviews for the product:: 854

Product with the least spam reviews:: cosrx official kem chống nắng cosrx dưỡng da vitamin e spf 50 50ml

Number of spam reviews for the product:: 2

We also found the product with the most spammy reviews was “dầu gội chống gàu selsun 100ml” with 854 spam reviews. I suggest the company needs to re-evaluate the product.

On another hand “cosrx official kem chống nắng cosrx dưỡng da vitamin e spf 50 50ml” has the least spam review with 2



The pie chart tells us the rating percentage.

The company has a good interaction for all products. It's a good thing

Comparing the 2 charts, we can see that the majority of reviews are concentrated at 4 and 5 stars, and normal reviews account for the majority with 2 times spam reviews.

Wordpress Spam Detection Reviews Plugin

This section describes the functionalities of the Spam Detection Reviews Plugin in Wordpress.

Very Awesome Comments Analyzer (VACA) is a plugin that combines the product reviews data from WooCommerce with the in-house Spam Detection Solution in order to analyze and detect whether a product review is spam or not.

The plugin first modifies the wprt_comments table, which is used by WooCommerce to manage Post comments/Product reviews by adding 4 columns

- vaca_is_spam: Indicates whether that review is spam or not.
- vaca_evaluated_score: The accuracy returned from the model which is used to detect whether a review is spam or not.
- vaca_evaluated_at: The point in time where a review is being evaluated through the Machine Learning model.
- vaca_notes: Additional notes.

Next, the plugin retrieves all reviews that have not yet been evaluated for spamming and perform evaluation where each record is sent to a separate API containing the logic to predict a target is spam or not. When the batch operation has completed, the plugin stores the generated scores corresponding to each record from the previous step into the wprt_comments table.

The plugin also listens for insert/update operations on the wprt_comments table, which happens whenever a customer posts a product review, and then applies the same processing flow as described above.

Benefits to E-commerce

The VACA plugin helps shop owners to identify and filter out spam reviews from their products, aid them in making decisions to approving the reviews or comments. The plugin is also expected to reduce the amount of time for manually filtering out spam comments, aiming for a healthier e-commerce platform as we have an automated process.

Limitations of the plugin

- As of now, VACA has yet to automatically batch process and update the spam attributes on each product review on a regular basis.
- The amount of information displayed on the main table of the plugin is quite limited, as there is no way to directly show or filter all reviews for a particular product on the main plugin page.
- The detection model lacks any capability to self-train based on the new data during its use to analyze product reviews, possibly leading to biased/outdated predictions.

The influence of our model and its application to ecommerce

Improved decision-making: The dashboard report generated by your model can provide valuable insights into the sentiment and nature of customer reviews. This information can help companies make informed decisions about product improvements, customer service enhancements, and marketing strategies.

Competitive advantage: By leveraging the power of deep learning to analyze customer reviews, your model can give the company a competitive edge. The ability to understand customer sentiments and preferences can help the company tailor their offerings to meet customer expectations more effectively.

Enhanced customer satisfaction: By identifying and addressing issues highlighted in customer reviews, the company can improve customer satisfaction and loyalty. Understanding customer feedback can guide them in resolving product issues, improving customer support, and addressing any concerns promptly.

Improved review quality: By filtering out spam reviews, your system can enhance the overall quality of reviews on the ecommerce platform. This ensures that genuine and relevant reviews are more prominently displayed, providing users with more reliable information when making purchasing decisions.

Protection against fraud: Spam reviews often include false information, biased opinions, or fake endorsements. By identifying and banning such reviews, your system helps protect customers from misleading information and fraudulent activities.

Trust and credibility: By maintaining a high standard for reviews, the ecommerce platform gains trust and credibility among users. Users are more likely to rely on the platform's recommendations and engage in more confident purchasing decisions.

Task name	Le Nguyen Lam Lam	Tran Anh Thy
Research	x	x
Find dataset and references	x	x
Data Clearning	x	x
Data preprocessing	x	x
Embadding	x	x
Execute the model	CNNs	LSTMs
Website crawler	x	x
Dashboard	x	
API		x
Filter & Ban spam feedback		x
Slide	x	x
Report	x	x
Presentation	x	x
Contribution rate to the results of the project	50	50