Assignment 7: Tweet Emotion analysis with LLMs

HXD220000

Harikrishna Dev

Tasks

The task was to identify emotions in Tweets using the labelled dataset provided. The task included building an RNN or LSTM and finetuning language models using PEFT techniques and HuggingFace libraries.

Dataset

This dataset was part of the **SemEval-2018 Task 1: Affect in Tweets competition.** It consisted of the *tweet* and the *11 labels* ('anger', 'anticipation', 'disgust', 'fear', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust').

Task 1:

I used *google/gemma-1.1-2b-it* using the following methods initially.

- Low Rank Adaption LoRa
- Inhibiting and Amplifying Inner Activations (IA3)

Task 2:

Use a model from the MTEB Leaderboard.

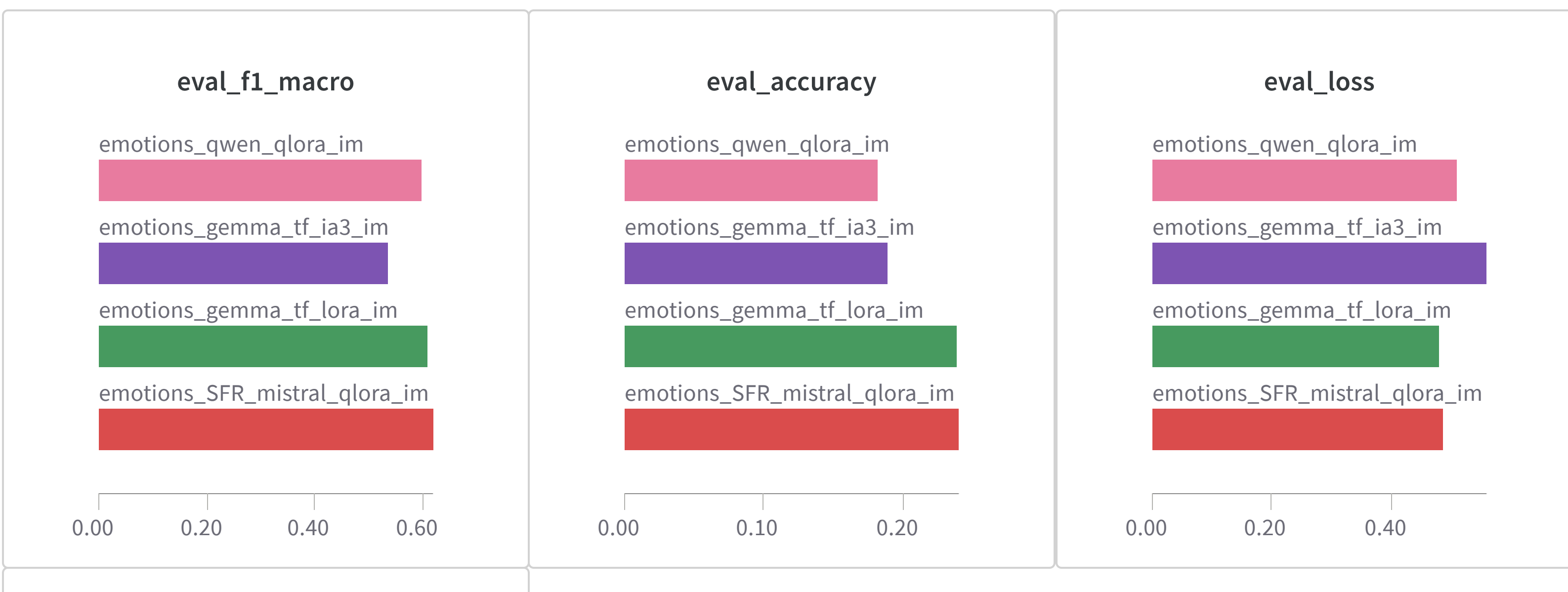
I used the Salesforce/SFR-Embedding-Mistral

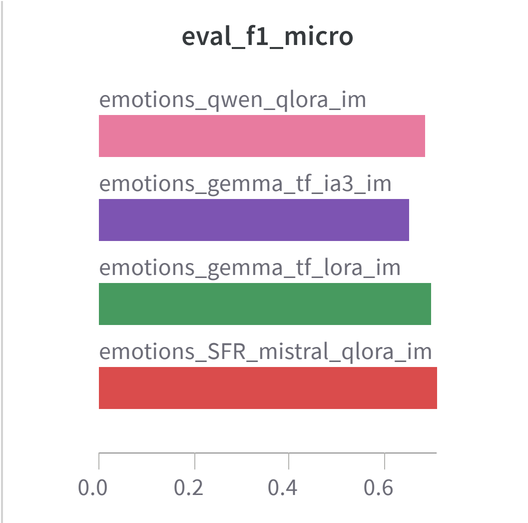
This model is trained on top of E5-mistral-7b-instruct and Mistral-7B-v0.1.

Task 3:

I used michellejieli/emotion_text_classifier for zero-shot classification.

Evaluation metrics from the following methods





Hyper parameter tuning

LoRa

While finetuning with LoRA, I used the following configs.

```
gemma_peft_config = LoraConfig(
    task_type=TaskType.SEQ_CLS,
    r=128,
    lora_alpha=256,
    lora_dropout=0.01,
    bias="lora_only",
    # modules_to_save = ['score'],
    target_modules="all-linear",
)
```

```
gemma_peft_model = get_peft_model(model, gemma_peft_config )
gemma_peft_model.print_trainable_parameters()
```

1. `task_type=TaskType.SEQ_CLS`: This parameter specifies the task type as sequence classification, indicating that the model will be used for classifying sequences.

2. `r=128`: The `r` parameter is a hyperparameter determining the number of attention heads in the model. A value of 128 indicates that the model will have 128 attention heads.

3. `lora_alpha=256`: This parameter controls the size of the feedforward layers in the model. A value of 256 indicates that the feedforward layers will have 256 units.

4. `lora_dropout=0.01`: Dropout is a regularization technique to prevent overfitting. This parameter specifies the dropout rate, which is set to 0.01, meaning that 1% of the input units will be randomly set to 0 at each update during training.

5. `bias="lora_only"`: This parameter specifies how bias terms are used in the model. Setting it to `"lora_only"` indicates that bias terms are only added to the linear layers in the model.

6. `target_modules="all-linear"`: This parameter specifies which modules of the model should be saved. Setting it to `"all-linear"` indicates that only the linear layers of the model will be saved.

I used all linear layers as I thought they would help in finetuning the model better.

IA3

I used a similar strategy for IA3.

```
gemma_peft_config = IA3Config(
    task_type=TaskType.SEQ_CLS,
    peft_type="IA3",
    target_modules="all-linear",
```

```
)
gemma_peft_model = get_peft_model(model, gemma_peft_config)
gemma_peft_model.print_trainable_parameters()
```

MTEB Model

I used LoRA

```
mistral_peft_config = LoraConfig(
    task_type=TaskType.SEQ_CLS,
    r=128,
    lora_alpha=256,
    lora_dropout=0.01,
    bias="lora_only",
    modules_to_save = ['score'],
    target_modules = ['v_proj',  'q_proj',  'up_proj', 'o_proj', 'down_proj', 'gate_proj','k_proj
```

I used a similar strategy to my Mistral tuning as it gave me the best result on Kaggle.

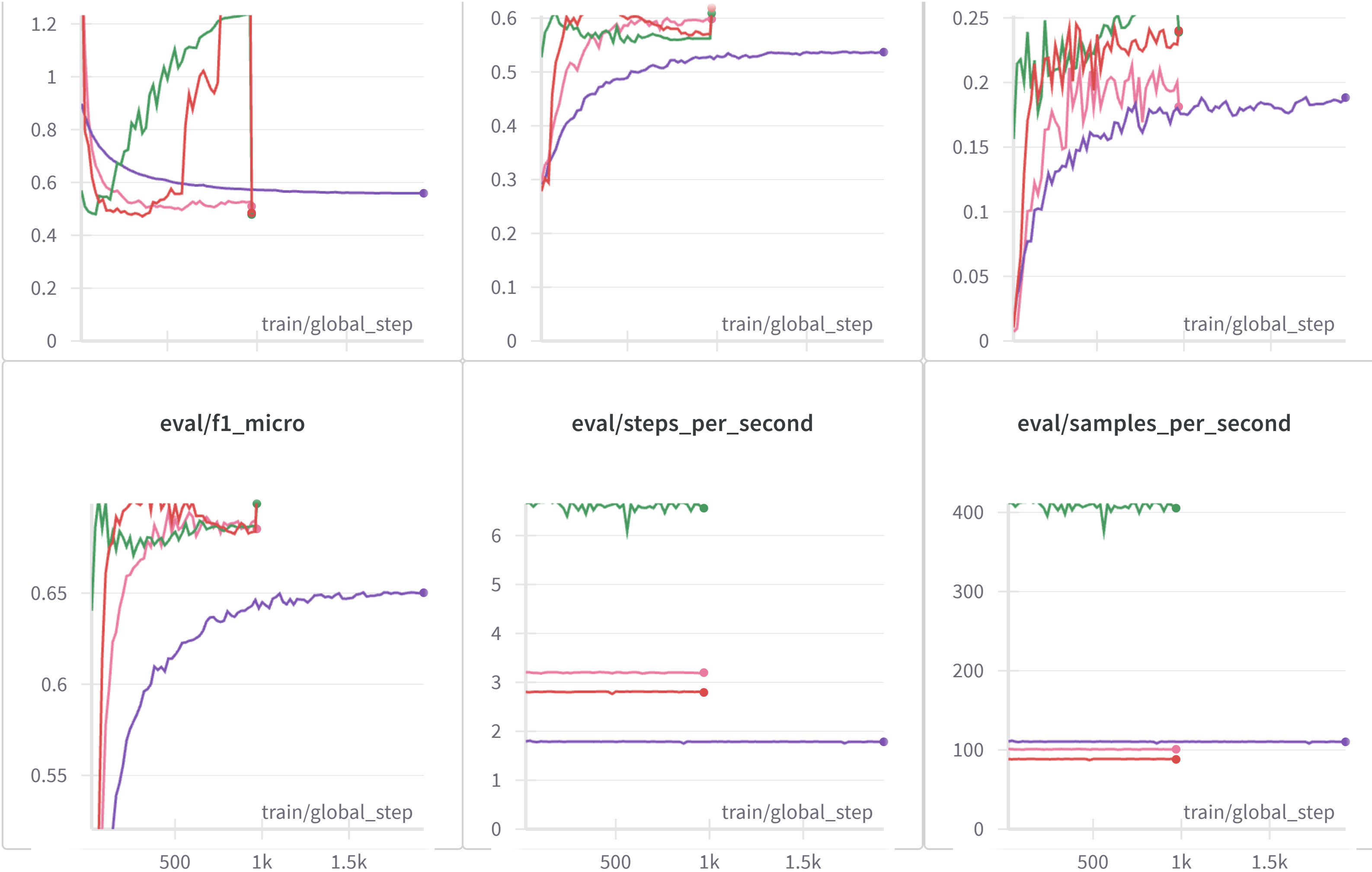
Conclusions

- LoRA has always produced better results and took less time to compute
- Mistral was better tuned for my tasks than Gemma.
- A bigger GPU like A100 or L4 was crucial as it helped me run bigger batch sizes.
- Gradient accumulation helped in faster computational effeciency

eval/loss

eval/f1_macro

eval/accuracy



Link: https://wandb.ai/harikrishnad/nlp_course_spring_2024-emotion-analysis-hf-trainer-hw8?nw=nwuserharikrish0607

Created with ❤️ on Weights & Biases.

https://wandb.ai/harikrishnad/nlp_course_spring_2024-emotion-analysis-hf-trainer-hw8/reports/Assignment-7-Tweet-Emotion-analysis-with-LLMs--Vmlldzo3ODUxODQy