

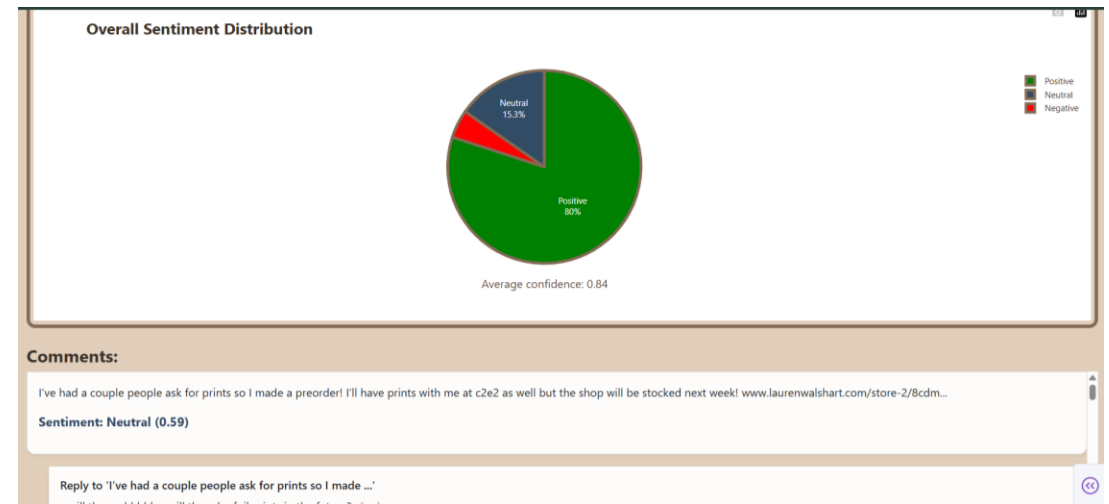
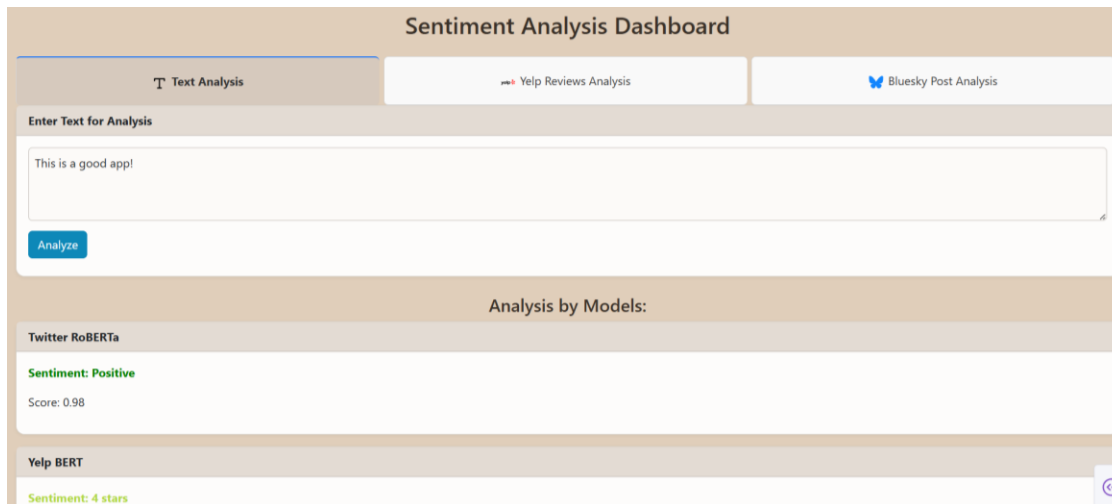
Team 7

Text Sentiment Analysis

Team Members: Ross Kugler, Jason Eigeman, Justin Lawrence,
Audrey Lindstrom, Linh (Jason) Nguyen, Huy (Harry) Ky

Project Overview

- Problem: How to analyze and classify the overall sentiment of text comments/reviews?
- Solution: Web app that utilizes AI models trained on text sentiment analysis to provide insights



Datasets

For Training:

- GoEmotions
 - human-annotated emotion data
 - dominant heuristic uses [emotion-related Twitter tags](#)

For Testing/Using with our Models:

- Yelp Academic Dataset
- Bluesky API

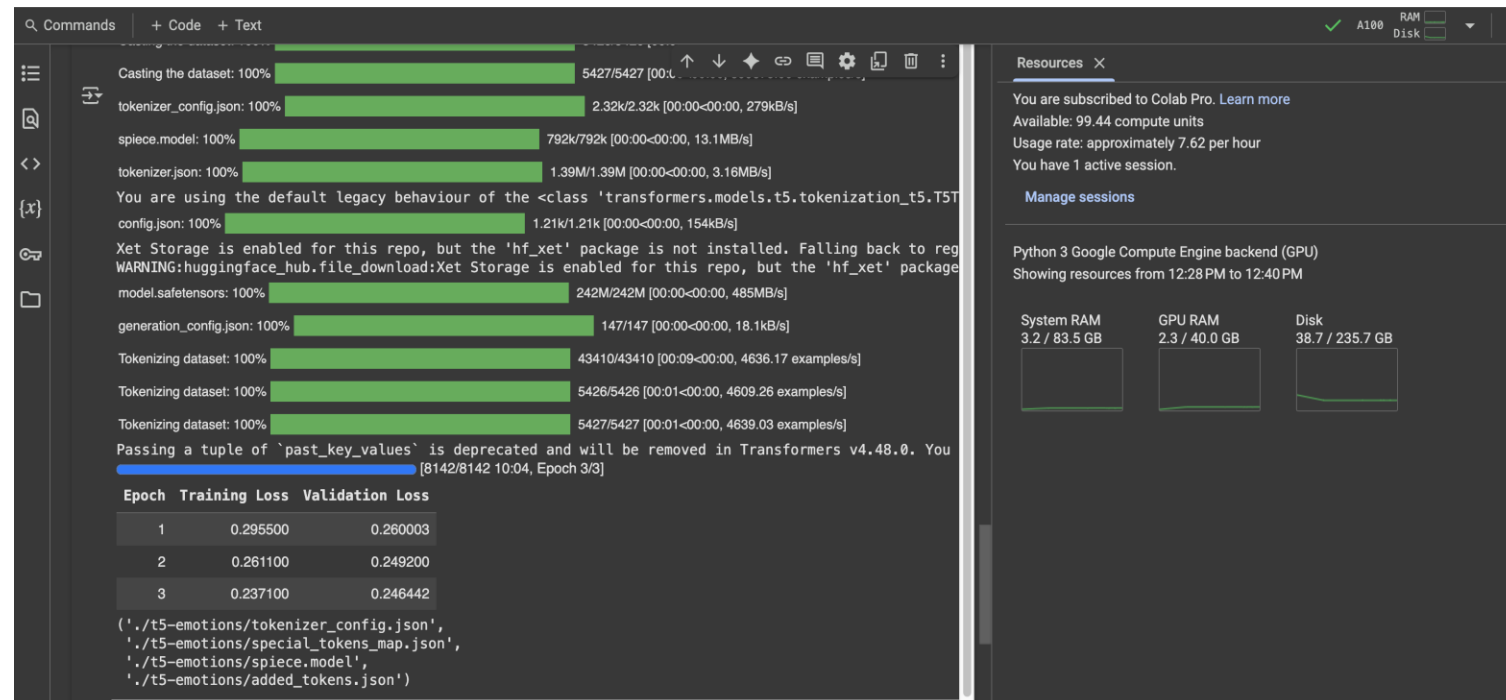
Models

- T5Emotions (custom model, fine-tuned on GoEmotions dataset)
- Twitter RoBERTa (pretrained, from HuggingFace)
- Yelp BERT (pretrained, from HuggingFace)

Model	Base Architecture	Dataset Source	Task Type	Output Format
GoEmotions	BERT-base	Reddit comments	Multi-label classification	List of emotions
T5Emotions	T5 (text-to-text)	GoEmotions	Text generation (emotions)	Generated label text
Twitter RoBERTa	RoBERTa-base	Twitter data	Classification	Single/multi emotion
Yelp BERT	BERT-base	Yelp reviews	Sentiment classification	Star rating or emotion


Fine-Tuning T5 with GoEmotions Dataset


- We utilized Google Colab Pro to access A100 to effectively train the model.
- With each epoch, both the training and validation loss consistently decreased, indicating that the model was effectively learning from the training data and improving its performance over time.





The screenshot displays the Google Colab Pro interface during the fine-tuning of a T5 model. The main window shows the progress of various steps, including casting the dataset and tokenizing it. A table at the bottom summarizes the training and validation loss over three epochs, showing a consistent decrease in loss. The right sidebar provides information about the Colab Pro subscription and the resources used, including system RAM, GPU RAM, and disk space.

```
Commands | + Code | + Text
```

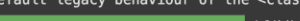
Casting the dataset: 100%  5427/5427 [00:00<00:00, 279kB/s]

tokenizer_config.json: 100%  2.32k/2.32k [00:00<00:00, 13.1MB/s]

spiece.model: 100%  1.39M/1.39M [00:00<00:00, 3.16MB/s]

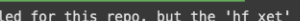
tokenizer.json: 100%  1.21k/1.21k [00:00<00:00, 154kB/s]

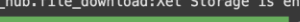
You are using the default legacy behaviour of the `<class 'transformers.models.t5.tokenization_t5.T5Tokenizer'>`


config.json: 100%  1.21k/1.21k [00:00<00:00, 154kB/s]


Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular file download.


WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular file download.

model.safetensors: 100%  242M/242M [00:00<00:00, 485MB/s]

generation_config.json: 100%  147/147 [00:00<00:00, 18.1kB/s]

Tokenizing dataset: 100%  43410/43410 [00:09<00:00, 4636.17 examples/s]

Tokenizing dataset: 100%  5426/5426 [00:01<00:00, 4609.26 examples/s]

Tokenizing dataset: 100%  5427/5427 [00:01<00:00, 4639.03 examples/s]

Passing a tuple of `past_key_values` is deprecated and will be removed in Transformers v4.48.0. You should pass a list of tuples instead.

[8142/8142 10:04, Epoch 3/3]

Epoch	Training Loss	Validation Loss
1	0.295500	0.260003
2	0.261100	0.249200
3	0.237100	0.246442

```
( './t5-emotions/tokenizer_config.json',  
  './t5-emotions/special_tokens_map.json',  
  './t5-emotions/spiece.model',  
  './t5-emotions/added_tokens.json' )
```

Resources

You are subscribed to Colab Pro. [Learn more](#)

Available: 99.44 compute units

Usage rate: approximately 7.62 per hour

You have 1 active session.

[Manage sessions](#)

Python 3 Google Compute Engine backend (GPU)

Showing resources from 12:28 PM to 12:40 PM

System RAM: 3.2 / 83.5 GB

GPU RAM: 2.3 / 40.0 GB

Disk: 38.7 / 235.7 GB

Fine-Tuned T5 Metrics

- Utilized GoEmotions Test data to run metrics on how well the model learned from the training data.
- Micro F1 Score: Measures the model's overall ability to correctly predict all emotions. Higher is better.
- Macro F1 Score: Averages the model's performance on each individual emotion. Higher is better.
- Subset Accuracy: Calculates how often the model predicts all the correct emotions for a sentence with no mistakes. Higher is better.
- Hamming Loss: Shows the percentage of individual label predictions the model got wrong. Lower is better

-----Evaluation Metrics on Google_Emotions-----

Test set was used for metrics.

The model was fine tuned using the training set

Micro F1 Score: 0.5940

Micro F1: The model was accurate 59.40% of the time across all emotion predictions.

Macro F1 Score: 0.4775

Macro F1: On average, the model was 47.75% accurate across each individual emotion.

Subset Accuracy: 0.5222

Subset Accuracy: The model made perfect emotion predictions for 52.22% of sentences.

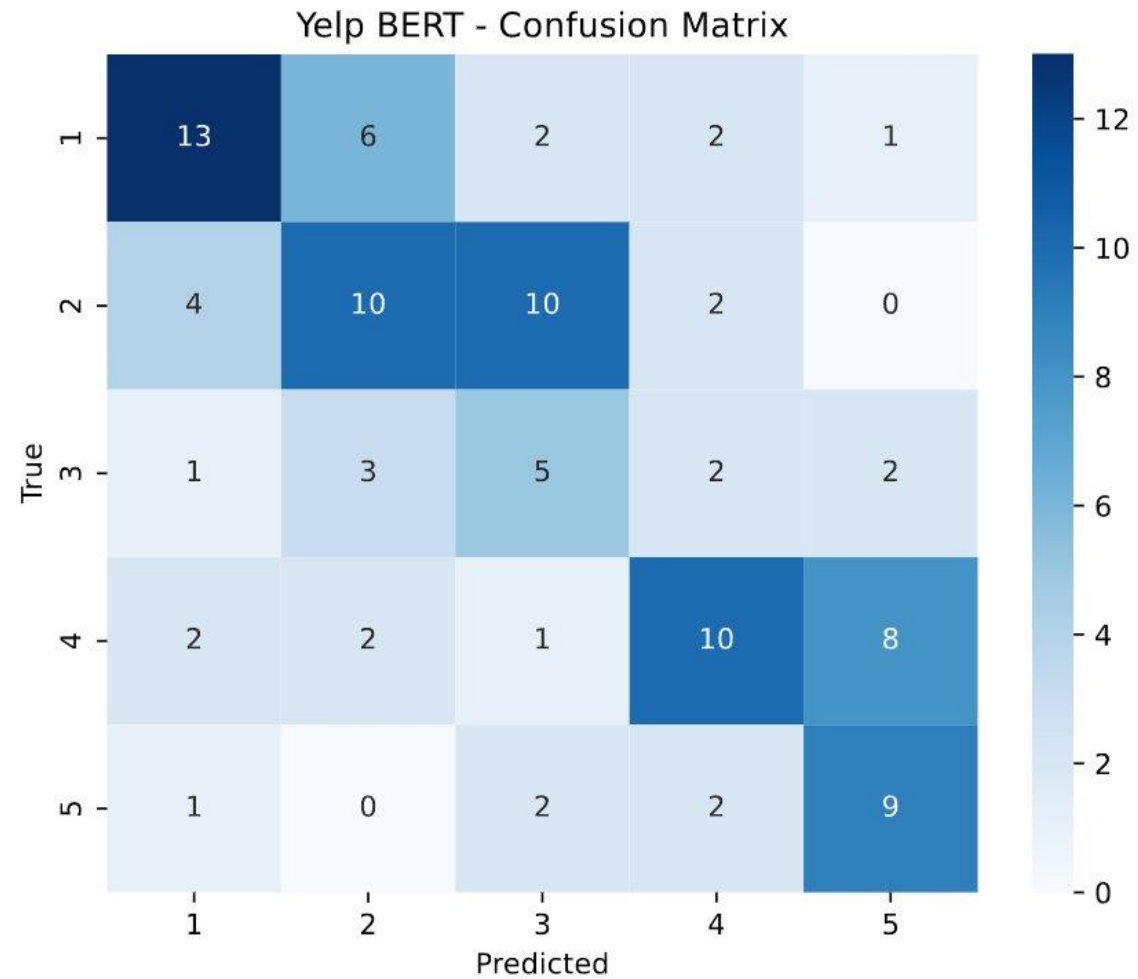
Hamming Loss: 0.0320

Hamming Loss: On average, the model made mistakes on 3.20% of possible emotion labels.

Evaluation

	Model	Micro F1	Macro F1	Accuracy	Hamming Loss	MAE	Exact Match	Within ± 1
0	T5Emotions	0.6061	0.3711	0.51	0.0325	-	-	-
1	GoEmotions	0.6239	0.3901	0.54	0.0293	-	-	-
2	Twitter RoBERTa	0.4	0.3315	0.4	0.6	-	-	-
3	Yelp BERT	0.47	0.4647	0.47	0.53	0.78	0.47	0.83

Evaluation



Project Difficulties

- Data Quality
 - Typos, Negations, and Sarcasm
- Combining and Normalizing different datasets
- Time-Constraints (would improve UI, more time to train models)