Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

CLUSTERING VIETNAMESE CONVERSATIONS FROM FACEBOOK PAGE TO BUILD TRAINING DATASET FOR CHATBOT

Trieu Hai Nguyen, Thi-Kim-Ngoan Pham, Thi-Hong-Minh Bui and Thanh- Quynh-Chau Nguyen

(Received: 26-Sep.-2021, Revised: 10-Dec.-2021, Accepted: 28-Dec.-2021)

ABSTRACT

The biggest challenge of building chatbots is training data. The required data must be realistic and large enough to train chatbots. We create a tool to get actual training data from Facebook messenger of a Facebook page. After text preprocessing steps, the newly obtained dataset generates FVnC and Sample dataset. We use the Retraining of BERT for Vietnamese (PhoBERT) to extract features of our text data. K-Means and DBSCAN clustering algorithms are used for clustering tasks based on output embeddings from PhoBERT$_{base}$. We apply V-measure score and Silhouette score to evaluate the performance of clustering algorithms. We also demonstrate the efficiency of PhoBERT compared to other models in feature extraction on the Sample dataset and wiki dataset. A GridSearch algorithm that combines both clustering evaluations is also proposed to find optimal parameters. Thanks to clustering such a number of conversations, we save a lot of time and effort to build data and storylines for training chatbot.

KEYWORDS

BERT, Clustering, Language models, Feature extraction, Word embeddings.

1. INTRODUCTION

Chatbot is certainly not an unfamiliar name in the field of Natural Language Processing (NLP). Previously, traditional chatbot simply interacts with the user *via* predefined rules, which means that the user is only allowed to enter by these rules to get answers. However, NLP chatbot is not only a word recognition algorithm, but it can also understand what the user is saying. It is one of the pioneering applications using Artificial Intelligence (AI); namely, NLP, to help humans interact with machine like humans with humans via Virtual Assistant autoresponder. Currently, there are many parties developing NLP chatbot, which can be mentioned as Google's DialogFlow, Watson of IBM and Rasa.

In the process of building NLP chatbots, all chatbots require real datasets for training bot. The training datasets can be large or small depending on the size and intelligence level of the chatbots. Raw training data can be collected from past conversations through social media, archived user chats, previous questions, email chains or live telephone transcripts. But, these data are messy, not in any structure or order and come from various sources collected with huge amounts of raw data. Thus, the first priority when constructing a chatbot is to transform those raw data into useful data for the purpose of training bot.

In order to fit chatbot building orientation, that raw dataset needs to be divided into specific intents, which serves to build conversations to train a chatbot. There are many ways to process that raw dataset into specific intents (topics, conversations). The first method to be mentioned is using Supervised Learning task [1] to classify intents. In particular, this method requires labeling for the input examples and then it predicts labels for remaining data in the raw dataset. In this case, label prediction corresponds to classified raw data into the intents that have been labeled previously. However, building and labeling manual intents on large datasets lead to big challenges for chatbot developers. With a simple raw dataset of about 8000 conversations, analyzing how many intents are created is a conundrum.

Instead, we can approach the above problem by using the second method, which is clustering similar raw data together into corresponding intents. The advantage of this approach is that it uses

Trieu H. N. (ORCID: 0000-0002-6086-9598), T-K-Ngoan P., T-H-Minh B. and T-Q-Chau N. are with Faculty of Information Technology, Nha Trang University, Nha Trang City, Vietnam. Emails: {trieunh, ngoanptk, minhbth, chauntq}@ntu.edu.vn.

Unsupervised Learning technique [1], which is only based on the features of the data to perform specific tasks such as clustering. As expected, this method has a significant effect on the analysis of raw data. It saves us a lot of time and effort to make a training dataset for chatbot. There are many clustering algorithms, such as K-Means, DBSCAN, BIRCH and Spectral clustering [2]-[3]. In this article, we use K-Means [4] and DBSCAN [5] techniques to cluster our dataset and consider that clustering is a downstream NLP task. Each technique has its own advantages and disadvantages. K-Means algorithm is a simple and fast-implementation algorithm, but it requires knowing the number of clusters to perform clustering whereas DBSCAN does not. Nevertheless, DBSCAN technique is more difficult to implement and requires finding the optimal parameters [5], [6], [7], which leads to drastically increased costs, especially for large datasets. Thus, we can combine the advantages of both techniques to serve the purpose of efficient clustering.

The input of clustering algorithms in particular and downstream NLP tasks in general is document embeddings extracted from the dataset. There are many ways to extract information from text datasets; for example, we can use traditional machine learning algorithm like TF-IDF [8], proposed word embedding models in recent years such as Word2Vec, GloVe [9]-[10], FastText [11] or popular language models like GPT-2 [12], BERT model and its variations [13], [14], [15]. In this work, we use BERT (Bidirectional Encoder Representations from Transformers), which is state-of-the-art embeddings [13] to extract features of documents. Recently, a clustering approach with the BERT model has been proposed by O. Gencoglu [16]. As suggested in [17], clustering techniques using pre-trained transformer language models are applied to short text clustering. The combination of word embeddings using BERT models and clustering algorithms to obtain topics was presented in [18]. Distinctively, PhoBERT represents pre-trained language models for Vietnamese, being used to embed our Vietnamese dataset [15]. V-measure score [19] and Silhouette score [20] are used to evaluate the performance of clustering algorithms as well as the feature extraction efficiency of the language models.

The aim of the present paper is to study and apply PhoBERT model to our Facebook Vietnamese conversations dataset, thereby deriving document embeddings in order to serve the clustering task. The combination of both K-Means and DBSCAN clustering algorithms is proposed by us to achieve the best clustering results on the actual dataset. The finding of these data clusters allows us to simplify and accelerate the building of a training dataset for chatbot. In Section 2, we recall some theories of Transformer and BERT architecture proposed by Vaswani et al. in [21] and Devlin et al. in [13], respectively. In Section 3, we offer an approach to apply PhoBERT to the clustering task from the idea of classification task [15], [22]. Next, we also recall clustering algorithms in machine learning and evaluation metrics for unsupervised learning algorithms in Section 4. Some experiments on our Facebook Vietnamese conversations dataset (including FVnC and sample dataset) and wiki dataset, such as searching optimal parameters, clustering performance evaluations as well as clustering results, are considered in Section 5. In particular, we show that among the models that support Vietnamese, PhoBERT's feature extraction efficiency is the best based on V-measure score. Ultimately, we give some conclusions in Section 6. The code, datasets and pre-trained models are available at https://github.com/trieuntu/conversation_ clustering.

2. RELATED WORK

We provide some background knowledge about Transformer architecture, Pre-Trained Language Models, especially BERT. From these theoretical constructs, we apply them to solve our NLP tasks.

2.1 Transformer

Transformer architecture was first introduced in the paper "Attention Is All You Need" by [21]. At the time of launch, this architecture was considered a new breakthrough in the field of natural language processing and related tasks. Currently, when dealing with sequence-to-sequence models in NLP, the transformer is still one of the state-of-the-art (SOTA) types of model and completely replaces RNN/LSTM [23]. Transformer architecture overcomes the disadvantages of RNN and its variations. For instance, it doesn't take advantage of GPU parallelism, because it has to process input word-by-word sequentially into encoder/decoder and the information is easily lost during propagation through hidden layers for long input sentences.

Transformer architecture contains two parts; Encoder attention and Decoder attention. According to the original article of [21], the encoder part has 6 layers, each of which has two sublayers, which are multi-head self-attention and fully connected feed-forward. Decoder part is similar to the encoder part, but it adds a masked multi-head attention sublayer and the last layer of the encoder part will be passed to the multi-head attention sublayer in the decoder part. Note that the input of both parts is the sum of positional encoding vector and word vector embedding.

The attention mechanism is the most important component of transformer architecture. Self-attention sublayer is an attention mechanism, which contains the weight sets of the model W_q, W_k, W_v to be trained. The attention mechanism presents the relation of a word to all its related words in the sequence based on the adjustment of the above sets of weights. The product of the input embedding layer and W_q, W_k, W_v is matrices Query Q, Value V and Key K. In order to calculate Attention vector of word *i* to the rest of the words, Vaswani et al. [21] have given the formula:

$$Attention_i(Q_i, K_i, V_i) = softmax\left(\frac{Q_i K_i^T}{\sqrt{d_{K_i}}}\right).V_i$$

where d_K is the dimension of *K*. Each computed *Attention* obtains a head-attention. We can compute the *Attention* in parallel, which leads to the multi-head attention mechanism by concatenating head-attentions:

$$MultiHead(Q,K,V) = Concatenate(head_1, head_2, \ldots, head_n).W_O$$

where $head_1$ corresponds to $Attention_1$. Matrix W_O has the same number of columns as the input matrix.

2.2 Pre-trained Language Models: BERT

Training models from scratch on large datasets is impossible for most people. Thus, using pre-trained models is an inevitable trend in the development of Artificial Intelligence. Taking into account the advantage of the weights that can be learned from trained models, we just need to fine-tune them to suit specific purposes. Formerly, pre-trained models in NLP have been mentioned in many studies [9]-[10], [24][25][26]. One of the great advantages of the transformer's architecture is that it allows the creation of NLP models trained, which can be reused in downstream NLP tasks. Some of the pre-trained language models based on transformer's architecture have achieved state-of-the-art results, like BERT of [13] from Google, GPT of Radford and Narasimhan [27] from Open AI and their variations. These new models can do things that the old models can't, such as allowing transfer learning in NLP with both low- and high-level features. Transfer learning is a combination of reusing the architecture of pre-trained model and fine-tune parameters of the original layers to accommodate downstream tasks.

Specifically, BERT is an easily fine-tuned pre-train word embedding on a large unlabelled text corpus (unsupervised) which is trained based on Masked Language Model Task and Next Sentence Prediction Task. BERT's architecture is built only on the Encoder part of the Transformer. The input text before applying fine-tuning for Vietnamese in particular and other languages in general is a combination of Token Embeddings, Segment Embeddings and Position Embeddings. If the input text consists of two or more sentences (pair-sequence), we must add token [CLS] at the beginning of the sentence and token [SEP] to separate the sentences.

Masked Language Model task allows us to fine-tune word representations on any unsupervised text corpus. This task creates embeddings for the above Vietnamese dataset. The principle of operation of model training can be understood by predicting a missing word in the sequence instead of trying to predict the next word in the sequence itself. A missing word is equivalent to [MASK] token. We randomly mask 15% of the total tokens in the sequence and predict these [MASK] tokens. Note that a missing word can be replaced by [MASK] token 80% of the time, 10% of the time for a random token and 10% of the time for the unchanged token.

Next Sentence Prediction (NSP) is a binary classification task applied practically to the Question Answering (QA) task. NSP helps us understand the relationship between sentences. The input of the model is a pair-sequence, which has been added tokens [CLS], [SEP]. During model training, we select 50% of the time of the second sentence, which is the next sentence of the first one and labeled as IsNext, while the remaining 50% of the second sentence is randomly chosen from unrelated sentences in the dataset and labeled as NotNext.

There are many versions of BERT with different parameters on transformer architectures. The two most

basic models are $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$. In essence, both models are the same, but they are different in size. Specifically, according to Devlin et al. [13], these models have the following sizes:

$$\text{BERT}_{BASE}(L=12, H=768, A=12, Total\ Parameters=110M),$$
$$\text{BERT}_{LARGE}(L=24, H=1024, A=16, Total\ Parameters=340M)$$

where L is the number of layers in the Encoder part of transformer architecture, H is the hidden size and A is the number of heads in multi-head self-attention.

3. PHOBERT FOR TEXT CLUSTERING

PhoBERT represents pre-trained language models for Vietnamese proposed by Nguyen and Nguyen [15]. At the time of launch, pre-trained PhoBERT models established state-of-the-art results in most tasks related to Vietnamese NLP. Although BERT can be applied to many tasks, like Classification, Clustering, Dependency parsing, Sentiment analysis, Summarization text, Part-of-speech tagging, Question Answering, Named-entity recognition and Machine translation, in this work, we only focus on clustering task to analyze our Vietnamese conversations dataset.

$\text{PhoBERT}_{\text{base}}$ and $\text{PhoBERT}_{\text{large}}$ are two versions of PhoBERT, whose architectures are similar to the $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ above. PhoBERT uses RoBERTa, which is based on pytorch framework [28] to retrain the BERT models on new 20GB pre-training Vietnamese dataset. Since PhoBERT architecture is based on RoBERTa, it only trains BERT model with Masked Language Model task. Another difference between PhoBERT and RoBERTa is fastBPE used to tokenize input sentences. Currently there are many methods to tokenize, such as Word Level Tokenizer, Multi-Word-Level Tokenizer, Character Level Tokenizer, Subword Units Level (BPE algorithm) Tokenizer, but only BPE (Byte-Pair Encoding proposed by Sennrich et al. [29]) achieves SOTA and is applied to most modern NLP models.

BPE is a compression technique and is adapted for word segmentation tasks. Most words can be represented by subwords using the BPE method. It overcomes the disadvantages of Word and Character Tokenizers; for instance, words that do not appear in the dictionary can be represented in these subwords and the index length of sequence output is significantly shorter than Character Tokenizers. Code[1] of BPE algorithm to segment word into subword units was published by [29]. For example, assume that the given Vietnamese vocabulary is:

vocab = {'x i n h </w>': 10, 'đ ẹ p </w>': 20, 'x i n h _ đ ẹ p </w>': 10, 'x i n h _ x ắ n </w>': 15, 'x ắ n </w>': 8}

Notice that, unlike English, the Vietnamese language does not use white space to separate words, because Vietnamese words can have more than one syllable. For illustration take a simple Vietnamese sentence "*Cô ấy rất xinh đẹp*" (English version is "*She is very beautiful*"), which can be rewritten in the monosyllable form "*Cô_ấy* $_{She}$ *rất* $_{very}$ *xinh_đẹp* $_{beautiful}$ ". Therefore, we can apply a Multi-Word-Level Tokenizer on the pre-training Vietnamese dataset before going into BPE. There are many toolkits to support word segmentation based on Multi-Word-Level Tokenization, like RDRSegmenter from VnCoreNLP [30], pyvi [31] and underthesea [32]. In the example above, tokens </w> are appended to the end of the words to mark the end of a word in Vietnamese vocabulary. After merging the most frequent pair at the 9^{th} iteration, we obtain a new vocabulary as follows

vocab_{new}={'xinh': 10, '</w>': 10, 'đẹp</w>': 30, 'xinh_': 25, 'xắn</w>': 23}

It is clear that the word '*xinh_đẹp</w>*' can be represented by subwords '*xinh_*' and '*đẹp</w>*' from the above vocab_{new}. Especially, word '*xinh_xinh*' (English meaning is *pretty*) is out of vocabulary words, which can also be represented by the word pair '*xinh_*' and '*xinh*'.

In order to fine-tune PhoBERT for downstream tasks, we can use library packages, such as Transformers of Hugging Face [33] and FAIRSeq of Facebook [34], to implement. We use $\text{PhoBERT}_{\text{base}}$ with 12 block sub-layers of the Encode part to obtain the Embedding vectors as features of input sequences. More specifically, this Embedding vector is an output vector of the first token [CLS] from the final hidden state h (Figure 1). According to the idea of [22] [35], the vector of token [CLS] is the feature of the whole sentence for Classification task. To verify the idea just mentioned

[1] Scripts are available at https://github.com/rsennrich/subword-nmt

earlier, let's consider the following three sentences in Table 1.

Statement 1. *Assume that Embedding vectors* $\boldsymbol{E}^{i}_{[CLS]}$ *and* $\boldsymbol{E}^{j}_{[CLS]}$ *represent the whole sentences i and j, respectively. If sentences i and j are similar, then the Cosine Similarity between* $\boldsymbol{E}^{i}_{[CLS]}$ *and* $\boldsymbol{E}^{j}_{[CLS]}$ *must be sufficiently larger than a certain threshold and converges to 1 with identical sentences.*

Proof. The Cosine Similarity Formula is:

$$Cosine_{Similarity}\left(E^{i}_{[CLS]}, E^{j}_{[CLS]}\right) = \frac{E^{i}_{[CLS]} \cdot E^{j}_{[CLS]}}{\left\|E^{i}_{[CLS]}\right\| \left\|E^{j}_{[CLS]}\right\|} \quad (1)$$

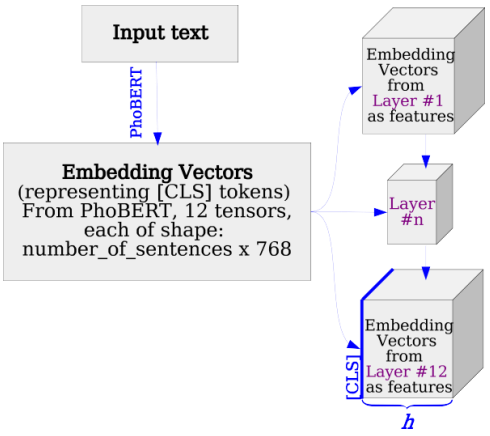

Figure 1. The first token [CLS] as feature of the input sentence.

Table 1. Three Vietnamese sentences are used as examples to extract Embedding vectors with PhoBERT model. Embedding vectors of sentences **A**, **B** and **C** are $E^{A}_{[CLS]}, E^{B}_{[CLS]}$ and $E^{C}_{[CLS]}$, respectively.

Sentence	Embedding Vector	Vietnamese	English
A	$E^{A}_{[CLS]}$	hà_nội là thủ_đô của việt_nam	Hanoi is the capital of Vietnam
B	$E^{B}_{[CLS]}$	thủ_đô của nước chxhcn việt_nam có tên gọi là hà_nội	The capital of the socialist republic of Vietnam is called Hanoi
C	$E^{C}_{[CLS]}$	hôm_nay trời sẽ có mưa dông, gió mạnh	Today there will be thunderstorms and strong winds

The statement above can be easily demonstrated through the example in Table 1. As observed, sentences **A** and **B** are almost similar and have higher similarity over sentence **C**. The computation of cosine similarity between Embedding vectors $E^{A}_{[CLS]}, E^{B}_{[CLS]}$ and $E^{C}_{[CLS]}$ is shown in Table 2. Since **A** and **B** are almost alike, their similarity metric will be high and converge to *1* and *vice versa* for **C**.

Table 2. Computing cosine similarity between embedding vectors for Table 1 with PhoBERT.

Cosine Similarity $(\boldsymbol{E}^{i}_{[CLS]}, \boldsymbol{E}^{j}_{[CLS]})$			
$\boldsymbol{E}^{A}_{[CLS]}, \boldsymbol{E}^{A}_{[CLS]}$	$E^{A}_{[CLS]}, E^{B}_{[CLS]}$	$E^{A}_{[CLS]}, E^{C}_{[CLS]}$	$E^{B}_{[CLS]}, E^{C}_{[CLS]}$
1.0	0.8519334	0.4461445	0.43029878

Using the Embedding vector of token [CLS] as a feature of the whole sentence, we adapt this idea to our Clustering task. The Clustering implementation process with PhoBERT_{base} model is shown in Figure 2. After obtaining the output embeddings of sentences with PhoBERT_{base} model, we use the algorithms K-mean and DBSCAN to cluster our text data. The output embeddings of sentences have the form as follows:

$$E^{i}_{[CLS]} = hW \quad (2)$$

where $W \in \Re^{d,H}$ and h are projection matrices at the linear projection layer and the final hidden state, respectively.

4. CLUSTERING ALGORITHM

K-Means is one of the most basic algorithms in unsupervised learning [3]. According to the K-Means algorithm, a set of N samples $E^i_{[CLS]}$ is divided into K disjoint clusters ($K < N$). Let Y is the set of all label vectors for N samples, i.e., each sample $E^i_{[CLS]}$ has a label vector $y_i = [y_{i1}, y_{i2}, \ldots, y_{iK}] \in Y$. If vector $E^i_{[CLS]}$ belongs to cluster k, then $y_{ik} = 1$ and $y_{ij} = 0, \forall i \neq k$. Each cluster is characterized by a cluster "centroids". A set of centroids is denoted $M = [m_1, m_2, \ldots, m_K]$. In K-means algorithm, the clustering problem will be reduced to the optimization for loss function $\mathcal{L}(Y, M)$.

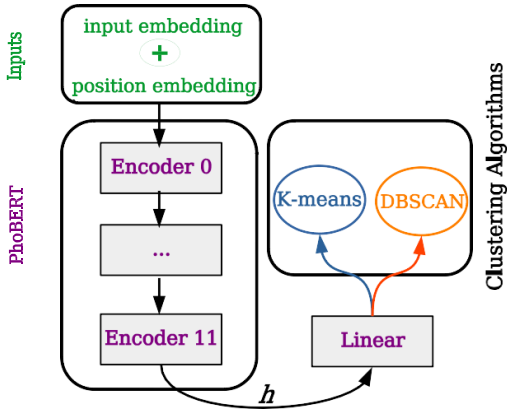

Figure 2. The overall flow for Clustering task with PhoBERT$_{base}$ model, starting from Vietnamese pre-training data, passing the layers Encoder$_{0\rightarrow11}$ to obtain embedding vectors from the final hidden state h through the linear projection layer and finally using unsupervised Learning algorithms K-mean and DBSCAN to cluster the text data.

$$\mathcal{L}(Y, M) = \sum_{i=1}^{N} \sum_{j=1}^{K} y_{ij} \left\| E^i_{[CLS]} - m_j \right\|^2 \quad (3)$$

Along with that, we also apply DBSCAN technique to cluster data points [5]. In real-life data, DBSCAN can work well for nonconvex clusters with arbitrary shapes and noises. DBSCAN algorithm focuses on radius *eps*–ε and the minimum number of neighbors required to create a cluster *minPts*. Radius *eps* defines a circle for each point to determine its neighbors. A point becomes *core point* if the circle surrounding this point with radius *eps* contains more than *MinPts* neighbors. In case that the number of neighbor points is less than *MinPts*, the *core point* is the *border points*. On the other hand, a point without any neighbors within radius *eps* is called *noise*. The relationship state of two points in DBSCAN can be *direct density reachable*, *density reachable* or *density connected*. A point is called *direct density reachable* for C_i point if and only if it lies within the circle centered *core point* C_i. If a *core point* is connected unidirectionally to any other *core point* through a chain of *core points*, there is a *density reachable* state between them. In case that there are two points, which are *density reachable* from the same point, they are *density connected* states. Pseudocode describing DBSCAN clustering algorithm [5], [36] is shown in Algorithm 1.

Unlike the evaluation metrics for supervised learning algorithms, the evaluation of clustering performance can be applied to datasets with known or unknown ground truth labels. If the ground truth class assignment of dataset is known, we use entropy-based measure, **V-measure** proposed in [19] is used to evaluate clustering performance for our sample dataset. The sample dataset will be described in detail in Section 5. Based on the conditional entropy analysis of two terms of *homogeneity* and *completeness*, V-measure is a harmonic mean function of those terms and can be calculated as follows:

$$v(\beta, h, c) = \frac{(1+\beta) \times h \times c}{(\beta \times h) + c} \quad (4)$$

where $h = \frac{1 - H(C|K)}{H(C)}$ and $c = \frac{1 - H(K|C)}{H(K)}$ are *homogeneity* and *completeness*, respectively. The conditional entropy $H(K|C)$ and entropy $H(K)$ are symmetric. In formula (4), β weight represents the contributions of homogeneity or completeness and the default value of β is equal to 1.

Unfortunately, in fact, we don't know anything about the ground truth classes for document clustering

task. Thus, we can evaluate clustering performance based on the partition obtained from clustering techniques and two types of proximities, which are similarity or dissimilarity between objects. Suggested in [20], **Silhouette** is a typical evaluation for this case. Besides providing a graphical overview of the partitional clustering (silhouette plot), Silhouette also allows evaluating clustering validity based on the average silhouette width. From those analyses, we can obtain a suitable number of clusters for the K-means algorithm. The Silhouette Coefficient $s(i)$ for object i has the form:

$$s(i) = \frac{b(i)-a(i)}{\max\{a(i),b(i)\}} \quad (5)$$

Algorithm 1: Pseudocode of original DBSCAN algorithm for our data

```
Data: E[CLS]
Input : ε, MinPts
Input : metric                          // calculating distance between data points
Input : cluster
foreach p in E[CLS] do                  // linear scan for all data points in E[CLS]
    if cluster(p) != unassigned then continue; // p is unassigned to any cluster or
     noise
    Neighbors N ← find_neighbors(E[CLS], metric, p, ε)
    if |N| < minPts then
        cluster(p) ← noise
        continue
    end
    c ← create a new cluster
    cluster(p) ← c
    S ← N \ {p}
    foreach q in S do
        if cluster(q) == noise then cluster(q) ← c;
        if cluster(q) != unassigned then continue;
        Neighbors N ← find_neighbors(E[CLS], metric, q, ε)
        cluster(q) ← c
        if |N| < minPts then continue;
        S ← N ∪ S
    end
end
```

Here, $a(i)$ is the average dissimilarity of object i to the remaining objects in the same cluster and $b(i)$ is the average dissimilarity of i to all objects of the next nearest cluster. The value of Silhouette Coefficient is in the range $[-1,+1]$, where near -1 indicates the object for incorrect clustering and *vice versa* for +1. The value around 0 represents overlapping clusters.

5. EXPERIMENT

We apply K-means and DBSCAN algorithms with PhoBERT$_{base}$ to cluster our Facebook Vietnamese conversations dataset (FVnC) and Sample dataset. In order to implement clustering task, we use PhoBERT$_{base}$ with the Transformers package of Hugging Face and Scikit-learn library [33], [37]. Furthermore, we search the optimal parameters for the clustering algorithms in this article. The clustering results will be used to build Intents for chatbot later.

5.1 Clustering Task Dataset

We evaluate our approach on Facebook Vietnamese conversations dataset. There are plenty of free tools or extensions to download conversations from a personal page, because it is quite simple. However, collecting conversations from public page is more difficult, so most tools or extensions to carry out this task are paid. In order to acquire this dataset, we created a tool named *NTUCrawler*[2] for scraping conversations from a Facebook messenger page of our University. This tool is written in Python language and based on Facebook's Graph API platform to get messages. It has two versions, one is

[2] Tool is available at https://archive.org/download/NTUCrawler

linux executable (run on Ubuntu distribution) and the other is Windows executable. The UI of the Windows version is shown in Figure 3. *NTUCrawler* requires users to provide four parameters, start time and finish time to get data, PageID and Token of page. Downloaded dataset contains 8000 conversations with more than 150 thousand raw text sentences of clients and admins of a Facebook page in the six-month period of the year 2020. The contents of the conversations are FAQ (frequently asked questions), which are related to information already published on the university website; for example, tuition fees, insurance, dormitory, English language test, course registration, …etc.

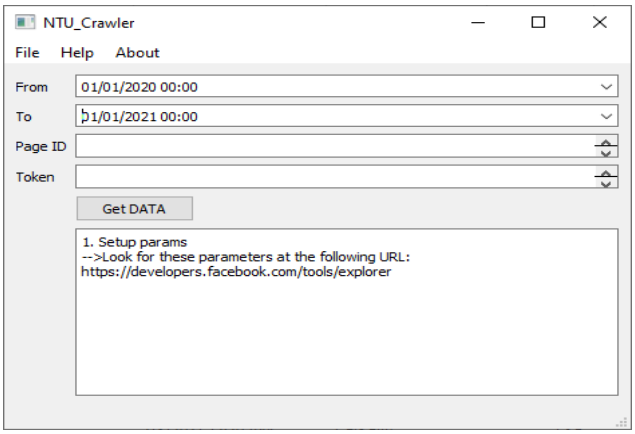

Figure 3. User interface of *NTUCrawler* on Windows.

5.2 Data Preprocessing

The raw downloaded dataset above must be preprocessed. Some punctuations "!", "", "?", "", ":" are removed from the dataset. Further, we eliminate stop words that have no value or no meaning to the NLP model, such as "dạ", "vâng", "chào ad", "vâng ạ", "alo", "ừ", "vậy", "ok", "nhé". We also filter out duplicate sentences or those that contain less than 4 words. After data preprocessing, 44846 text sentences are obtained and it is our desired **FVnC** dataset. From FVnC dataset, we randomly selected 95 sample text sentences (0.2% of FVnC dataset size) to form the **Sample** dataset. Besides that, we also use another sub-dataset which is called the **wiki** dataset. This dataset contains 396 text sentences of articles on 5 topics collected from Wikipedia. The Wikipedia library was applied to access and parse data from Wikipedia. The reason we use those Sample dataset and wiki sub-dataset is that we can evaluate the effectiveness of applying PhoBERT$_{base}$ for downstream task (clustering). Label assignment to sample text sentences and analyzing the number of clusters will cost less in our task. In addition, clustering on the small Sample dataset not only helps easily evaluate clustering performance, but also saves time compared to the original FVnC dataset. These sample text sentences were completed using manual labeling by us and divided into 3 classes, which describe questions between users (students) and admin (university) about information related to insurance, dormitory and English language test. Specifically, in order to specify the lables of classes, we rely on the experience of the specialists of the training department, who are responsible for answering students' questions directly or *via* social platforms. In their opinion, first- year students of our university are often concerned with insurance, dormitory and English language test. During data collection for each label, we carefully selected the sentences in the dataset that matched the recommendations of the specialists. Those 3 labels are one of the intents used to train the chatbot. Table 3 shows the details of the two datasets.

Table 3. Brief description of the datasets used for PhoBERT$_{base}$.

Dataset	Label	Description	Tasks	$E^i_{[CLS]}$ size
FVnC	unknown	clean downloaded dataset	Silhouette evaluation, clustering	44846 × 768
Sample	class 1	feature relating to insurance	clustering, clustering performance evaluation	31 × 768
	class 2	feature relating to English language test		38 × 768
	class 3	feature relating to dormitory		26 × 768

According to [15], input text must be already word-segmented before going through the BPE algorithm.

We use "pyvi" toolkit of Tran [31] to perform word segmentation in our datasets. After passing the fastBPE step, we have the index of tokens and attention masks for the text data. Taking them through PhoBERT's architecture leads to output embeddings of $E_{[CLS]}$. From this step, we use these embeddings as feature vectors to cluster text data.

5.3 Dealing with Varying Length

Because the length of the sentences in FVnC dataset is different, we have to use padding to make sure that the input texts have the same length. In particular, the maximum sequence length of PhoBERT is 256.

We truncate sentences with padding length less than 256 tokens. To choose the optimal padding length for all sentences, we can analyze the distribution of sentence lengths in Figure 4. Based on the above distribution, most sentences have lengths of less than 33 words. Thus, we decide that the padding length is equal to 33 in all datasets.

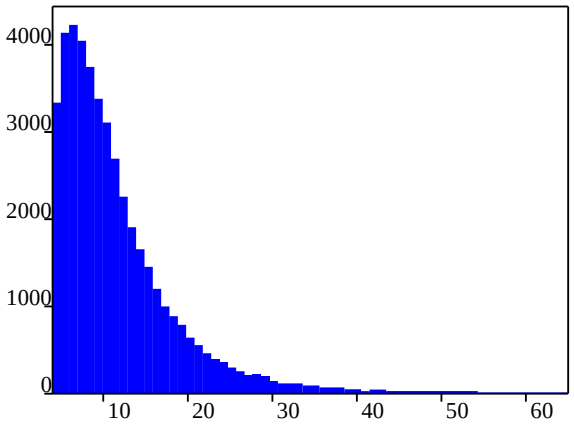

Figure 4. The distribution of the sentence lengths in FVnC dataset.

5.4 Parameter Optimization

When using the DBSCAN technique, we need to take care of two parameters; *MinPts* and ε. Choosing these two parameters is not easy. Their influence is very large on the clustering results. There is no way to accurately determine the parameter *MinPts*. However, there are several ways of choosing *MinPts* which have been proposed in [5]-[6]. Besides, the value of *MinPts* must also depend on domain knowledge and the data distribution observation. So, we derive that *MinPts* should be greater than the number of dimensionality of feature vector $\mathbf{E}_{[CLS]}$ (44846×768). As we can see, the number of dimensions of $\mathbf{E}_{[CLS]}$ is too large (*768*), which affects computation time and cost for large datasets. Therefore, we use the dimensionality reduction method to reduce $\mathbf{E}_{[CLS]}$ to lower-dimensional while retaining most of the original information. Principal Component Analysis (**PCA**) and t-distributed Stochastic Neighbour Embedding (**t-SNE**) are common techniques for data dimensionality reduction. **PCA** relies on eigenvalues and eigenvectors of $\mathbf{E}_{[CLS]}$ to reduce the original data to a specific number of dimensions (commonly known as *principal components*), but it still ensures a threshold of allowable variance. If we use **PCA,** then *MinPts* can be selected as follows:

$$MinPts \geq principal\ components + 1$$

Parameter ε can be found from a K-Distance graph, which is based on the average distance between objects and their MinPts nearest neighbors [7]. The K-Distance graph with MinPts=3 for FVnC and Sample dataset is shown in Figure 5.

The blue solid curve and red dashed curve correspond to the average distance of objects to MinPts nearest neighbors which are sorted in ascending order for FVnC and Sample dataset, respectively. Usually, a point at the position with the largest slope change in K-Distance graph or what we popularly call the "knee/elbow" of the graph is the optimal value of parameter ε [38]. Especially, the greatest slope change zones are highlighted in Figure 5(a) and Figure 5(b) for specific datasets. In order to take exactly the point mentioned above or the "knee point" of the graph, the kneedle algorithm is considered in our work [39]. The knee point obtained from the kneedle algorithm is determined by the intersection of the

specific data curve with the vertical straight line in Figure 6. The optimum values for parameter ε are 0.57 and 0.12 in the case of Figure 6 (a) and Figure 6 (b), respectively. However, optimizing parameter ε by choosing a fixed value of knee point in some cases does not lead to good clustering efficiency.

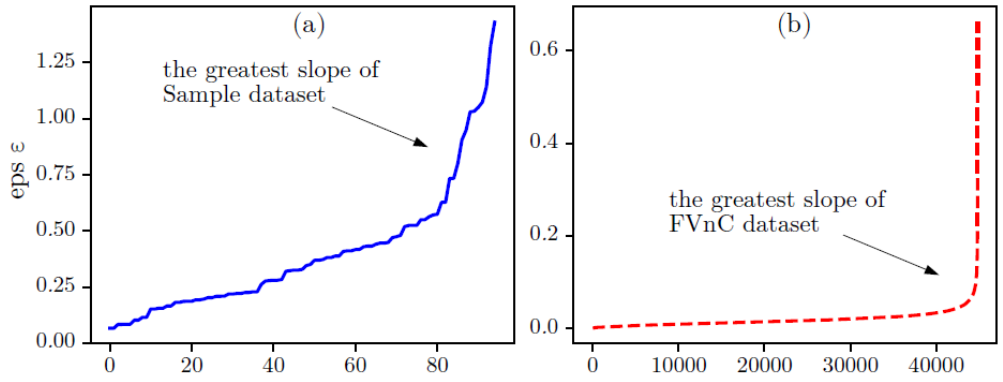

Figure 5. Data points are sorted ascending by the average distance to *MinPts* nearest neighbors. (a) Calculated on Sample dataset; (b) Calculated on FVnC dataset.

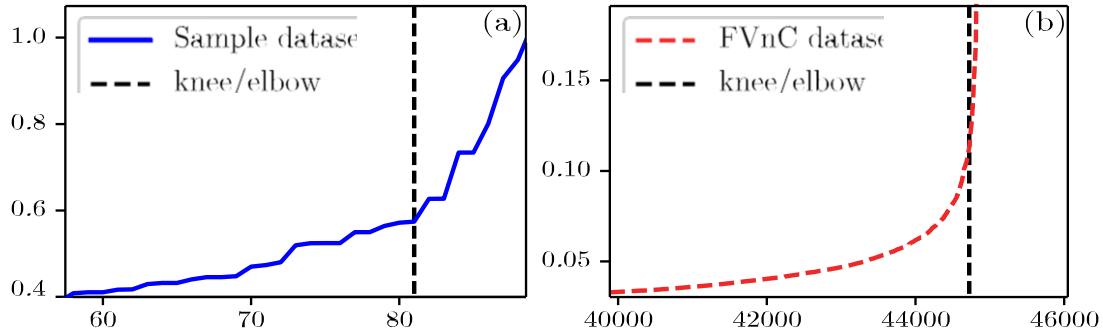

Figure 6. Determining knee points at the greatest slope change zones using the kneedle algorithm for Figure 5. (a) Solid blue curves – the greatest slope change zone of Sample dataset; (b) Dashed red curve – the greatest slope change zone of FVnC dataset.

Specifically in the test of the good separability between clusters in the sub-section below, Silhouette score is quite low. This could lead to objects being assigned to wrong clusters. With the expectation of improved clustering performance, in this work, we propose another technique based on the combination of K-Distance graph and clustering performance evaluations to find the right optimal value ε. Besides observing the greatest slope change zone of the line from a pair of points on the K-Distance graph, the maximum values of the Silhouette coefficient and V-measure score are also considered. The pseudocode for our technique is given in Algorithm 2. The sequence of MinPts is taken from principal components + 1 to 2 × principal components + 1 and incremented by a step of the minimum distance between data points. Gridsearch technique is applied in algorithm 2 for the sequence of MinPts and the greatest slope change zone in K-Distance graph. At the position where the Silhouette Coefficient is maximum, we obtain the optimal pair of values (ε, MinPts) for unlabeled data. For the labeled Sample dataset, the search of the optimal values (ε, MinPts) is based on the greatest mean of V-measure and Silhouette evaluation.

5.5 Result

In this part, our first task is to cluster text documents and evaluate clustering performance on the Sample dataset. As a consequence, analyzing the Sample dataset will be generalized to the general dataset FVnC, such as choosing the number of clusters using silhouette analysis for K-Means algorithm. As mentioned above, the Sample dataset has three labeled clusters (see Table 3). We use Silhouette evaluation to confirm that the Sample dataset has exactly three clusters and the way to choose the rightnumber of clusters when using it. Besides considering average silhouette scores, the silhouette plot is also an important factor in determining the number of clusters. Figure 7 represents the graphical overview

Algorithm 2: Pseudocode of the proposed technique to find appropriate ε and *MinPts*

Data : $\mathbf{E}_{[CLS]}$, label ; // label: points labels (unassigned or assigned)
Input : K-Distance
Input : *n_components*, step ; // step: minimum distance between points
Output: Index // the appropriate ε and $MinPts$ values for DBSCAN **Initial** : $\text{Max}_{Silhouette} \leftarrow -1$; $\text{Max}_{V\,measure} \leftarrow 0$; $\text{Max} \leftarrow -0.5$

```
slope ← CalculatingSlope(K-Distance)// the greatest slope change zone
nearest_neighbors ← arange(n_components+1, 2×n_components+1, step)
foreach ε in slope do
    foreach MinPts in nearest_neighbors do
        p ← PCA(E_[CLS], n_components)
        ClusterAssignment ← DBSCAN(p, MinPts, ε)
        SilCoeff ← SilhouetteScore(p, ClusterAssignment)
        if label = unassigned then
            if SilCoeff > Max_Silhouette then
                Max_Silhouette ← SilCoeff
                Index ← (ε, MinPts)
        else
            VScore ← VMeasureScore(label, ClusterAssignment)
            if (SilCoeff+VScore)/2 > Max then
                Max ← (SilCoeff + VScore)/2
                Index ← (ε, MinPts)
        end
    end
end
```

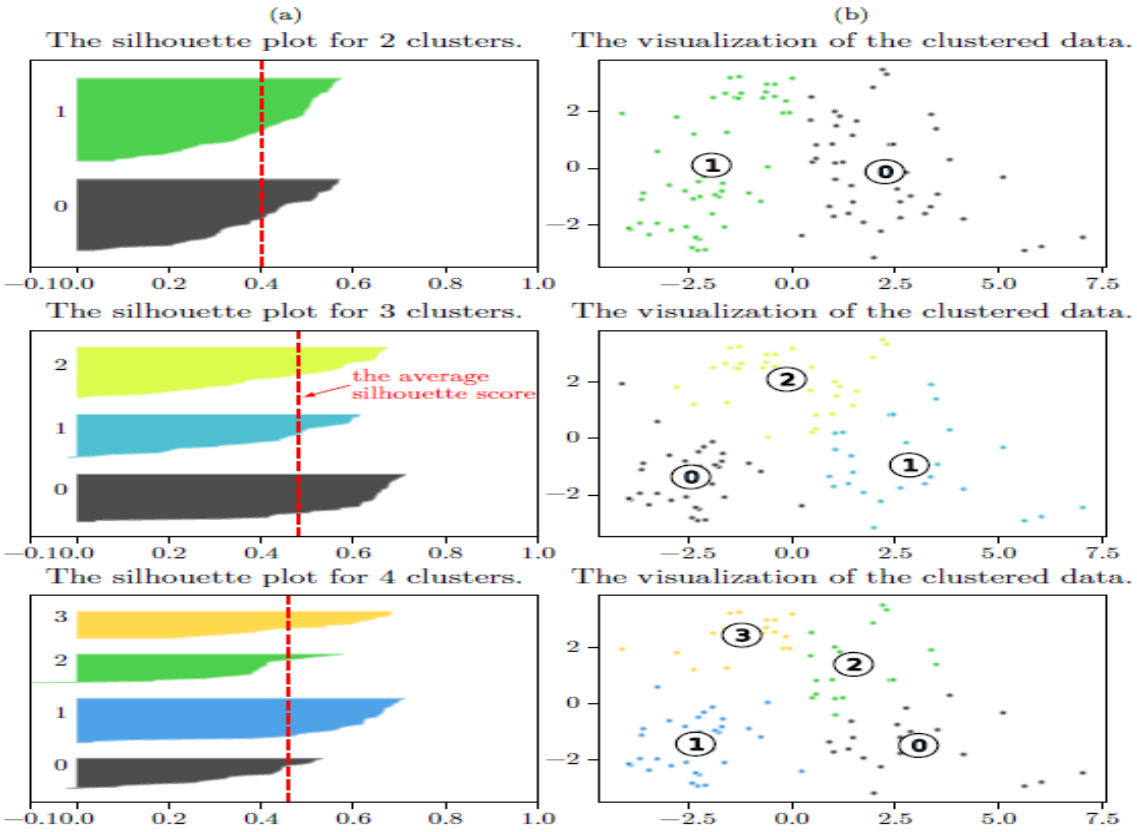

Figure 7. The graphical overview of the partitional clustering for 2, 3, 4 clusters using K-Means algorithm on the Sample dataset. (a) The silhouette plot; (b) The visualization of the clustered data for corresponding silhouette plots.

of the partitional clustering using K-Means algorithm on Sample dataset. X-axis of Figure 7 (a) corresponds to the silhouette coefficient values and the vertical dashed line is the average of the silhouette coefficients of data points. Clusters corresponding to silhouette plots in Figure 7(b) are visualized in two-

dimensional space using PCA. The maximum value of average silhouette scores is close to 0.481 for 3 clusters. On the other hand, the thickness of the silhouette plot for clusters is similar. The analysis outlined above fits the facts in Table 3 for the Sample dataset.

Moreover, in order to evaluate the efficiency of the PhoBERT$_{base}$ model in feature extraction for clustering tasks, we compared it with other models, such as BERT$_{base}$ Uncased, BERT$_{base}$ Multilingual Uncased [13], DistilBERT$_{base}$ Multilingual Cased [14], GPT-2 [12] based on V-measure score, among which GPT-2 and BERT$_{base}$ Uncased models do not support Vietnamese. We also use traditional approaches, like FastText, GloVe to compare with transformer models. Both models are applicable to Vietnamese language. However, both of these models are commonly used for learning word representations. Thus, to obtain sentence embedding, our method is averaging the word embeddings of all the words in the sentence. Especially, in order to make the comparison better, we also retrained the GloVe model on a new Vietnamese 20GB dataset based on the improvement proposed in [40]. Pre-trained word vectors **vncorpus.3B.100d** of new GloVe model correspond to the corpus 3B tokens, 1.3M vocab and 100d vectors and 1.17 GB download. Comparison results on the Sample dataset and wiki dataset using K-Means algorithm with 3 clusters and 5 clusters respectively are presented in Table 4. Through the obtained results, the models that support Vietnamese achieve significantly better V-measure scores than the rest of the models, especially the PhoBERT$_{base}$ model, which obtained the highest V-measure score on both the Sample dataset and wiki dataset (0.76 and 0.62, respectively). The comparisons obtained are in good agreement with the pointed theoretical and experimental works. Therefore, we use PhoBERT$_{base}$ model to extract features for the FVnC dataset, which leads to the output embeddings serving the clustering task.

Table 4. Evaluating feature extraction efficiency of models through V-measure scores for clustering task on the Sample dataset and wiki dataset.

Approach	Dataset	
	Sample	wiki
FastText (cc.vi.300)	0.64	0.27
GloVe (glove.6B.100d)	0.49	0.26
GloVe (vncorpus.3B.100d)	0.61	0.27
BERT$_{base}$ Uncased	0.11	0.19
BERT$_{base}$ Multilingual Uncased	0.36	0.43
DistilBERT$_{base}$ Multilingual Cased	0.37	0.12
GPT-2	0.04	0.04
PhoBERT$_{base}$	**0.76**	**0.62**

For DBSCAN clustering method on the Sample dataset, we need to find the optimal parameters ε, *MinPts* using Algorithm 2 and kneedle algorithm. Based on K-Distance graph of Sample dataset in Figure 5(a) and knee visualization in Figure 6(a), the greatest slope zone is selected in the range of [0.5,0.85] and the knee point value is 0.57. Due to "*principal components*" equal to 2 in PCA dimensionality reduction, the value of *MinPts* will be selected from 3 to 5. Some experimental results in finding the optimal parameters are shown in Table 5. As shown in the table, ε, *MinPts* and the average of V-measure and Silhouette score equal to 0.84, 4 and 0.42, respectively, are the best choice to cluster data points. Results in Table 5 also show that our approach – Algorithm 2- gives better clustering performance than kneedle algorithm on the Sample dataset. Using DBSCAN algorithm with the obtained parameters, 5 clusters are formed in Figure 8, in which cluster "-1" contains noisy objects in Figure 8 (a), noisy objects are removed in Figure 8 (b) and Silhouette plot of denoised Sample dataset is shown in Figure 8 (c). After removing noise, our clustering result has 4 clusters, while the Sample dataset has only 3 clusters. Based on actual observation in Figure 8 (b) and Silhouette plot in Figure 8 (c), cluster 0 and 1 must be merged into one cluster.

Remark 1. *DBSCAN is a density-based spatial clustering algorithm. The biggest disadvantage of DBSCAN is working in cases of varying-density clusters. From the Silhouette plot in Figure 8(c), there are two separate clusters of very different density comparing to the other two clusters. Two low-density clusters (cluster 0 and 1) are a matter of concern to us for predicting the number of clusters. Obviously, we hope that our clusters will be more equally distributed to choose the number of clusters more precisely.*

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

In order to solve this problem, we propose some solutions as follows:

Table 5. Experiment to find the optimal values of parameters ε, *MinPts* for DBSCAN algorithm on the Sample dataset.

Approach	eps (ε)	*MinPts*	V-measure score	Silhouette score	Average	N-Clusters
	0.84	**4**	**0.56**	**0.28**	**0.42**	**5**
	0.78	4	0.52	0.27	0.4	5
Algorithm 2	0.72	4	0.5	0.22	0.36	6
	0.76	3	0.47	0.22	0.34	4
	0.54	5	0.37	0.0	0.18	8
	0.57	3	0.42	0.15	0.29	9
Knee point	0.57	4	0.41	0.1	0.26	8
	0.57	5	0.41	0.05	0.23	7

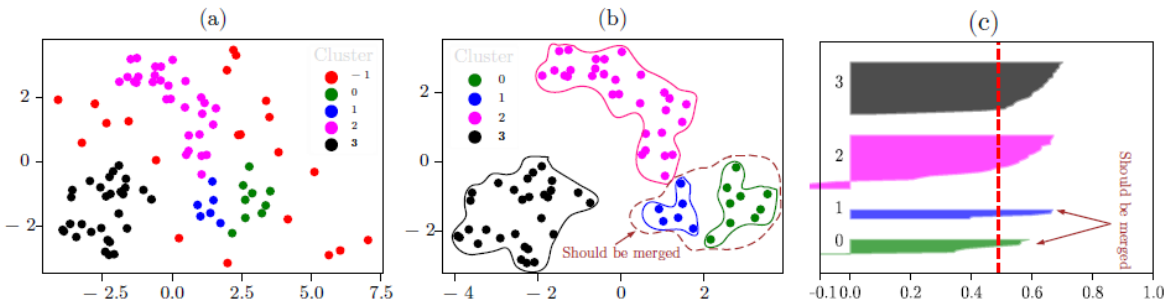

Figure 8. DBSCAN clustering on the sample dataset.

(i) Based on the thickness of the silhouette plot, the silhouette plots for clusters 0 and 1 are much smaller than for clusters 2 and 3. In particular, clusters 0 and 1 are close neighbors. Therefore, we have reasons to merge these small two clusters into a larger cluster. As a result, 3 clusters are the correct number of clusters when using DBSCAN clustering on the Sample dataset. After all, we should combine the selection of the optimal parameters and the width of Silhouette plots for clusters when using DBSCAN algorithm to obtain the most accurate number of clusters.

(ii) On the other hand, we can use a criterion for determining the minimum number of core points in a cluster. If at least two clusters are close to each other and have a smaller minimum number of core points than the specified criteria, we can merge them into a larger cluster.

Finally, we perform the main task, which is text clustering for the FVnC dataset. Note that the ground truth classes of the FVnC dataset are unknown. In order to estimate the number of specific clusters for K-Means clustering method, we take full advantage of DBSCAN algorithm. In the same way, we get the number of clusters from Algorithm 2. Specifically, we choose the number of clusters through Silhouette scores with large values corresponding to each pair of parameters ε and MinPts. For FVnC dataset, the optimal values of ε can be found in Figure 5(b). After applying DBSCAN algorithm, the number of clusters actually obtained must be subtracted by 1 for noisy objects (points labeled -1). Before using K-Means algorithm to perform clustering, these noisy objects are removed from FVnC dataset, which leads to the form of the corresponding denoised FVnC datasets. Lastly, we obtain clustering results of FVnC dataset without noise from the K-Means method based on the Silhouette evaluation. Silhouette scores test on original FVnC and denoised FVnC datasets for several different cluster numbers can be examined, as shown in Table 6. In case of using ε = 0.14 and MinPts = 25, the best obtained Silhouette scores for the original FVnC and denoised FVnC dataset are 0.3359 and 0.3460, respectively.

The Silhouette plots and clustering visualization of the best Silhouette scores can be seen in Figure 9. By comparing the Silhouette plots, we believe that the clustering result on the denoised FVnC dataset (see Figure 9 (b)) is better than on the original FVnC dataset (see Figure 9 (a)), because there are some noise points outside the clusters and the size of the silhouette plots of clusters 3 and 4 represents a wide fluctuation in the visualization of Figure 9(a). With the number of clusters received from the DBSCAN algorithm, the K-Means algorithm gives very good clustering results for denoised FVnC

datasets. After all, the clusters obtained from the clustering process are considered as big intents to help us build data for chatbot. For this reason, we can save time and effort and build chatbot faster.

Table 6. Silhouette scores using K-Means algorithm on original FVnC and denoised FVnC datasets for several different cluster numbers found from DBSCAN algorithm.

DBSCAN			K-Means Silhouette Scores	
eps ε	*MinPts*	N-Clusters	Original FVnC	Denoised FVnC
0.14	22	10	**0.3359**	0.3436
0.14	25	12	**0.3359**	**0.3460**
0.13	12	15	0.3310	0.3368
0.14	19	16	0.3294	0.3326
0.13	16	20	0.3267	0.3314
0.12	30	23	0.3252	0.3418
0.11	26	29	0.3225	0.3343
0.1	14	53	0.3196	0.3257
0.09	12	75	0.3196	0.3275
0.1	3	81	0.3217	0.3205
0.09	4	102	0.3220	0.3229
0.09	3	128	0.3207	0.3228
0.08	4	136	0.3212	0.3235

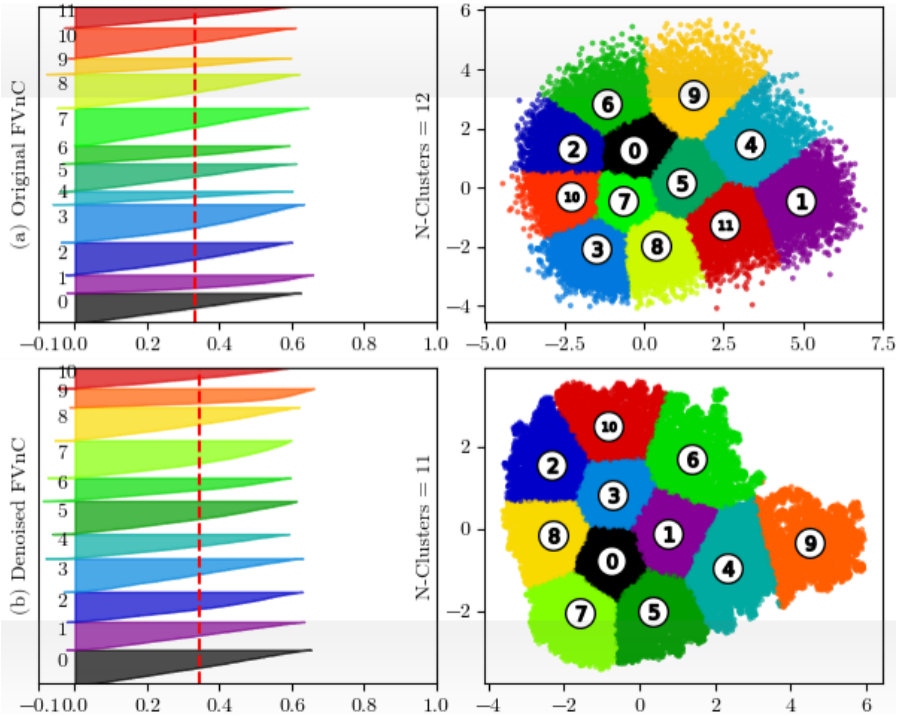

Figure 9. Silhouette plots and clustering visualization.

6. CONCLUSION

In this work, we research transformer architecture as well as pre-trained language models, such as BERT and PhoBERT. We also cover how to apply PhoBERT to our Facebook Vietnamese conversations dataset (FVnC). Furthermore, we have built a tool to crawl conversations from a Facebook Messenger Page. After extracting embedding vectors at the final hidden layer, we use the unsupervised learning algorithms K-means and DBSCAN to cluster text data. V-measure score and Silhouette score are used to evaluate the performance of clustering algorithms. A GridSearch algorithm that combines these two clustering evaluations is also proposed to find optimal parameters for the DBSCAN algorithm. The algorithm proposed by us obtained better clustering performance than kneedle algorithm through experimentations

based on V-measure scores and Silhouette score on the Sample and FVnC datasets. In addition, we compare the efficiency of the PhoBERT$_{base}$ model in feature extraction for clustering tasks with those of other models. PhoBERT$_{base}$ model achieves the best V-measure score on the Sample dataset and wiki dataset. We apply the K-Means clustering method with the number of clusters received from the DBSCAN algorithm to cluster the FVnC dataset. Topics obtained from clustering are similar to intents in building chatbot. From a pre-analysis data screening perspective, clustering results are valuable for building stories in our chatbot. Thanks to the implementation of this clustering, we save a lot of time and effort to build data and storylines for training chatbot.

ACKNOWLEDGMENTS

This work was partially supported by Nha Trang University (project TR2020-13-42). The authors thank Hien Thao Le for proofreading our manuscript and fruitful discussions. We also address special thanks to the reviewers for their helpful comments and suggestions.

REFERENCES

[1] A. Jung, "A Gentle Introduction to Supervised Machine Learning," Computing Research Repository (CoRR), vol. abs/1805.05052, pp. 6–7, 2018.

[2] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman and A. Wu, "An Efficient K-means Clustering Algorithm: Analysis and Implementation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881–892, 2002.

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[4] J. B. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297, University of California Press, 1967.

[5] M. Ester, H.-P. Kriegel, J. Sander and X. Xu, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proc. of 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96), pp. 226–231, 1996.

[6] J. Sander, M. Ester, H. Kriegel and X. Xu, "Density-based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications," Data Mining and Knowledge Discovery, vol. 2, pp. 169– 194, 1998.

[7] M. Gaonkar and K. Sawant, "AutoEpsDBSCAN: DBSCAN with Eps Automatic for Large Dataset," IRD India, vol. 2, no. 2, pp. 11–16, 2013.

[8] G. Salton and M. J. McGill, "Introduction to Modern Information Retrieval," McGraw-Hill Computer Science Series, 1986.

[9] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Proc. of the 1st Int. Conf. on Learning Representations (ICLR 2013), Scottsdale, USA, 2013.

[10] J. Pennington, R. Socher and C. Manning, "GloVe: Global Vectors for Word Representation," Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, Doha, Qatar: Association for Computational Linguistics, Oct. 2014.

[11] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, "Enriching Word Vectors with Subword Information," arXiv preprint arXiv:1607.04606, 2016.

[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei and I. Sutskever, "Language Models are Unsupervised Multitask Learners," Proceedings{Radford2019LanguageMA}, [Online], Available: https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf, 2019.

[13] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pretraining of Deep Bidirectional Transformers for Language Understanding," Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technol., vol. 1, pp. 4171–4186, 2019.

[14] V. Sanh, L. Debut, J. Chaumond and T. Wolf, "DistilBERT, A Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter," ArXiv, [Online], Available: https://arxiv.org/abs/1910.01108, 2019.

[15] D. Q. Nguyen and A. T. Nguyen, "PhoBERT: Pre-trained Language Models for Vietnamese," Proc. of Findings of the Association for Computational Linguistics (EMNLP 2020), pp. 1037–1042, arXiv: 2003.00744, 2020.

[16] O. Gencoglu, "Deep Representation Learning for Clustering of Health Tweets," Computing Research Repository (CoRR), vol. abs/1901.00439, [Online], Available: https://arxiv.org/pdf/1901.00439, 2019.

[17] L. Pugachev and M. Burtsev, "Short Text Clustering with Transformers," arXiv preprint arXiv:2102.00541, [Online], available: https://arxiv.org/pdf/2102.00541, 2021.

[18] S. Sia, A. Dalmia and S. J. Mielke, "Tired of Topic Models? Clusters of Pretrained Word Embeddings Make for Fast and Good Topics Too!" arXiv preprint arXiv: 2004.14914, [Online], Available: https://arxiv.org/pdf/2004.14914, 2020.

[19] A. Rosenberg and J. Hirschberg, "V-measure: A Conditional Entropy-based External Cluster Evaluation Measure," Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 410–420, Prague, Czech, Jun. 2007.

[20] P. J. Rousseeuw, "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis," Journal of Computational and Applied Mathematics, vol. 20, pp. 53–65, 1987.

[21] A. Vaswani et al. , "Attention Is All You Need," Proc. of Advances in Neural Information Processing Systems, vol. 30, [Online], Available: https://arxiv.org/pdf/1706.03762, Curran Associates, Inc., 2017.

[22] C. Sun, X. Qiu, Y. Xu and X. Huang, "How to Fine-tune BERT for Text Classification?" Proc. of the China National Conference on Chinese Computational Linguistics (CCL 2019), vol. 11856, pp. 194–206, Cham: Springer Int. Publishing, 2019.

[23] J. Chung, C. Gulcehre, K. Cho and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," Proc. of the NIPS 2014 Workshop on Deep Learning, NYU, 2014.

[24] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," Proc. of the 31st International Conference on Machine Learning, Ser. Proceedings of Machine Learning Research, vol. 32, no. 2, pp. 1188–1196, PMLR, Bejing, China, 22–24 Jun. 2014.

[25] A. Joulin, E. Grave, P. Bojanowski and T. Mikolov, "Bag of Tricks for Efficient Text Classification," Computing Research Repository (CoRR), vol. abs/1607.01759, 2016.

[26] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, "Deep Contextualized Word Representations," arXiv Preprint arXiv: 1802.05365, [Online], Available: https://arxiv.org/pdf/1802.05365, 2018.

[27] A. Radford and K. Narasimhan, "Improving Language Understanding by Generative Pre-training," [Online], Available: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.

[28] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and Stoyanov, "Roberta: A robustly optimized BERT Pretraining Approach," arXiv: 1907.11692, 2019.

[29] R. Sennrich, B. Haddow and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," Proc. of the 54th Annual Meeting of the Association for Computational Linguistics, vol. 1: Long Papers), pp. 1715–1725, DOI: 10.18653/v1/P16-1162, Aug. 2016.

[30] T. Vu, D. Q. Nguyen, M. Dras and M. Johnson, "VnCoreNLP: A Vietnamese Natural Language Processing Toolkit," Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pp. 56–60, New Orleans, Louisiana, 2018.

[31] V.-T. Tran, "Python Vietnamese Toolkit," pyvi 0.1.1, pypi, [Online], Available: https://pypi.org/project/pyvi/, 2020.

[32] V. Anh, B. N. Anh and D. V. Dung, "Open-source Vietnamese Natural Language Process Toolkit," VnCoreNLP, Github, [Online], Available: https://github.com/undertheseanlp/word_tokenize, 2018.

[33] T. Wolf et al., " HuggingFace's Transformers: State-of-the-art Natural Language Processing," Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45, Oct. 2020.

[34] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier and M. Auli, "FAIRSEQ: A Fast, Extensible Toolkit for Sequence Modeling," Proceedings of NAACL-HLT 2019: Demonstrations, pp. 48-53, Minneapolis, Minnesota, 2019.

[35] T. Neeraj, "Feature-based Approach with BERT," Trishala's Blog, Github, [Online], Available: trishalaneeraj.github.io, 2020.

[36] E. Schubert, J. Sander, M. Ester, H. Kriegel and X. Xu, "DBSCAN Revisited: Why and How You Should (Still) Use DBSCAN," ACM Trans. Database Syst., vol. 42, no. 3, pp. 19:1–19:21, DOI: 10.1145/3068335, 2017.

[37] Scikit-learn, "Clustering," Scikit-learn 1.0.2 documentation, [Online], Available: https://scikit-learn.org/stable/modules/clustering.html#k-means, 2011.

[38] N. Rahmah and I. S. Sitanggang, "Determination of Optimal Epsilon (Eps) Value on DBSCAN Algorithm to Clustering Data on Peatland Hotspots in Sumatra," Proc. of the IOP Conference Series: Earth and Environmental Science, Workshop and International Seminar on Science of Complex Natural Systems, vol. 31, p. 012012, Bogor, Indonesia, Jan. 2016.

[39] V. Satopaa, J. Albrecht, D. Irwin and B. Raghavan, "Finding a" kneedle" in a Haystack: Detecting Knee Points in System Behavior," Proc. of the 31st IEEE International Conference on Distributed Computing

Systems Workshops, pp. 166–171, DOI: 10.1109/ICDCSW.2011.20, Minneapolis, MN, USA, 2011.
[40] T. H. Nguyen, "Analyze the Effects of Weighting Functions on Cost Function in the Glove Model," arXiv preprint arXiv: 2009.04732, [Online], Available: https://arxiv.org/pdf/2009.04732, 2020.

ملخص البحث:

يتمثّل التّحدي الأكبر في بناء برامج المحادثة في بيانات التّدريب. ويتعيّن أن تكون البيانات المطلوبة واقعيةً وضخمةً بما يكفي لتدريب برامج المحادثة. نقوم ببناء أداةٍ للحصول على بيانات تدريب حقيقية من بريد مراسلات فيسبوك في إحدى صفحات فيسبوك. وبعد خطوات المعالجة الأوّلية للنّصّ، فإنّ مجموعة البيانات التي تمّ الحصول عليها للتّوّ، تولّد مجموعة بيانات (FVnC) ومجموعة بيانات العيّنة. نستخدم إعادة التّدريب لـِ (BERT) من أجل (PhoBERT) باللغة الفيتنامية لاستخلاص سِمات بيانات النّصّ.

تم استخدام خوارزميات (K-Means) و (DBSCAN) للتّجميع للقيام بمهمات التّجميع بناءً على تضمينات المخرج من (PhoBERT). وجرى تطبيق درجة مقياس V ودرجة الظّل (Silhouette) لتقييم أداء خوارزميات التّجميع. كذلك تم عرض فعاليّة (PhoBERT) مقارنة مع أداء نماذج أخرى لاستخلاص السِّمات في مجموعة بيانات العينة ومجموعة بيانات الموسوعة (ويكي). من ناحية أخرى، تمّ اقتراح خوارزمية بحث في الشّبكة (GridSearch) تجمع بين التّقييميْن المتعلّقيْن بالتّجميع بُغية إيجاد المتغيرات المثالية. وبفضل تجميع هذا العدد من المحادثات، فإننا نوفّر الكثير من الوقت والجهد لبناء البيانات من أجل تدريب برامج المحادثة.