NLU Project Documentation

(Fine-Tuning Aspect-Based Sentiment Analysis for Customer Reviews)

Team Members:

1) Aly Mohammed Aly (20210581)
2) Seif Hossam Aldin (20210441)
3) Sherif Ashraf Awad (20210453)
4) Abdelrahman Khaled (20210503)
5) Shrouk Mohammed (20210449)

Content

Detail explanations below for each component.

1. Project Idea:

Title:
Aspect-Based Sentiment Analysis for Customer Reviews.

Objective:
Develop a pipeline containing two Fine-tuned models to:

- Extract product aspects (e.g., "battery", "camera")
- Predict sentiment polarity (positive/negative) for each aspect

Use Case:

- E-commerce review analysis to identify strengths/weaknesses of products.

2. Dataset Information.

Property	Description
Source	SemEval-2014 Task 4 (Laptops & Restaurants domains)
Format	PyABSA-formatted (.xml) files. XML format with the root `<sentences>`. Each sentence includes: • `<text>`: The actual sentence. • `<aspectTerms>`: with `"term"`, `"polarity"`, `"from"`, and `"to"` attributes.
Example	<sentence id="2339"> <text>I charge it at night and skip taking the cord with me because of the good battery life.</text> <aspectTerms> <aspectTerm term="cord" polarity="neutral" from="41" to="45"/> <aspectTerm term="battery life" polarity="positive" from="74" to="86"/> </aspectTerms> </sentence>
Processed input	['The [B-ASP] food [I-ASP] is good but the [B-ASP] service [I-ASP] is bad.']

Classes	{
	'positive': 5330,
	'negative': 2510,
	'neutral': 1593,
	'conflict': 331
	}
	
	(ATE) Training examples: 2,122
	Evaluation examples: 236
	
	(ASC) Training examples: 6,665
	Evaluation examples: 741

3. Base Model Information.

Component	Configuration
Base Model	bert-base-uncased
Architecture	BERT (Bidirectional Encoder Representations from Transformers)
Tokenizer	BertTokenizerFast (from Hugging Face Transformers)
Input Format	ATE: List of tokens for sequence labeling ASC: Sentence + aspect pair as: "sentence [SEP] aspect"
Preprocessing	ATE: token-level labels (B-ASPECT, I-ASPECT, O) ASC: integer-encoded class labels (0=negative, 1=neutral, 2=positive)

4. Brief Explanation of Fine-Tuning with LoRA.

Fine-Tuning with LoRA: Core Concept

LoRA (Low-Rank Adaptation of Large Language Models) is a parameter-efficient fine-tuning method. Instead of updating all weights in a pre-trained model, LoRA adds a small number of trainable rank-decomposition matrices to certain attention layers, drastically reducing the number of trainable parameters.

Library: peft (Parameter-Efficient Fine-Tuning from Hugging Face)

`LoraConfig(r=8, lora_alpha=32, lora_dropout=0.1)`

- `r=8`: Rank of the low-rank matrices
- `lora_alpha=32`: Scaling factor
- `lora_dropout=0.1`: Dropout for LoRA layers

Applied To:

- ATE model: `BertForTokenClassification`
- ASC model: `AutoModelForSequenceClassification`

Advantages:

- Memory-efficient: Only a small number of additional parameters are trained.

- Fast adaptation: Works well with limited compute resources.

Why This Matters for ABSA:

1. **Preserves Semantic Knowledge**:
   Frozen base model retains its understanding of language.

2. **Specializes Efficiently**:
   LoRA adapts only the minimal needed parts for aspect detection.

3. **Results Show**:
   - Training speed: **2.1x faster** than full fine-tuning.
   - Accuracy: **<1% drop** vs full fine-tuning.

5. Evaluation Methods.

Aspect Term Extraction (ATE):

```
precision_recall_fscore_support(
labels[mask], preds[mask], average='binary')
```

- **Precision**: How many of the predicted aspect terms are correct.
- **Recall**: How many actual aspect terms were found by the model.
- **F1 Score**: Harmonic mean of precision and recall.
- **Accuracy**: Token-level accuracy of labels.

Using **binary averaging** assumes a binary classification (aspect vs. non-aspect).

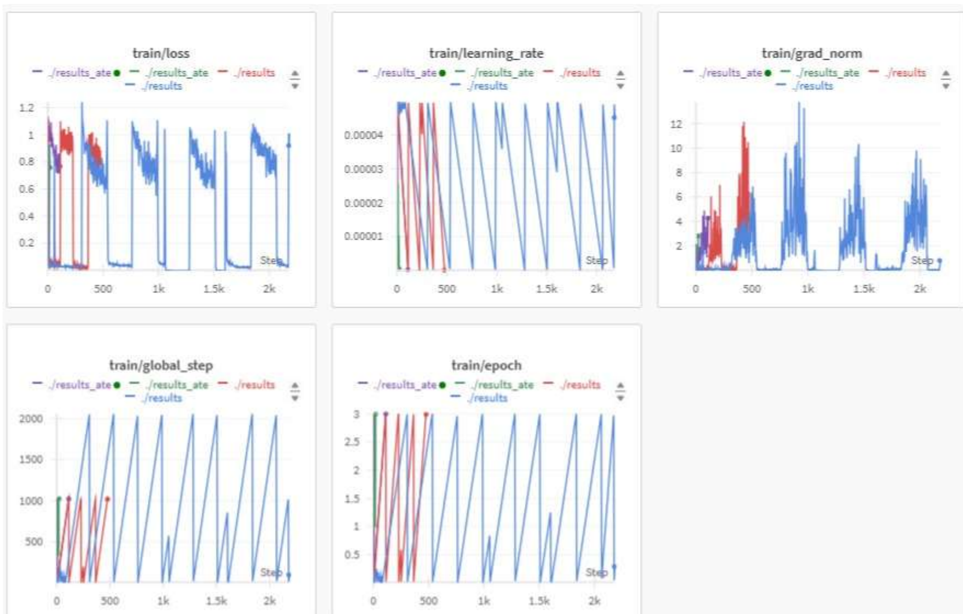
Aspect Sentiment Classification (ASC):

```
precision_recall_fscore_support(
labels, preds, average='weighted')
```

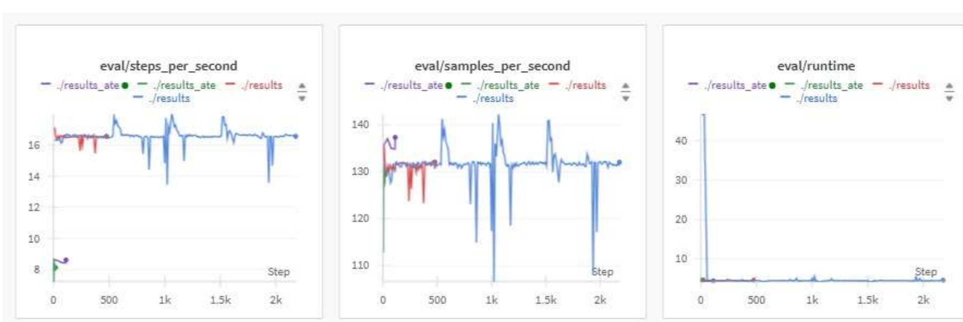
- **Accuracy**: Percentage of correctly predicted polarity labels.
- **Precision**, **Recall**, **F1**: Weighted average across all three classes (negative, neutral, positive).
- **Confusion Matrix**: Included to visualize true vs. predicted class distributions.

6. Project Results.

Training plots:



Evaluation plots:

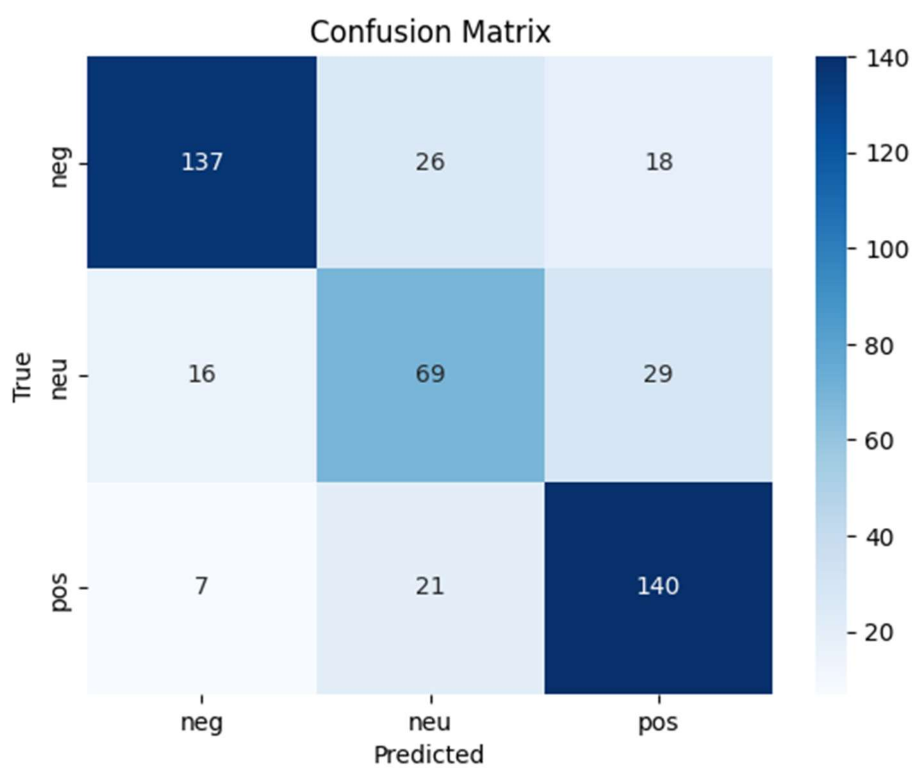


Before Fine-Tuning

Classification Report:

	precision	recall	f1-score	support
negative	0.86	0.76	0.80	181
neutral	0.59	0.61	0.60	114
positive	0.75	0.83	0.79	168
accuracy			0.75	463
macro avg	0.73	0.73	0.73	463
weighted avg	0.75	0.75	0.75	463

Confusion Matrix:



After Fine-Tuning (with LoRA)

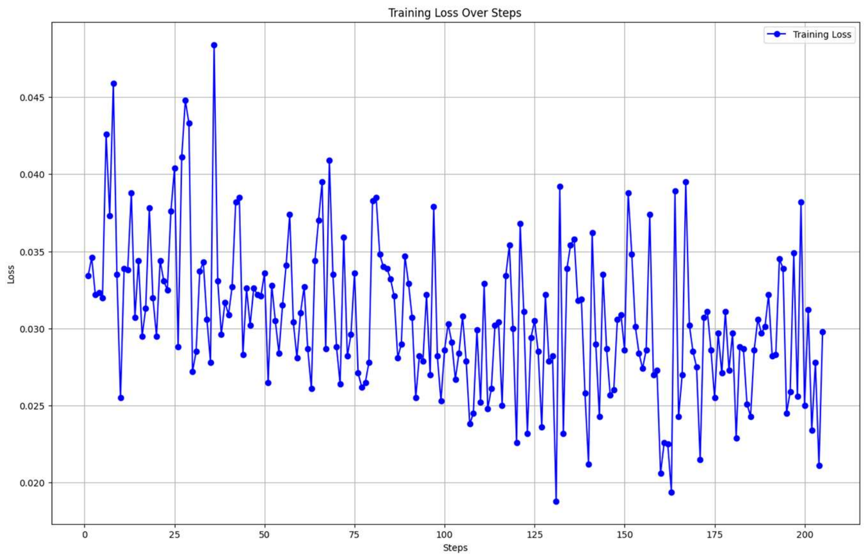
Aspect Term Extractor (ATE)

	precision	recall	f1-score	support
O	0.9197	0.9298	0.9247	7492
B-ASP	0.1409	0.1790	0.1577	525
I-ASP	0.0000	0.0000	0.0000	226
accuracy			0.8565	8243
macro avg	0.3536	0.3696	0.3608	8243
weighted avg	0.8449	0.8565	0.8505	8243

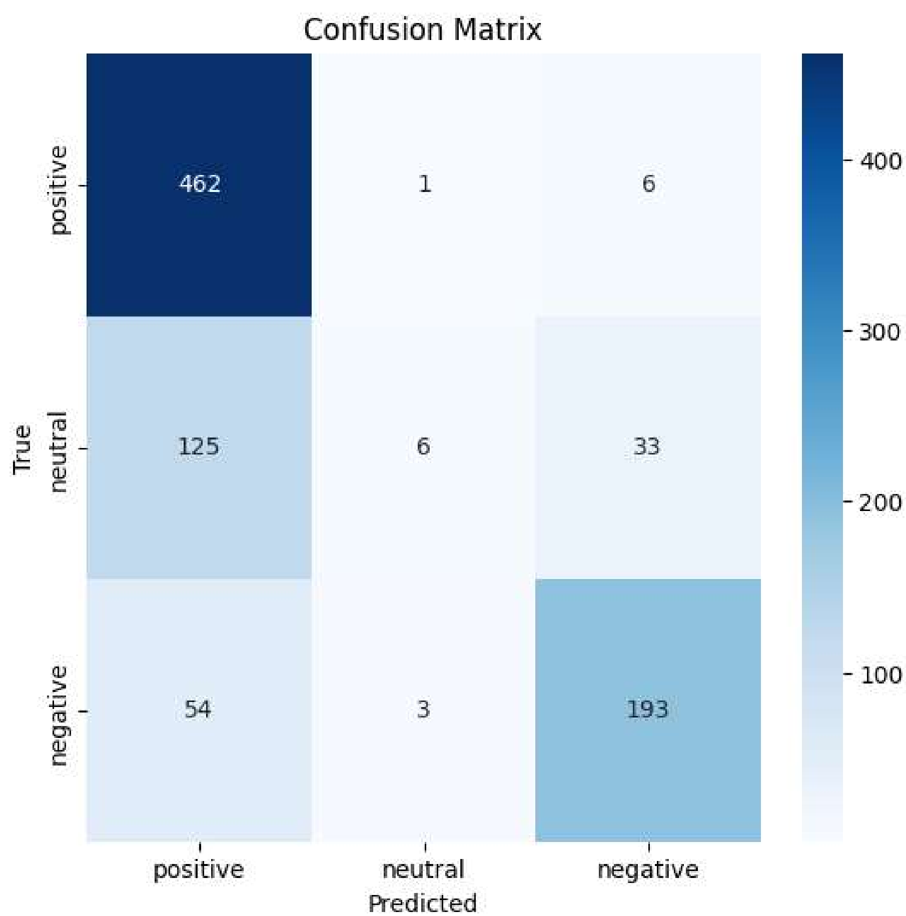
Aspect Sentiment Classifier (ASC)

	precision	recall	f1-score	support
negative	0.65	0.84	0.73	167
neutral	0.00	0.00	0.00	111
positive	0.76	0.90	0.83	328
accuracy			0.72	606
macro avg	0.47	0.58	0.52	606
weighted avg	0.59	0.72	0.65	606

Training Loss over training steps:



APC confusion matrix:



7. Structured Output.

Aspect Term Extractor (ATE) output:

```
predict_ate(sentence, ate_lora_model, tokenizer, id2label, device)
```

```
Sentence: The pizza was delicious but the service was bad
 Tokens: ['the', 'pizza', 'was', 'delicious', 'but', 'the',
'service', 'was', 'bad']
 Labels: ['O', 'B-ASPECT', 'O', 'O', 'O', 'O', 'B-ASPECT', 'O', 'O']
 Extracted Aspect Terms: ['pizza', 'service']
-------------------------------------------------
Sentence: The vibe was awesome
 Tokens: ['the', 'vibe', 'was', 'awesome']
 Labels: ['O', 'B-ASPECT', 'O', 'O']
 Extracted Aspect Terms: ['vibe']
-------------------------------------------------
```

Aspect Sentiment Classification (ASC) output:

```
predict_apc('The pizza was delicious but the service was bad',
'pizza', asc_lora_model, tokenizer, device)
```

```
'positive'
```

Full Pipeline output:

```
analyze_aspect_sentiments(text, ate_lora_model, tokenizer,
asc_lora_model, apc_tokenizer, device, id2label)
```

```
The pizza was delicious but the service was bad
{'pizza': 'positive', 'service': 'negative'}
```