University of Science and Technology of Hanoi

Final report

ASPECT-BASED SENTIMENT ANALYSIS ON VIETNAMESE CUSTOMER FEEDBACK

Course: Natural Language Processing
Lecturer : Nghiem Thi Phuong

Group 2

22BI13269	Luu Linh Ly
22BI13268	Vu Hai Thien Long
22BI13320	Cao Nhat Nam
22BI13120	Nguyen Duc Duy
22BI13063	Pham Ngoc Minh Chau
BA12-174	Do Thi Huong Tra

Abstracts

This study uses **Aspect-Based Sentiment Analysis (ABSA)** to analyze restaurant reviews from **Foody.vn**, focusing on customer opinions about food, service, pricing, ambiance, and delivery. A dataset of 12,418 labeled reviews was created, and two methods were tested: a **two-step process** with the **PhoBERT model** for finding aspects and their sentiment, and a single-model approach for direct sentiment analysis. The two-step method performed better than other models, reaching **F1 scores** of 96.25 for aspect detection and 89.96 for sentiment classification. These results show that **ABSA** helps provide useful insights to improve restaurant services.

TABLE OF CONTENTS

I. Introduction

The rapid expansion of the Internet has led to an overwhelming amount of user-generated data, particularly in e-commerce, social media, and digital search platforms. This surge has fueled advancements in **Artificial Intelligence (AI)**, especially in **Natural Language Processing (NLP)**, where **Sentiment Analysis** plays a key role in evaluating customer satisfaction. However, traditional **Sentiment Analysis** often overlooks the complexity of opinions within a single review. To address this, **Aspect-based Sentiment Analysis (ABSA)** provides a more refined approach by examining sentiments tied to specific aspects of a product or service.

ABSA categorizes text into relevant aspects and determines **sentiment polarity** for each one, offering deeper insights than conventional **Sentiment Analysis**. In restaurant reviews, for example, it assesses **food quality**, **service**, **pricing**, and **ambiance** separately. The rise of online reviews and mobile devices has amplified the availability of real-time feedback, enabling businesses to better understand customer preferences and areas for improvement. By distinguishing sentiments at a granular level, **ABSA** helps restaurants identify key strengths and weaknesses in their offerings.

This research applies **ABSA** to restaurant reviews, focusing on customer sentiments regarding **food**, **service**, **pricing**, and **atmosphere**. The goal is to extract actionable insights that can enhance customer satisfaction, refine menu offerings, improve staff training, and strengthen marketing strategies. Additionally, **ABSA** aids in **competitive benchmarking**, allowing restaurants to assess their market position. By transforming vast review data into practical recommendations, this study not only benefits restaurant management but also contributes to **NLP** research in service industries.

II. Problems description

In this project, we focus on **Aspect Category Sentiment Analysis** , which is divided into two main subproblems: **Aspect Category Detection** and **Sentiment Polarity Classification**. Aspect Category Detection involves identifying the aspect **A** mentioned in a review, selected from a predefined set of categories relevant to customer feedback, such as **"SERVICE"** or **"AMBIANCE"**. Once an aspect is identified, Sentiment Polarity Classification assigns it a sentiment label—**Positive, Negative, or Neutral**—based on the review's context.

For example, consider:

Vietnamese review: *"Cafe ngon, nhân viên thân thiện nhiệt tình, quán mộc mạc sang trọng."*

The English translation is: *"Delicious coffee, friendly and enthusiastic staff, rustic yet elegant café."*

From this review, two aspect-based sentiment annotations can be extracted: **{aspect: "Food", sentiment: "Positive"}, {aspect: "SERVICE", sentiment: "Positive"}** and **{aspect: "AMBIANCE", sentiment: "Positive"}**.

This structured approach ensures that each review is analyzed in terms of relevant aspects and their corresponding sentiment, improving the accuracy and effectiveness of sentiment analysis.

III. Data Crawling

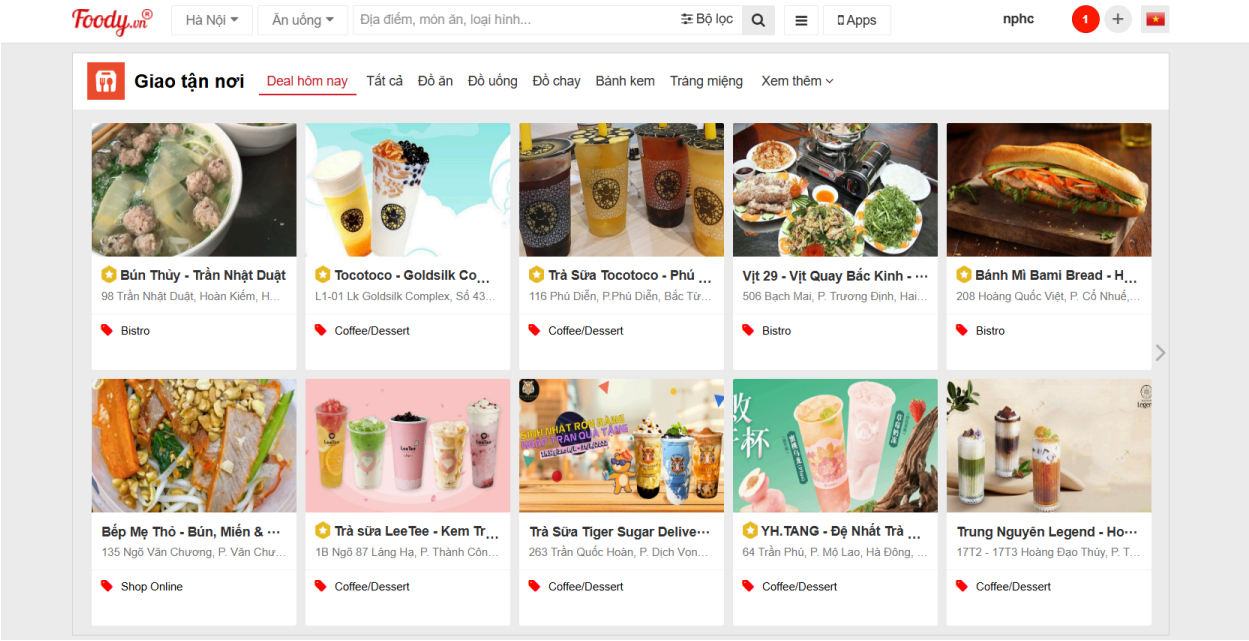

Figure 1: Foody.vn's front page

To collect data for this project, we implemented a **web scraping** approach using **Selenium** to extract **store information** and **customer reviews** from the **Foody.vn** website. The process consists of two main stages: collecting **store links** and extracting **customer comments**.

1. Collecting store links

In the first stage, we retrieve **store URLs** from various **provinces** listed on **Foody.vn**. The function ***f(prn)*** automates the navigation through the **"Load More" button** to gather as many **stores** as possible.

We configure **Selenium** with **Chrome options**, including **incognito mode** and a **small window size** to minimize resource usage. The script constructs the **URL** for each **province** and loads the page, then uses **CSS selectors** to identify and extract the **URLs of stores** listed on the page. The **"Load More" button** is clicked repeatedly to retrieve additional **stores** until a threshold of **3000**

stores is reached. Finally, the extracted **store URLs** are stored in a **list** with **duplicates removed** before proceeding to the next stage.

The **12 provinces** included in this process are: **Đà Nẵng, Cần Thơ, Hải Phòng, Đồng Nai, Bình Dương, Điện Biên, Khánh Hòa, Hà Nội, Lâm Đồng, Bình Định, Huế,** and **Hồ Chí Minh City**.

2. Extracting customer reviews

Once we have the **store links**, we proceed to extract **customer comments** using the **g(stores)** function.

For each **store URL**, the script appends **"/binh-luan"** to access the **review section**. If the **"Load More" button** is available, the script repeatedly clicks it to load all **reviews**. The script then collects **review texts** from elements with the **CSS selector .rd-des**. If an issue occurs, such as a **missing page** or **connection issue**, the script logs the **error** and proceeds. Finally, **redundant reviews** are removed to ensure **data integrity**.

3. Parallel processing

Given the **large dataset**, we **parallelized** the **scraping process** using Python's **multiprocessing** library. This enables multiple instances of the **scraping function** to run **simultaneously** for different **provinces**, significantly improving **efficiency.**

4. Data storage

After **extraction**, the collected **reviews** are stored in a structured **Pandas DataFrame** with columns:

- **id**: Unique **store identifier**
- **comments**: List of extracted comments

Each **province's dataset** is saved as a **CSV file** for further **analysis**.

IV. **Methodologies**

1. Data exploration

After crawling, we have 180,000 customer reviews from Foody.vn. These reviews are processed and prepared for evaluation and annotation, with the goal of training a Natural Language Processing (NLP) model to analyze and interpret customer feedback effectively.

To ensure relevance, we filter the raw dataset to include only reviews that discuss food quality and overall dining experience. This refinement allows for more precise aspect-based sentiment analysis, as the selected reviews focus on key factors influencing customer satisfaction. By narrowing the scope to food and experience, we enhance the model's ability to extract meaningful insights from the data.

```
comments
ăn rất ngon được phục vụ chu đáo với 2 cô chú người huế có rất nhiều m
quán này khá đông khách vào buổi tối nên phải đi sớm mới có chỗ được v
nếu như muốn ăn các món đặc sản của miền bắc việt nam thì đây là địa đ
cho mình số điện thoại để liên hệ với quán ơi hoặc liên hệ mình nhé hẳ
nha hang phuc vu cac mon an rat ngon nha
```

Figure 2: A sample review of a restaurant

Due to time limits, we used **Gemini-2.0-flash API** from Google for **initial review annotation** of only 12,418 reviews with **few-shot prompting** (giving examples in the prompt). Then, our team **manually checked** the results to create the training dataset for the sentiment analysis model.

The annotation process focuses on five key aspects: **FOOD, AMBIENCE (or space), SERVICE, DELIVERY, and PRICE**. Each aspect is assigned one of three sentiment labels: **POSITIVE, NEGATIVE, NEUTRAL**, based on the sentiment expressed in the review. If a review does not mention a specific aspect, no label is assigned to it. This approach ensures that the sentiment analysis remains precise and directly relevant to the aspects discussed.

Each review is annotated in the following format:

#1	#index

ăn rất ngon được phục vụ chu đáo với 2 cô chú người huế có rất nhiều món như bò mọc chân giò chả cua rất đặc biết các bạn nên thử thập cẩm là đầy đủ tất cả một bát chỉ có 30 nghìn đồng	review_content
{FOOD,positive}, {SERVICE, positive}, {PRICE, positive}	{aspect, sentiment} pair(s)

This structured annotation format helps maintain consistency and clarity in the dataset, improving the model's ability to recognize patterns and analyze customer opinions effectively.

We then do some statistics for the annotated data.

Aspect/Sentiment	negative	neutral	none	positive
AMBIENCE	8.92	1.36	44.81	44.90
DELIVERY	15.17	1.12	0.00	83.71
FOOD	12.96	9.56	5.88	71.60
PRICE	10.76	5.73	44.70	38.80
SERVICE	12.91	1.73	39.13	46.24

Table 1: Proportion of sentiment per aspects

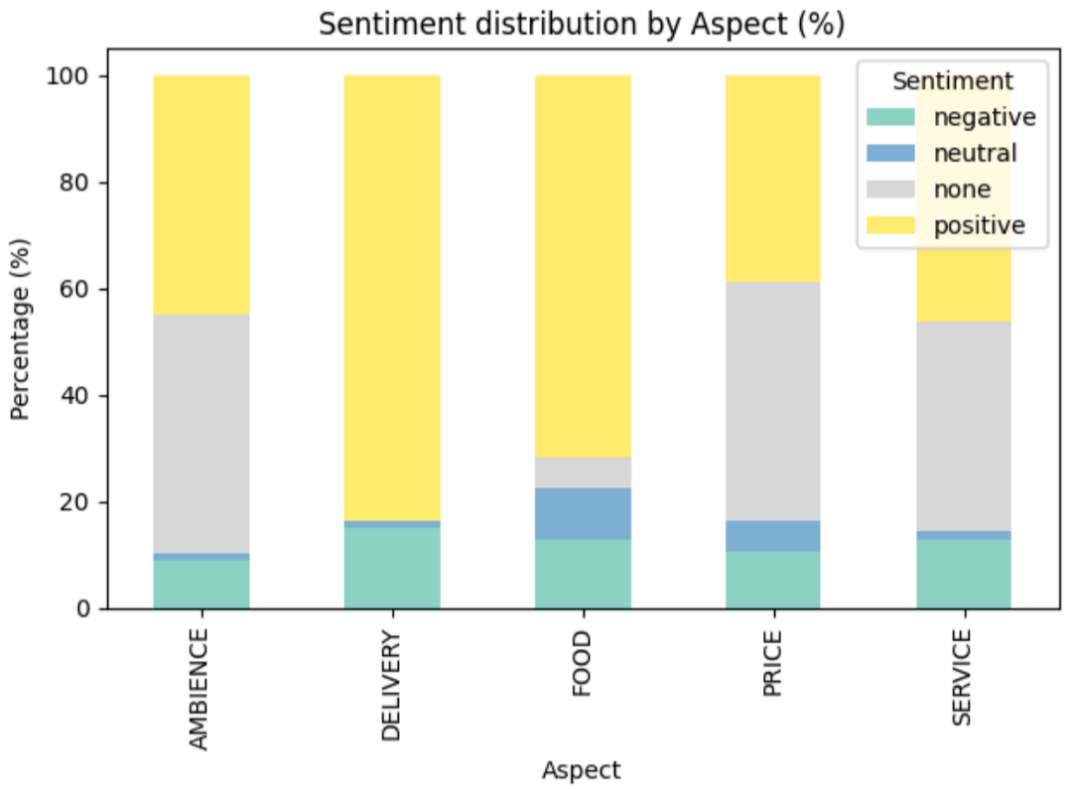

Figure 3: Plot of distribution in table.

The above distribution shows that the data is **seriously** imbalanced, for each approach we will implement different treatments.

2. Combined-model approach

This approach fine-tunes the PhoBERT Vietnamese transformer model using a two-stage pipeline. In Stage 1, we adapt PhoBERT for multi-label classification to extract aspects (such as AMBIENCE, PRICE, FOOD, SERVICE, and DELIVERY) from review texts. In Stage 2, we leverage a sentiment classification model—initialized from PhoBERT with encoder weights transferred from Stage 1—to predict a single sentiment (negative, neutral, or positive) for each extracted aspect.

2.1 Base Model: VinAI PhoBERT-Base

PhoBERT pre-training approach is based on RoBERTa which optimizes the BERT pre-training procedure for more robust performance. PhoBERT achieves high performance across a range of NLP tasks in Vietnamese. Its pre-trained models provide an excellent basement for fine-tuning on various tasks, making it a powerful tool for Vietnamese language understanding and applications.

PhoBERT Base v2 model has 135 million parameters, making it a powerful model for various Vietnamese language tasks. Its architecture consists of 12 transformer layers (each consisting of multi-head attention and feed-forward neural networks), 768 hidden units per layer, and 12 attention heads in each layer.

PhoBERT's pre-training process involves a **single** main objective: Masked Language Modeling (MLM). In this task, PhoBERT randomly masks some words in the input text and challenges the model to predict the masked words. This helps the model capture both local and global contextual information, allowing it to understand word relationships within a sentence. Unlike BERT, PhoBERT does not use Next Sentence Prediction (NSP). Instead, it follows RoBERTa's training approach, which removes NSP and relies on training with larger amounts of shuffled and continuous text segments.

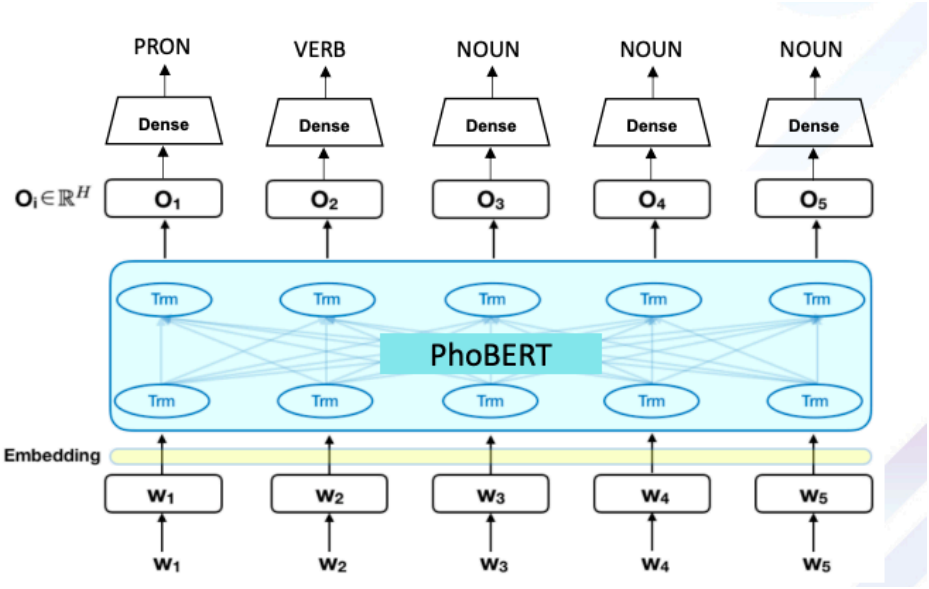

Figure 4: PhoBERT in Part-of-Speech (POS) tagging task

2.2 General pipeline for this approach

Our ABSA pipeline is divided into two distinct stages:

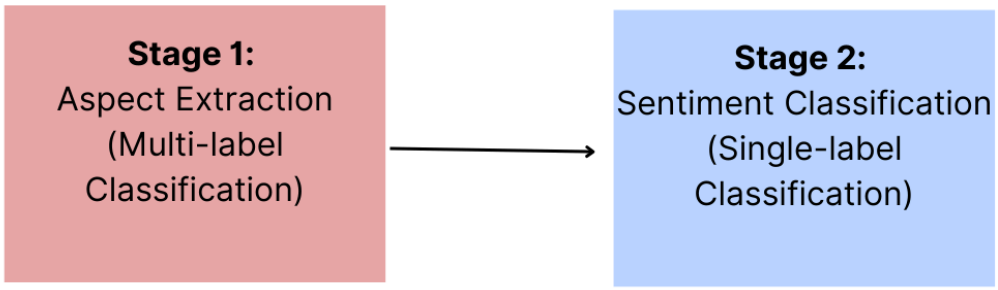

Figure 5: Combined-model pipeline

Each stage uses PhoBERT-Base but applies different architectural modifications, training strategies, and data preprocessing techniques.

Throughout this pipeline, the dataset is divided into training (80%) and validation sets (20%) for both stages of the ABSA pipeline: aspect extraction (Stage 1) and sentiment classification (Stage 2). The split ensures a balanced representation of aspects and sentiment labels across both sets, preventing data leakage, ensuring the model generalizes well, and providing a clear way to evaluate the full pipeline.

2.3 Stage 1: Aspect Extraction

Multiple aspects (e.g., "AMBIENCE", "PRICE", "FOOD", "SERVICE", "DELIVERY") needed to be extracted from review text. Since a review might mention several aspects, we frame this as a multi-label classification problem.

Data Preprocessing

The data preprocessing begins by reading and parsing the raw data file. This file is structured so that each review is separated by a double newline and contains multiple lines: an identifier, the actual review sentence, and a string with aspect-sentiment pairs. Next, a splitting by aspect function processes the aspect-sentiment string by removing unwanted characters and splitting it into individual pairs (for example, {"FOOD", "positive"}).

However, for Stage 1, only the aspect names are retained. We then process each review to extract the sentence and its list of aspects, skipping any reviews that do not contain at least two aspect-sentiment pairs to maintain data quality.

Once the raw aspects are extracted, we apply label encoding using scikit-learn's MultiLabelBinarizer, which transforms the list of aspect labels into a binary vector for each review. For instance, if a review mentions "FOOD" and "SERVICE," the corresponding positions in the binary vector are set to 1, while all others remain 0. The

MultiAspectFeedbackDataset class then leverages the PhoBERT tokenization to convert the review sentences into input IDs, attention masks, and their corresponding labels. Each review is either padded or truncated to a fixed sequence length of 128 tokens. Finally, the prepared dataset is batched using a DataLoader with a batch size of 16, ensuring efficient iteration during the training process.

Model Adjustment

We define a custom model, PhoBERTMultiLabelClassifier, which is tailored for the multi-label aspect extraction task. This model loads the pretrained PhoBERT encoder and applies a dropout layer with a dropout rate of 0.3 to mitigate overfitting. Following the dropout, a linear layer is used to map the encoder's [CLS] token output to logits corresponding to the number of aspect labels. The architecture is designed to extract multiple aspects from a single review by generating independent logit outputs for each potential aspect.

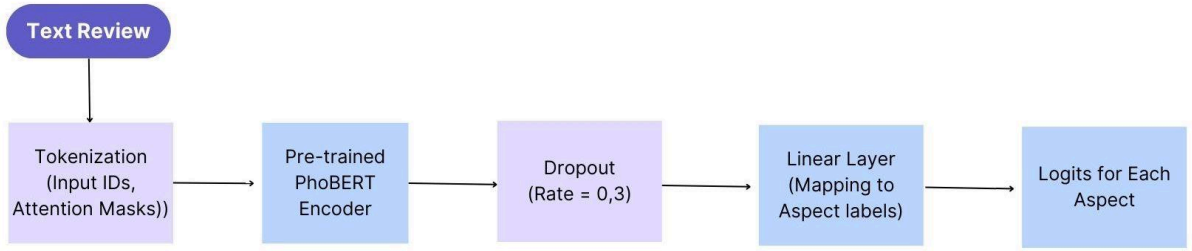

Figure 6: Adjustment in model architecture to extract main aspects.

Training Process

The model uses BCEWithLogitsLoss, which combines a sigmoid activation with binary cross-entropy loss-making it ideal for multi-label classification. Also, AdamW optimizer is chosen as the optimizer with a learning rate of 5e-5. Training runs for 5 epochs, with per-batch loss monitored using a tqdm progress bar. After training, the model's performance is evaluated on the validation set using metrics such as micro-averaged F1 score and accuracy. Binary predictions are generated by thresholding the output probabilities at 0.5. All this processing is done using Kaggle's T4 GPU P100, and the training loss is monitored using a tqdm progress bar. The loss decreases steadily, indicating effective learning.

Metric	F1 Score	Accuracy	Precision	Recall
Value	0.9625	0.8345	0.9297	0.9978

Table 2: Evaluate stage 1 model using standard metrics.

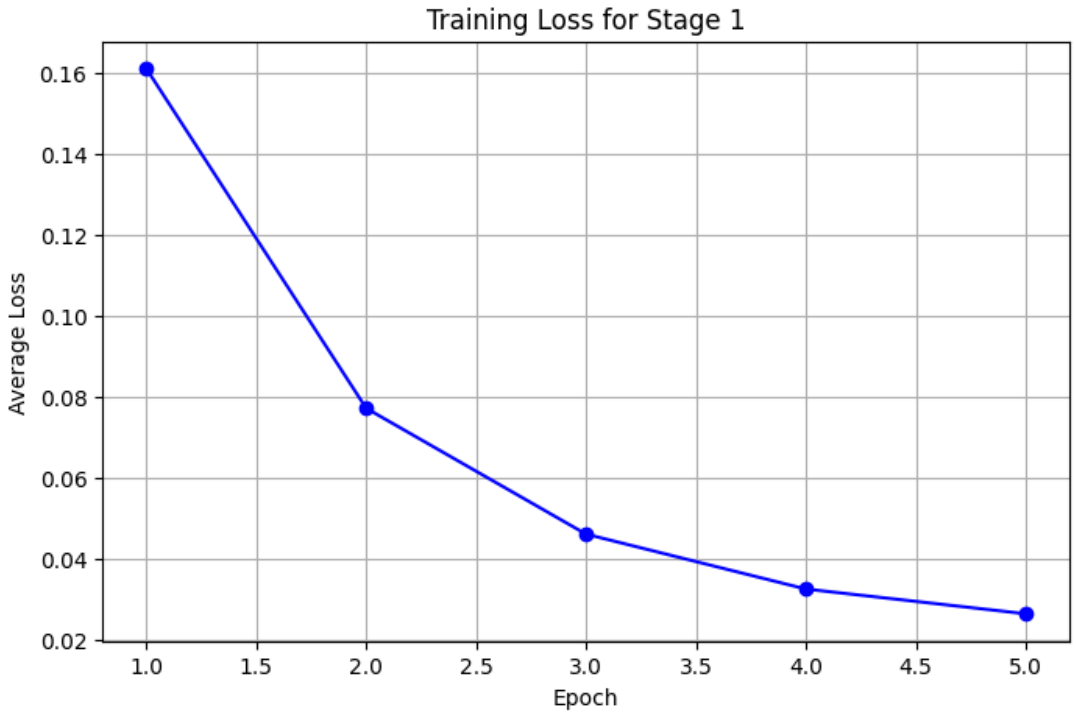

Figure 7: Plot of decreasing training loss across 5 epochs.

The Stage 1 model performs well in aspect extraction, with good recall suggesting that most aspects are correctly identified. The precision is slightly lower, indicating some false positives in aspect prediction, but overall, the model performs effectively in this stage. Once Stage 1 training is complete, the model's weights are saved (e.g., as absa_aspect_model.pt). These checkpoints are later used to transfer learned representations to Stage 2.

2.4 Stage 2: Sentiment Classification

For each aspect extracted in Stage 1, predict the corresponding sentiment (negative, neutral, or positive) as a single-label classification problem.

Data preprocessing

Each example is prepared by concatenating the review text with the aspect of interest in the following format:

"Review: [review text] | Aspect: [aspect]"

This combined input format provides essential context for sentiment classification. We then define a sentiment mapping ({"negative": 0, "neutral": 1, "positive": 2}) along with its inverse to decode predictions later. Raw sentiment examples are loaded into a Hugging Face Dataset, where a mapping function constructs the combined input text. The dataset is subsequently tokenized

with a maximum sequence length of 256 tokens to capture extended context and is split into training and validation sets using both scikit-learn's train_test_split and the Hugging Face train_test_split method.

Model Adaptation

We begin by initializing a sentiment classifier using the AutoModelForSequenceClassification class, with PhoBERT-Base as its backbone. This model is designed for single-label classification, and its classifier head is configured to output three logits—each corresponding to one of the sentiment classes: negative, neutral, or positive.

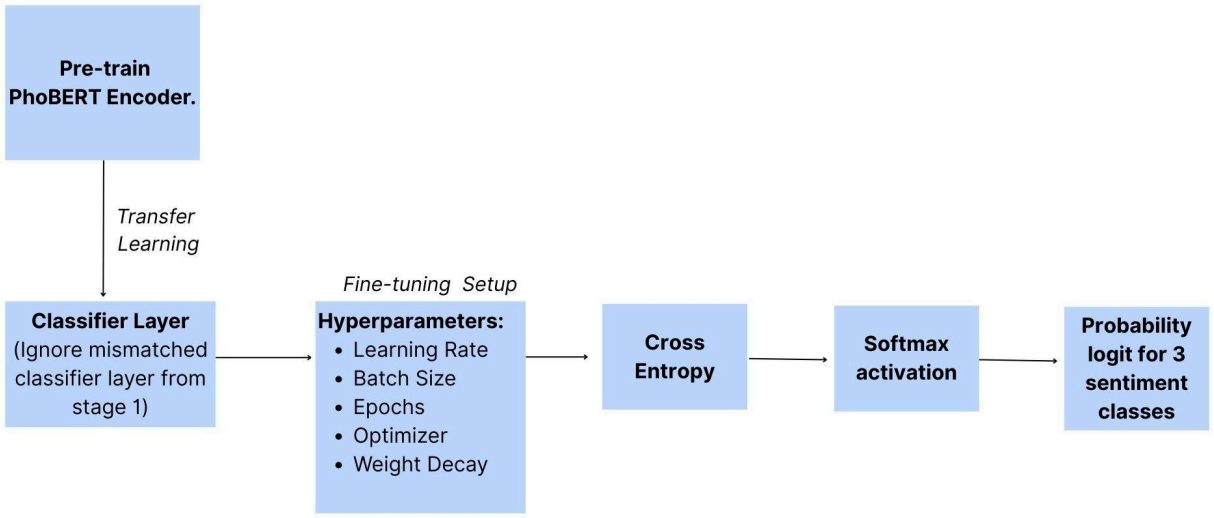

Figure 8: Pipeline of Sentiment Classification model (stage 2)

To effectively utilize the knowledge learned during Stage 1 (aspect extraction), we perform a weight transfer. Specifically, we load the encoder weights from the trained Stage 1 model into the Stage 2 sentiment classifier. During this transfer, we deliberately ignore the classifier weights from Stage 1 to resolve any dimensional mismatches between the models' output layers. This approach ensures that the rich, context-aware representations from the aspect extraction stage are seamlessly carried over to improve sentiment prediction.

Training Process

Once the model is initialized and adapted, training is carried out on Kaggle using GPU P100, and the Hugging Face Trainer API. The key training configurations include:

- Number of Epochs: Training is performed over 5 epochs, which strikes a balance between sufficient learning and computational efficiency.

- Batch Size: A batch size of 8 is used for both training and evaluation. This size is chosen to optimize memory usage and training speed without compromising the model's performance.
- Evaluation Strategy: Evaluation is conducted at the end of each epoch. The Trainer API computes the primary evaluation metric: accuracy by comparing the predicted sentiment labels with the true labels. The model checkpoint that achieves the best validation accuracy is retained.
- Learning Rate and Optimizer Settings: AdamW optimizer with a specific learning rate (e.g., 5e-5) is used to ensure stable training and optimal convergence. These settings are fine-tuned for effective learning during training.

Epoch	Training Loss	Validation Loss	Accuracy
1	0.487200	0.492715	0.855268
2	0.356500	0.440674	0.873748
3	0.199200	0.368026	0.897927
4	0.246200	0.372716	0.900518
5	0.204700	0.401167	0.910017

Table 3: Showing training loss, validation loss, and accuracy of 5 epochs in detail.

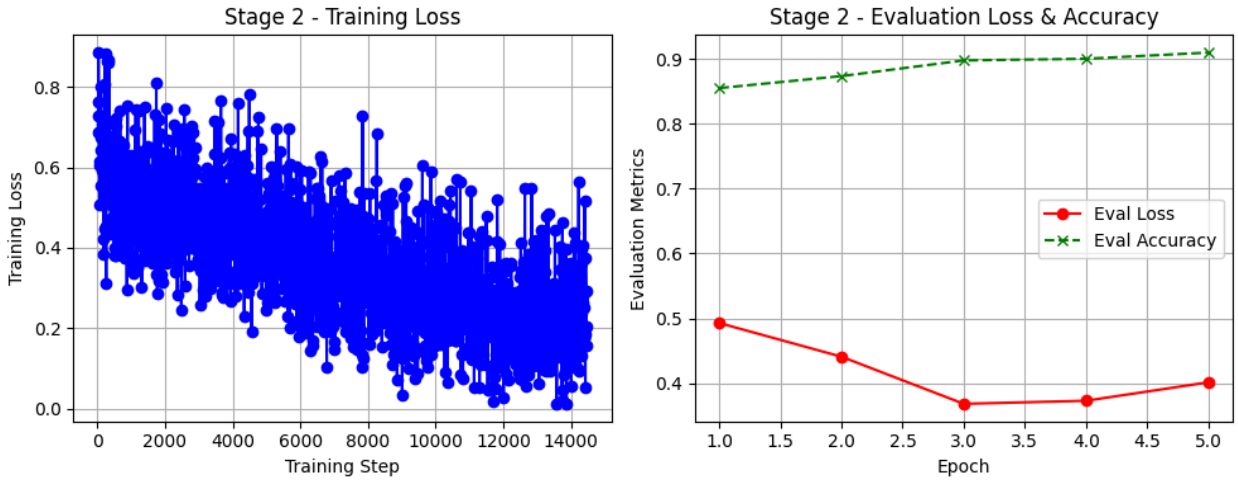

Figure 9: Training loss, evaluation loss, and accuracy were plotted using 2 subplots.

By leveraging the Trainer API, we benefit from built-in functionalities such as automatic checkpointing, logging, and evaluation, streamlining the training process and facilitating model reproducibility. This rigorous training setup enables the sentiment classifier to effectively learn the nuances of sentiment from the combined review-aspect inputs.

To assess the effectiveness of the combined pipeline, we evaluated its performance on the Stage 2 validation set by computing key classification metrics, including accuracy, precision, recall, and F1-score. The validation set consists of customer reviews, each annotated with various aspects and corresponding sentiment labels. To facilitate evaluation, we constructed a ground truth dictionary where each review was mapped to its respective aspects and sentiment labels. Using the trained Stage 1 and Stage 2 models, the pipeline processed each review to generate predicted sentiment labels for each aspect. The results were then stored and compared against the ground truth. All details we will describe in part V.

3. Single model approach

The idea of this approach simplifies the previous approach, here we will put into the review model along with the list of aspects and the model will now predict sentiment according to each aspect. In addition to the usual labels like positive, negative and neutral, we add none to denote that the aspect does not exist.

3.1 Prepare data and model

The data is divided into 3 sets: train, validation, test in the ratio of 80/10/10 using stratified sampling to maintain sentiment distribution. Because it is a classification problem, we handle the imbalanced data by class_weight according to the number of sentiment labels.

For training, a CustomDataset class prepares the data for the model by integrating pairs of sentences and aspects into a single input format, tokenizing inputs using PhoBERT's tokenizer, and producing datasets that are compatible with PyTorch. Using PyTorch Lightning, the ABSADataModule effectively controls batching and data loading for GPU training. The core model ABSAModel extends LightningModule and integrates PhoBERT with a custom classification head, featuring multiple linear layers and dropout for sentiment prediction across four classes (none, positive, negative, neutral).

3.2 Training process

The process was executed for 13 epochs, with a learning rate of 2e-5 for classification and 1e-5 for BERT, freezing most of the BERT layers except the last 4, dropout of 0.3 in the classifier head, optimization using AdamW and ReduceLROnPlateau learning rate scheduling based on F1 score validation. The batch size was 8 and the maximum sequence length was 256 tokens. All this processing is done using Kaggle's T4 GPU.

3.3 Evaluation

This problem will still be evaluated according to 2 parts Aspect Category Detection (ACD) and (Sentiment Polarity Classification) SPC. In terms of ACD, the labels will be divided into 2 types

of labels, normal and none. If the model predicts the correct label type, we will consider the aspect to be detected correctly. And the SPC part will be evaluated like a normal classification problem with metrics like accuracy, recall, precision, f1 score.

The following is the original metric on the testing set of our single model, the standard metrics we mentioned above will be calculated later.

Accuracy	**F1**	**Test_loss**
0.8911	0.8904	1.1790

Table 4: Evaluate single model using some standard metrics.

V. Results and Discussion

In addition to the PhoBERT approach, we present some other results taken from VLSP SHARE TASK: SENTIMENT ANALYSIS for the restaurant data domain to compare with our outcomes.

Task	Method	Precision	Recall	F1
Aspect Category Detection (ACD)	VSLP best submission	79.00	76.00	77.00
	Bi-LSTM + CNN	82.02	77.51	79.23
	BERT-based Hierarchical	-	-	84.23
	Multi-task PhoBERT	81.09	85.61	83.29
	Multi-task Multi-branch	80.81	87.39	83.97
	Our Combined-Model	**92.97**	**99.78**	**96.25**
	Our Single-Model	**93.91**	**93.94**	**93.92**
Sentiment Polarity Classification (SPC)	VSLP best submission	62.00	60.00	61.00
	Bi-LSTM + CNN	66.66	63.00	64.78
	BERT-based Hierarchical	-	-	71.30
	Multi-task PhoBERT	69.66	73.54	71.55
	Multi-task Multi-branch	68.69	74.29	71.38
	Our Combined-Model	**87.96**	**92.04**	**89.96**
	Our Single-Model	**92.31**	**92.45**	**92.38**

Table 5: Compare our models with other available models that solve the same tasks.

The table above shows the results of models doing the same problem as us with processed data. Our F1 for ACD and SPC significantly outperform the other models in both tasks, including VSLP best submissions, Bi-LSTM+CNN, and Multi-task PhoBERT, but **NOT** demonstrating the effectiveness of our two-model approach in capturing both aspects and sentiment polarity more precisely. (higher metrics do not indicate better performance (not on the same dataset), this is really just a tracking of the performance of our models).

In this project, we successfully fine-tuned PhoBERT for the Aspect-Based Sentiment Analysis (ABSA) task. By leveraging the pre-trained PhoBERT model, which is optimized for Vietnamese text, we were able to build a model that can effectively extract aspects and predict sentiment for each aspect in customer reviews.

However, despite the success in fine-tuning, our dataset remains a limitation due to two primary issues: the size and imbalance of the data. For the imbalance problem in the dataset, where certain sentiment classes or aspects are underrepresented, can lead to biased predictions and

suboptimal performance. For instance, the model might have difficulty predicting minority sentiment classes or aspects accurately, leading to lower precision or recall in those areas.

In future work, we plan to address these limitations by implementing the model with a larger, more diverse dataset. Furthermore, we will explore research methods to handle the imbalance problem. Another promising direction is the use of generative models or data augmentation techniques to artificially expand the dataset, especially for the underrepresented sentiment classes. By addressing these challenges, we aim to improve the model's robustness, performance, and generalization on real-world ABSA tasks.

VI. References

[1]. N. Zhang, "Design and Implementation of Aspect-Based sentiment Analysis Task," Comput. Sci. Math. forum, vol. 8, no. 1, art. no. 56, Aug. 2023, doi: 10.3390/cmsf2023008056.

[2]. A. Nazir, Y. Rao, L. Wu, and L. Sun, "A Survey on Aspect-Based Sentiment Analysis," IEEE Transactions on Affective Computing, vol. 13, no. 2, pp. 845-863, Apr.-Jun. 2022, doi: 10.1109/TAFFC.2020.2970399.

[3]. N. T. Hoai et al., "PhoBERT: Pre-trained language models for Vietnamese," arXiv preprint arXiv:2003.00744, 2020.

[4]. ds4v, "absa-vlsp-2018: End-to-end Multi-task Solutions for Aspect Category sentiment Analysis (ACSA) on Vietnamese reviews, using PhoBert as pretrained model," GitHub, 2021. [Online]. Available: https://github.com/ds4v/absa-vlsp-2018

[5.] A. L. Chau et al., "Multi-task Solution for Aspect Category sentiment Analysis on Vietnamese Datasets," IEEE Access, vol. 10, pp. 7732-7745, 2022, doi: 10.1109/Access.2022.3174812.