Task 3: Training Considerations

Discuss the implications and advantages of each scenario and explain your rationale as to how the model should be trained given the following:

1. If the entire network should be frozen.
2. If only the transformer backbone should be frozen.
3. If only one of the task-specific heads (either for Task A or Task B) should be frozen.

Answer:

1. **Freezing the Entire Network**: If the entire network is frozen, no learning occurs as the model's weights remain fixed. This approach is typically used only when the model is already well-suited to the target task and does not need further adaptation.
2. **Freezing Only the Transformer Backbone**: Freezing just the transformer backbone allows the model to leverage pre-trained abstract features while training the task-specific heads separately. This is advantageous because the backbone retains valuable general features, while the heads are fine-tuned to learn high-level, task-specific representations. This approach is effective in multi-task settings, where each task-specific head can specialize without altering the shared features in the backbone.
3. **Freezing One Task-Specific Head (Task A or Task B)**: Freezing only one of the task-specific heads (e.g., for Task A) means the transformer backbone and the other head (Task B) are still trainable. This can lead to bias toward the unfrozen head, as the shared backbone continues to adjust in alignment with the non-frozen head's objective. This setup may be useful in cases where one task is already optimized, and the focus is on fine-tuning the remaining task.

Consider a scenario where transfer learning can be beneficial. Explain how you would approach the transfer learning process, including:

1. The choice of a pre-trained model.
2. The layers you would freeze/unfreeze.
3. The rationale behind these choices.

Answer:

In modern machine learning, transfer learning often yields significant advantages by leveraging pre-trained models from similar or even different domains. Here's an outline of how to approach transfer learning in this context:

1. **Choosing a Pre-Trained Model**: Select a model pre-trained on a dataset similar in structure or features to the target dataset. For instance, if the task involves visual recognition, a CNN pre-trained on a large-scale image dataset like ImageNet would be a good starting point, as it contains rich representations of visual features.

2. **Layer Freezing**: Freeze the initial layers (or backbone) to retain the model's learned low- and mid-level features while unfreezing and fine-tuning the task-specific layers. This allows the model to adapt to the new task without retraining from scratch, leveraging the previously learned representations for faster convergence.
3. **Rationale**: Starting with a pre-trained model often improves convergence speed and model performance by leveraging the existing knowledge in the weights. Fine-tuning specific layers while freezing others allows the model to specialize in the new task, achieving a balance between retaining general features and adapting to new, task-specific requirements.

Task 4:

Learning Rates for our model:

- **Embedding Layer**: Set a learning rate of 1e-3.
    - This is a low-level layer responsible for capturing foundational features, so a finer learning rate helps prevent drastic changes that could drift away from good initial representations.
- **Transformer Layer**: Use a learning rate of 5e-3.
    - As a middle layer, this part of the model focuses on generalizing patterns from the dataset. A slightly higher learning rate allows it to adapt faster, helping it develop better generalizations over time.
- **Sentence Embedding and Sentiment Classification Layers**: Set a learning rate of 1e-2.
    - These layers capture task-specific, high-level features crucial for task understanding. A higher learning rate here allows faster learning of these details, which is important to avoid long convergence times.

Benefits of Layer-Wise Learning Rates
Applying layer-wise learning rates in deep networks like transformers has a few key benefits:

1. Maintaining Pre-Trained Knowledge with lower learning rates for the initial layers helps retain the core features learned during pre-training. This helpful to maintain general language features, without relearning it.
2. Quick Task-Specific Fine-Tuning, with higher learning rate, helps the model adjust more quickly to task-specific details. This can save training time and also make the model more efficient at tackling nuanced differences in the tasks.
3. Better Regularization: Having different learning rates for each layer is like a form of regularization. It keeps the model from overfitting to one task too quickly, as each layer is updating at the rate that makes the most sense given its purpose.

Impact of Multi-Task Setting
In a multi-task model, layer-wise learning rates are especially effective. They allow shared backbone layers to stay stable with smaller updates, benefiting all tasks, while task-specific heads can adapt with larger updates. This helps the model perform better on each task without

negatively impacting the others, which is crucial for balancing performance across multiple objectives.