**Homework 6: Report**

**Qwen2.5-7B Model Architecture and Decoder-Style Classification Setup**

The Qwen2.5-7B model is a decoder-only transformer architecture with approximately 7 billion parameters. It follows a autoregressive design, originally intended for next-token prediction tasks. The model consists of a stack of transformer decoder blocks, each composed of self-attention and feedforward layers. It uses rotary position embeddings (RoPE) to enhance positional awareness and supports multilingual tokenization through a SentencePiece-based tokenizer. The model outputs a sequence of hidden states, one for each input token.

To adapt this decoder-style model for multi-label emotion classification, a classification head is added on top of the model. The input text is tokenized and passed through the model, and the final hidden state corresponding to the first token (typically the beginning-of-sequence token) is used as a summary representation of the entire sequence. This vector is then passed through a linear classification layer that outputs a vector of logits, one for each target label (11 in total, representing different emotions).

Since the task involves multi-label classification (i.e., multiple labels can be active simultaneously), a sigmoid activation is applied independently to each logit. To handle label imbalance and improve robustness, the model is trained using focal loss with label-specific positive class weights. During inference, label-specific thresholds—optimized post-training to maximize F1 scores—are applied to the sigmoid probabilities to determine the final binary predictions for each emotion label.

**QLoRA: How It Works and How It Was Applied**

QLoRA (Quantized Low-Rank Adaptation) is a memory-efficient fine-tuning technique that enables large language models (LLMs) to be adapted using significantly reduced hardware resources. It combines two key components: 4-bit quantization and Low-Rank Adaptation (LoRA).

First, 4-bit quantization is applied to the pre-trained model weights using quantization formats such as NF4 (NormalFloat 4-bit). This drastically reduces memory usage while preserving model accuracy. Importantly, the quantized base model remains frozen during fine-tuning.

Second, LoRA injects trainable low-rank matrices into specific linear layers of the transformer architecture, such as the query, key, value, and output projections in the attention mechanism. These low-rank matrices are the only parameters updated during fine-tuning, making the process computationally efficient and parameter-light.

In this homework, QLoRA was used to fine-tune the Qwen2.5-7B model for a multi-label emotion classification task. The application involved the following steps:

1. The model was loaded with 4-bit quantization (load\_in\_4bit=True) using the bitsandbytes library to reduce memory usage.
2. The model was prepared for quantization-aware training using prepare\_model\_for\_kbit\_training().
3. A LoRA configuration was defined with a rank of 16 (r=16), scaling factor (lora\_alpha=32), dropout (lora\_dropout=0.1), and target modules set to ["q\_proj", "k\_proj", "v\_proj", "o\_proj"] to apply LoRA to the attention mechanism.
4. The LoRA adapters were applied to the quantized model using get\_peft\_model(), resulting in a lightweight trainable model where only the adapters were updated.
5. Fine-tuning was performed using a custom Trainer that incorporated Focal Loss and label-specific thresholding for handling multi-label imbalance, with only the LoRA parameters being trained.

This QLoRA-based approach enabled efficient fine-tuning of a large-scale 7B parameter model on limited hardware, without the need to update or store the full model weights.

**Compare Performance to Encoder-only Models**

I compared the performance of a decoder-only model (Qwen2.5-7B) with three encoder-only models: RoBERTa-base, DistilBERT-base-uncased, and ALBERT-base-v2.

The decoder-only Qwen2.5-7B model achieved a **best validation F1 score of 0.63**, which outperformed all encoder-only models. For comparison, RoBERTa-base achieved 0.59032, DistilBERT-base reached 0.56498, and ALBERT-base-v2 attained 0.53475. This makes Qwen2.5-7B the top-performing model in terms of raw accuracy and generalization.

Unlike the encoder-only models, which used a single classifier head for all emotions, the decoder model was trained with a custom trainer that evaluated and optimized thresholds for each individual emotion class. This fine-grained approach allowed it to better handle class imbalance and improve F1 scores across minority emotion categories. It also leveraged data augmentation specifically for underrepresented emotions, which helped boost performance further.

While encoder models like RoBERTa and DistilBERT were faster and more efficient, they lacked this level of granularity. Additionally, the decoder model used a focal loss function with class-wise weighting to better learn from hard examples—something that the encoder models did not implement.

However, it is important to note that the decoder model required significantly more computational resources, even with 4-bit quantization and gradient accumulation. The training process was longer and more memory-intensive compared to the encoder models, especially DistilBERT and ALBERT which are designed to be lightweight.

In conclusion, the decoder-only Qwen2.5-7B model demonstrated superior performance in terms of F1 score and robustness, thanks to advanced optimization techniques and label-specific fine-tuning. Despite its computational cost, it proved to be the most accurate model for this multi-label emotion classification task.

**Discuss Any Challenges Faced (e.g., Model Compatibility, Resource Usage)**

Several challenges were encountered while working with the decoder-only Qwen2.5-7B model for multi-label emotion classification, particularly due to the model's size, architecture, and tooling requirements.

**1. Resource Usage:** The most significant challenge was the high computational and memory demand of the Qwen2.5-7B model. Despite using 4-bit quantization via bitsandbytes and applying parameter-efficient fine-tuning (QLoRA), the model still required a GPU with substantial VRAM to load and train. Batch sizes had to be kept very small (e.g., batch size 2 with gradient accumulation) to avoid out-of-memory errors. Training time was also considerably longer compared to encoder-only models like DistilBERT or ALBERT.

**2. Model Compatibility:** Another challenge was ensuring compatibility with Hugging Face tools and the Qwen model’s custom architecture. Special care was needed to:

* Set trust\_remote\_code=True when loading both the tokenizer and model.
* Replace the default padding token with the end-of-sequence (eos\_token), since decoder models do not natively handle padding in the same way as encoders.
* Adapt the Trainer class to work with decoder outputs, especially since logits sometimes had an extra dimension ([batch\_size, 1, num\_labels]) that required squeezing.

**3. Custom Loss and Thresholding:** Implementing and integrating a custom FocalLoss with class weights was essential to deal with label imbalance but required careful debugging. Additionally, threshold optimization for each emotion label introduced extra complexity, as it had to be manually run after training and dynamically updated in the evaluation pipeline.

In summary, while the Qwen2.5-7B decoder model delivered strong performance, it demanded advanced hardware, careful integration, and several custom modifications. These challenges made the development process more complex compared to typical encoder-only models.

Wandb Link: <https://wandb.ai/wadekarritin6-onpoint-insights/qwen2-emotion-multilabel-improved/runs/9y33hzm6?nw=nwuserwadekarritin6>

If this link is not visible, I am attaching the screenshots:  




