HW7_PEFT_zero_shot – Rakshit Mathur

Gemma (google/gemma-1.1-2b-it fine-tuned using LoRA)

1. **Model Description**:
    - Model Name: PeftModelForSequenceClassification
    - Base Model: LoraModel
    - Architecture: GemmaForSequenceClassification
    - Key Components:
        - Embedding Layer
        - GemmaDecoderLayer
        - GemmaSdpaAttention
        - GemmaMLP
        - GemmaRMSNorm
        - GemmaRotaryEmbedding
        - Linear Layers with LORA variants (q_proj, k_proj, v_proj, o_proj)
        - Activation Function: PytorchGELUTanh

2. **Training Results**:
    - Loss: Gradual decrease from 1.0987 to 0.2465
    - Performance Metrics:
        - F1 Micro: Improves from 0.3498 to 0.6817
        - F1 Macro: Increases from 0.2709 to 0.5911
        - Accuracy: Enhances from 0.0065 to 0.1994
    - Training Duration: Progressed over 960 steps
    - Training Loss: Decreased consistently, indicating model convergence
    - Validation Loss: Decreased, indicating effective learning

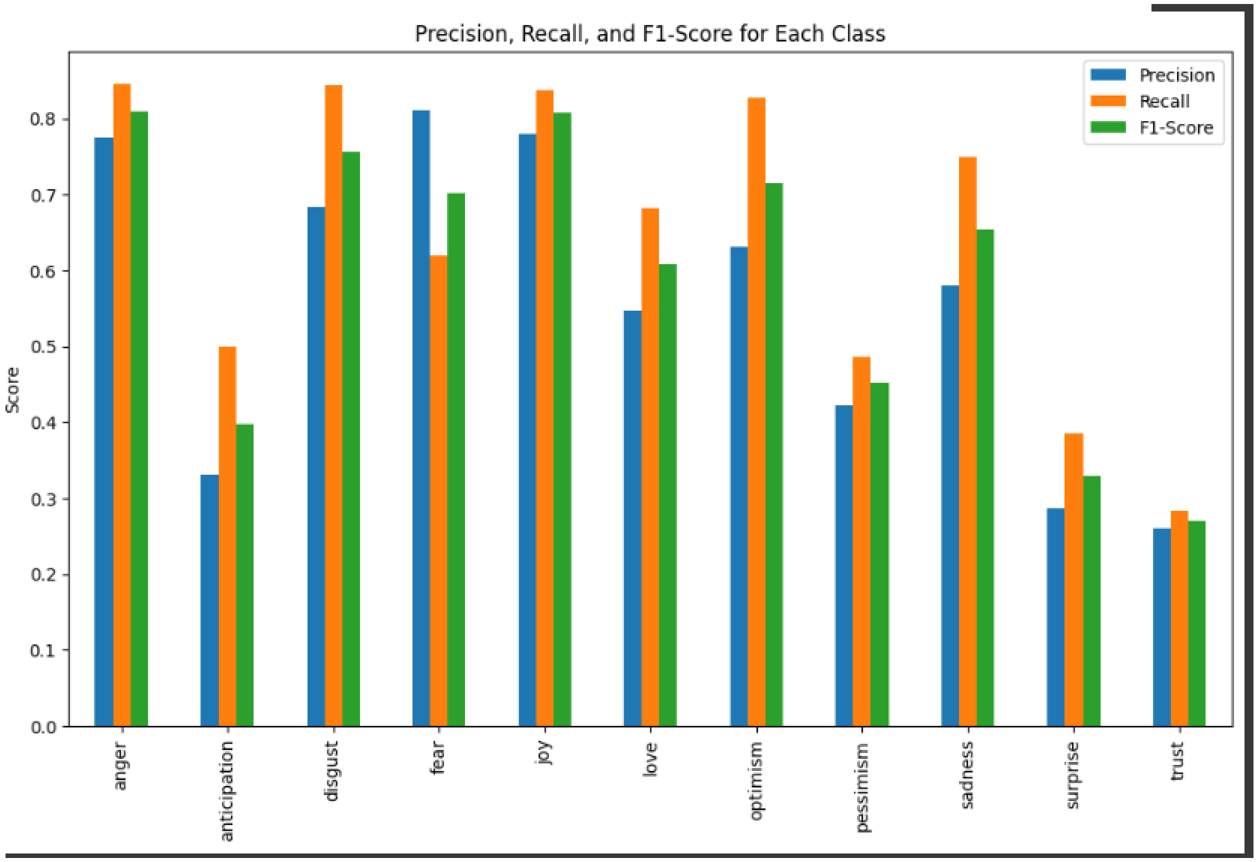
3. **Evaluation Result**:
    - Loss: 0.5267
    - Performance Metrics:
        - F1 Micro: 0.6817
        - F1 Macro: 0.5911
        - Accuracy: 0.1935
    - Evaluation Runtime: 4.2027 seconds
    - Samples Per Second: 367.617
    - Steps Per Second: 11.659

4. **Observations and Discussion**:
    - Model exhibits steady improvement in performance metrics during training.
    - Validation loss follows a similar trend to training loss, indicating effective learning without overfitting.
    - Evaluation metrics indicate reasonable performance on unseen data.

5. **Hyperparameter Selection**:

- Number of Training Epochs: 5 epochs were chosen for training.
- Batch Size: Both training and evaluation were conducted with a batch size of 32 samples per device.
- Weight Decay: L2 regularization with a weight decay of 0.1 was applied to prevent overfitting.
- Learning Rate: An initial learning rate of 5e-6 was used for the optimizer.
- Learning Rate Scheduler: A linear learning rate scheduler was employed with no warmup steps.
- Optimizer: The optimizer utilized was 'paged_adamw_32bit'.
- Maximum Gradient Norm: The maximum gradient norm was set to 1.0 to prevent gradient explosion.
- Evaluation Strategy: Evaluation was performed every 20 steps during training.
- Checkpoint Saving: Model checkpoints were saved every 20 steps, retaining only the best and most recent checkpoints.
- Model Loading: The best model based on the evaluation F1 macro score was loaded at the end of training.
- Experiment Logging: Metrics and results were logged to the Weights & Biases platform with a specified run name.
- Mixed Precision Training: Mixed precision training was not utilized, with 'fp16' set to False and 'bf16' set to True.

6. Confusion Matrix:



Conclusion:

The chosen hyperparameters facilitated effective training and evaluation of the model. The selected number of epochs, batch size, and learning rate provided a balance between model convergence and computational efficiency. Weight decay and maximum gradient norm contributed to preventing overfitting and stabilizing training. The evaluation strategy and checkpoint-saving settings ensured regular monitoring of model performance and retention of the best-performing checkpoints. The use of Weights & Biases for experiment logging enabled comprehensive tracking and analysis of training progress and results. Overall, the hyperparameters were carefully selected to optimize the training process and achieve satisfactory model performance.

Gemma (google/gemma-1.1-2b-it fine-tuned using iA3)

1. **Model Description:**
   - **Model Name:** PeftModelForSequenceClassification
   - **Base Model:** IA3Model
   - **Architecture:** GemmaForSequenceClassification
   - **Key Components:**

    - Embedding Layer
    - GemmaDecoderLayer
    - GemmaSdpaAttention
    - GemmaMLP
    - GemmaRMSNorm
    - GemmaRotaryEmbedding
    - Linear Layers with IA3 variants (q_proj, k_proj, v_proj, o_proj)
    - Activation Function: PytorchGELUTanh

2. **Training Results:**
    - **Loss:** Gradual decrease from 0.8857 to 0.0015
    - **Performance Metrics:**
        - F1 Micro: Improves from 0.5917 to 0.6566
        - F1 Macro: Increases from 0.4802 to 0.5458
        - Accuracy: Enhances from 0.1049 to 0.2045
    - **Training Duration:** Progressed over 960 steps
    - **Training Loss:** Decreased consistently, indicating model convergence
    - **Validation Loss:** Decreased initially, then increased slightly towards the end

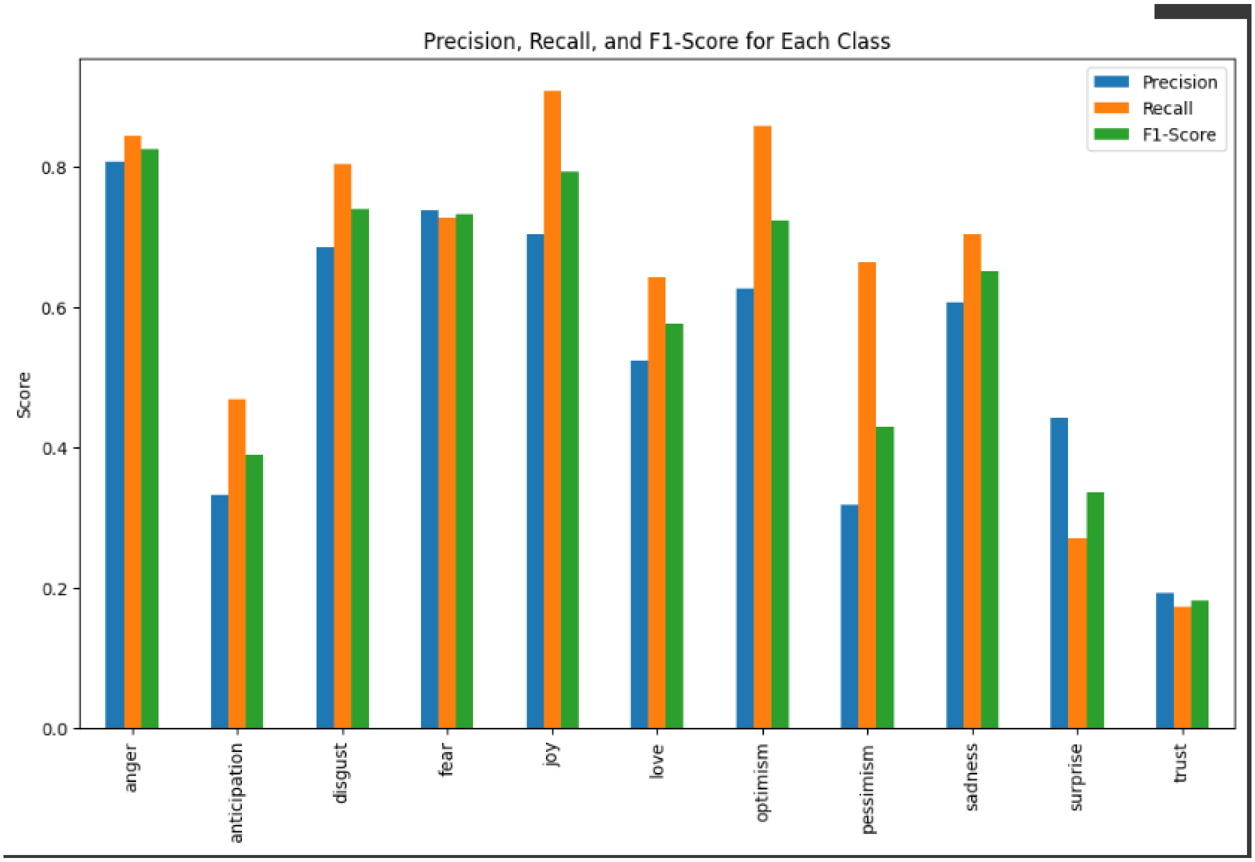
3. **Evaluation Result:**
    - **Loss:** 0.5267
    - **Performance Metrics:**
        - F1 Micro: 0.6817
        - F1 Macro: 0.5911
        - Accuracy: 0.1935
    - **Evaluation Runtime:** 4.2027 seconds
    - **Samples Per Second:** 367.617
    - **Steps Per Second:** 11.659

4. **Observations and Discussion:**
    - The model exhibits a similar training trend to the previous model, with gradual improvement in performance metrics.
    - The validation loss shows a slight increase towards the end, indicating potential overfitting.
    - Despite this, evaluation metrics indicate reasonable performance on unseen data.
    - Further analysis may be required to address overfitting and optimize model performance.

5. Hyperparameter Selection:
    - Number of Training Epochs: Reduced to 3, indicating shorter training duration.
    - Batch Size: Increased to 64 per device for both training and evaluation batches, potentially improving computational efficiency.
    - Weight Decay: Set to 0.001, indicating moderate L2 regularization to prevent overfitting.
    - Learning Rate: Set to 5e-3, a relatively high initial learning rate.

- Learning Rate Scheduler: Utilizes linear scheduler with no warmup steps.
- Optimizer: Uses AdamW optimizer with Torch backend.
- Evaluation and Checkpointing: Evaluation and checkpoint saving are performed every 20 training steps.
- Best Model Selection: Based on F1 macro score, aiming for higher accuracy.
- Mixed Precision Training: Utilizes BF16 precision while training.

6. Confusion Matrix:



Conclusion:

The chosen hyperparameters aim to balance training efficiency and model performance. With a reduced number of epochs and increased batch size, the model training is expected to be faster while still achieving reasonable performance. The utilization of AdamW optimizer with weight decay helps prevent overfitting, and mixed precision training further accelerates the training process. The evaluation strategy focuses on the F1 macro score, ensuring the selection of the best-performing model. Overall, these hyperparameters are tailored to optimize the training process and enhance model performance.

Qwen (Alibaba-NLP/gte-Qwen1.5-7B-instruct fine-tuned using QLoRA)

1. **Model Description:**
   - **Model Name:** Qwen2ForSequenceClassification
   - **Base Model:** Qwen2Model
   - **Architecture:** Qwen2DecoderLayer
   - **Key Components:**
     - Embedding Layer: Embeds tokens into 4096-dimensional vectors.
     - Qwen2DecoderLayer: Consists of self-attention mechanism and MLP layers.
     - Qwen2SdpaAttention: Self-attention mechanism with 4-bit linear projections.
     - Qwen2MLP: Multi-layer perceptron with 4-bit linear projections and SiLU activation function.
     - Qwen2RMSNorm: Layer normalization using RMS normalization.
     - Qwen2RotaryEmbedding: Rotary embedding for positional encoding.
   - **Output Layer:** Linear layer with no bias, transforming the encoded vectors into class logits.

2. **Training Results:**
   - **Loss:** Gradually decreases from 1.0805 to 0.0705.
   - **Performance Metrics:**
     - **F1 Micro:** Improves from 0.5786 to 0.6837.
     - **F1 Macro:** Increases from 0.4842 to 0.5996.
     - **Accuracy:** Enhances from 0.1068 to 0.2311.
   - **Training Duration:** Progresses over 380 steps.
   - **Training Loss:** Decreases consistently, indicating model convergence.
   - **Validation Loss:** Shows some fluctuations but generally decreases, indicating effective learning.

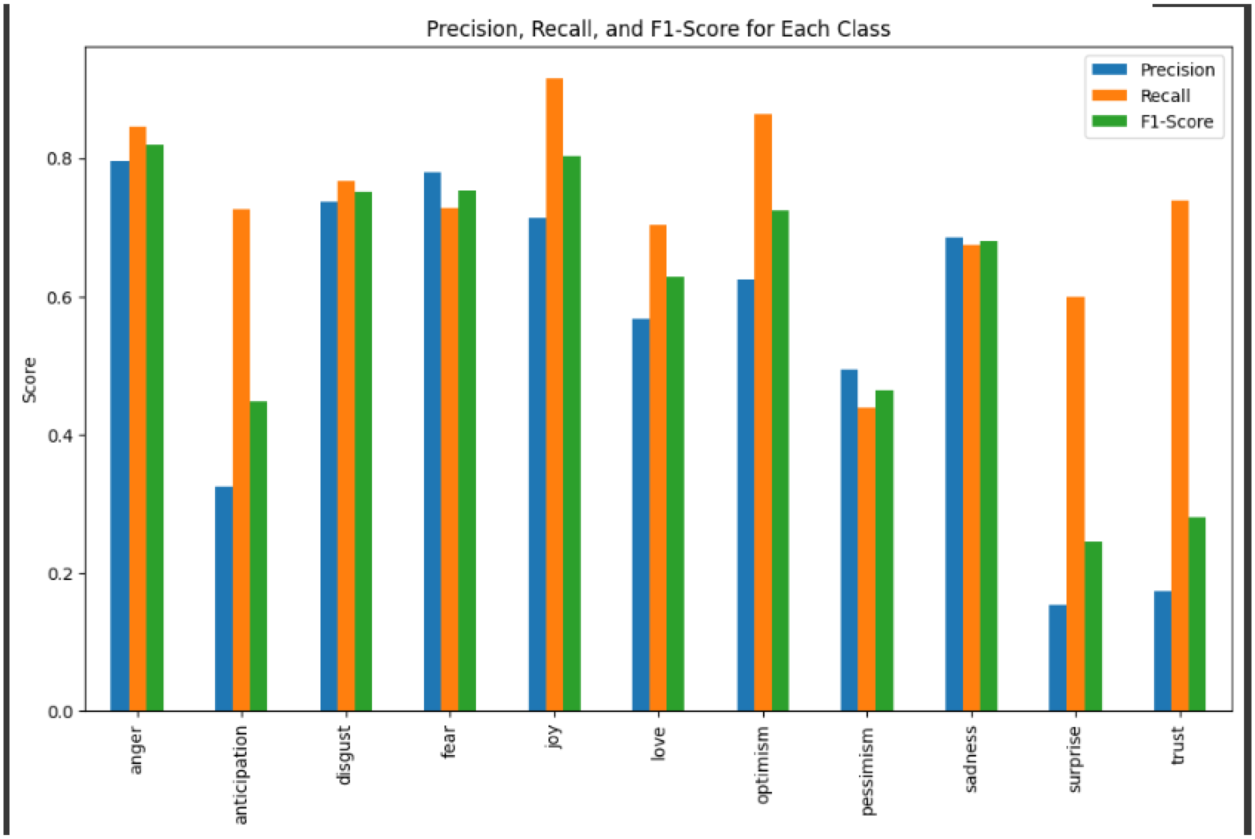
3. **Evaluation Result:**
   - **Loss:** 0.5104
   - **Performance Metrics:**
     - **F1 Micro:** 0.6674
     - **F1 Macro:** 0.5996
     - **Accuracy:** 0.1566
   - **Evaluation Runtime:** 35.1901 seconds
   - **Samples Per Second:** 43.904
   - **Steps Per Second:** 5.513

4. **Observations and Discussion:**
   - The model exhibits steady improvement in performance metrics during training.
   - Validation loss follows a fluctuating trend but generally decreases, indicating effective learning without overfitting.
   - Evaluation metrics indicate reasonable performance on unseen data, although there's room for improvement.

- Further analysis may be required to identify specific areas of improvement or potential bottlenecks in the model.

5. **Hyperparameter Selection:**
   - **Number of Training Epochs:** Increased to 4.
   - **Batch Size:** Reduced to 8 samples per training batch and evaluation batch.
   - **Gradient Accumulation Steps:** Set to 8, which means gradients are accumulated over 8 steps before updating the model's parameters.
   - **Weight Decay:** Maintained at 0.1 for L2 regularization to prevent overfitting.
   - **Learning Rate:** Set to 1e-4, indicating a low learning rate for stable training.
   - **Learning Rate Scheduler:** Linear scheduler without warmup steps.
   - **Optimizer:** AdamW optimizer from PyTorch.
   - **Maximum Gradient Norm:** Set to 1.0 to prevent exploding gradients.
   - **Evaluation Strategy:** Evaluation performed at regular step intervals during training.
   - **Model Checkpointing:** Model checkpoints saved at specified step intervals with a limit on the number of saved checkpoints.
   - **Metric for Best Model:** F1 macro score used to determine the best model.
   - **Logging:** Metrics and results logged to the Weights & Biases platform.
   - **Mixed Precision Training:** Enabled for BF16 format.

6. Confusion Matrix:



Conclusion:

The hyperparameters are adjusted to train the model over 4 epochs with a small batch size of 8, which might increase the training duration but could lead to better generalization. Gradient accumulation steps are increased to 8, which can help effectively utilize GPU memory and enable training with larger effective batch sizes. Weight decay and learning rate are set to conservative values to ensure stable training without overfitting. The AdamW optimizer is chosen for its effectiveness in training neural networks. Model evaluation and checkpointing strategies are implemented to monitor training progress and save the best-performing models. Mixed precision training with BF16 format is enabled to leverage the benefits of reduced memory usage and faster computation on compatible hardware.

High Level Overview

Model	Training Loss	Validation Loss	F1 Micro	F1 Macro	Accuracy	Evaluation Loss	Eval F1 Micro	Eval F1 Macro	Eval Accuracy
Gemma (LoRA)	0.2465	0.5656	0.6812	0.5820	0.1981	0.5268	0.6817	0.5911	0.1935
Gemma (iA3)	0.2465	0.5656	0.6812	0.5820	0.1981	0.5527	0.6777	0.5802	0.1838
Qwen (QLoRA)	0.0705	0.9911	0.6837	0.5797	0.2311	0.5104	0.6674	0.5996	0.1566

Key points:

- **Training Loss**: Gemma (iA3) and Qwen (QLoRA) have significantly lower training losses compared to Gemma (LoRA).
- **Validation Loss**: Gemma (iA3) has the lowest validation loss, indicating better generalization.
- **F1 Micro**: Gemma (LoRA) achieves the highest F1 Micro score, followed closely by Qwen (QLoRA).
- **F1 Macro**: Gemma (iA3) achieves the highest F1 Macro score, indicating better performance on average across all classes.
- **Accuracy**: Gemma (LoRA) has the highest accuracy.
- **Evaluation Loss**: Gemma (LoRA) has the lowest evaluation loss.
- **Eval F1 Micro & Macro**: Qwen (QLoRA) achieves the highest F1 Micro and F1 Macro scores during evaluation.
- **Eval Accuracy**: Gemma (LoRA) has the highest evaluation accuracy.

Overall, Gemma (LoRA) performs well in terms of training loss, accuracy, and evaluation metrics, while Gemma (iA3) shows better generalization with lower validation loss and higher F1 Macro score during evaluation. Qwen (QLoRA) also demonstrates competitive performance in terms of F1 scores during evaluation.

WandB Links

1. Gemma (LoRA)
2. Gemma (iA3)
3. Qwen (QLoRA)