E0270: MACHINE LEARNING

Assignment #2

Author
Pritam Trilochan Gouda
CSA, IISc
April 27, 2024

Acknowledgments

I would like to express my sincere gratitude to **Professor Ambedkar Dukkipati** for their invaluable guidance and teaching throughout the duration of the course. Their expertise and support have been instrumental in shaping my understanding of machine learning concepts. I am also grateful to my TA's for their assistance, feedback, and encouragement during the completion of this assignment. Their insights and assistance have greatly contributed to my learning experience and the successful completion of the tasks.

Contents

Problem 0: Text Generation with GPT-2

As part of the initial exploration of the GPT-2 model's capabilities, a text generation task was performed to understand the model's ability to continue a given prompt. The prompt provided to the model was: ”Hello, I am Pritam. Today when I was completing my assignment, I heard a loud noise.”

Generated Text Instance

The model was able to take this prompt and generate a continuation that not only maintained the context but also introduced a creative narrative. Below is the generated text:

> Hello, I am Pritam. Today when I was completing my assignment, I heard a loud noise. I raised my hand as if to allude to it but I couldn't answer…
>
> A low cry shot through the air. ”What the…” an assistant apparently whispered (”Is it puppy time?”). Oh Gods that was genius! A year ago we had thought this was just a puppy shower. Now it is..I still don't know.
>
> Thankfully my gun went off, no shots, no shots counting as corpses. We shuffled through the dumpster until we found Quiet '...

This instance of text generation exemplifies the model's capacity for creating a coherent and engaging narrative based on a minimal prompt. It demonstrates a grasp of story development, from a mundane activity to an unexpected event, all the while maintaining a consistent tone.

Analysis

The generated text illustrates the model's use of creative language and situational development, showing its potential for diverse applications such as story writing or idea generation. The narrative's unexpected turn towards a more dramatic scene, coupled with a touch of humor, reflects the model's ability to fuse various elements into a cohesive whole.

Moreover, the model's response highlights its understanding of context and progression, crafting a narrative that builds upon the prompt in a logical yet imaginative manner. The text leaves the reader with curiosity and a sense of suspense, showcasing the model's knack for engaging storytelling.

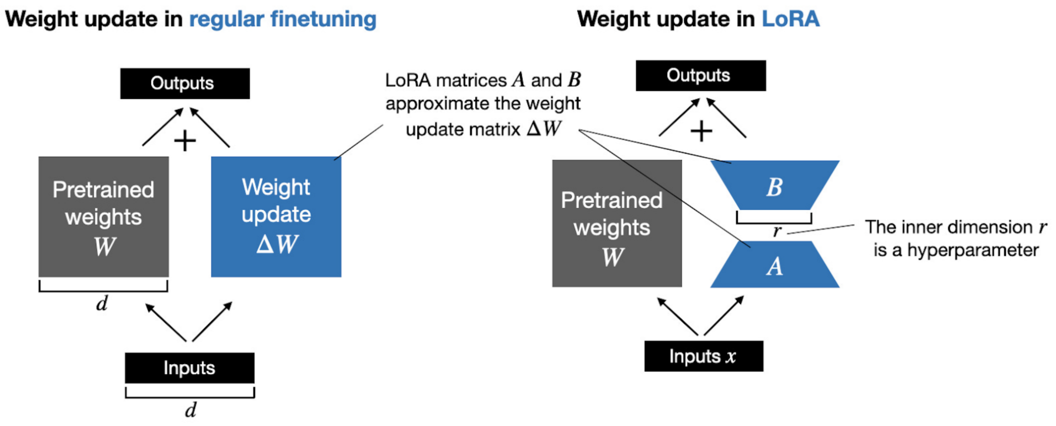
Problem 1: Low Rank Adaptation (LoRA)

Introduction

This report presents the application of Low Rank Adaptation (LoRA), a Parameter-Efficient Fine-tuning (PEFT) technique, on the GPT-2 model for the task of linguistic acceptability classification using the CoLA dataset. LoRA enables selective updating of model parameters, resulting in a compact and efficient adaptation with reduced computational overhead.

Implementation

The LoRA mechanism was integrated by implementing the `LoRALinear` class in the `model.py` file. This class introduced two matrices with a low-rank bottleneck, providing a decomposition that represents the adjustments to the pre-trained model. The injection of LoRA matrices was successfully completed for the self-attention and feedforward layers, after which the original dense layers' gradients were disabled to maintain the pre-trained weights. The tokenizer and positional embeddings were also frozen to limit the scope of training to the LoRA parameters alone.



Weight update in regular finetuning
Outputs
+
Pretrained weights W
Weight update ΔW
d
Inputs
d
Weight update in LoRA
LoRA matrices A and B approximate the weight update matrix ΔW
Outputs
+
Pretrained weights W
B
r
A
The inner dimension r is a hyperparameter
Inputs x

Figure 1: Weight update in LoRA

Training and Hyperparameter Tuning

The fine-tuning process employed the Adam optimizer, with an initial learning rate of 0.001. Experiments with the number of epochs indicated that extending training from 3 to 5 epochs further improved performance. The fine-tuning was finalized with a 5-epoch run, striking a balance between computational efficiency and model accuracy.

Results

The final fine-tuned model exhibited training and validation accuracies of 70.47% and 69.26% respectively after 5 epochs. An intermediate checkpoint at 3 epochs yielded training and validation accuracies of 70.66% and 69.45%. While the accuracy marginally improved from 3 to 5 epochs, it demonstrated diminishing returns, suggesting that the model had nearly converged.

Model Performance Analysis

The training and validation accuracy and loss plots highlight the model's performance over epochs. The training accuracy showed a consistent upward trend, while the validation accuracy peaked at the third epoch before declining slightly, indicating the potential onset of overfitting. The training loss decreased steadily across epochs, whereas the validation loss showed a minor increase after the third epoch. These observations emphasize the necessity of a vigilant hyperparameter tuning strategy to mitigate overfitting and enhance generalization.

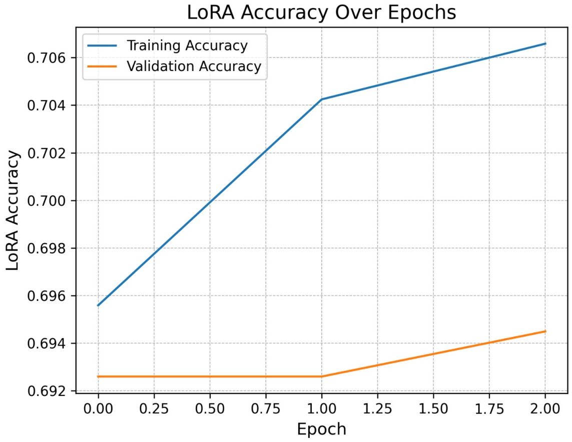
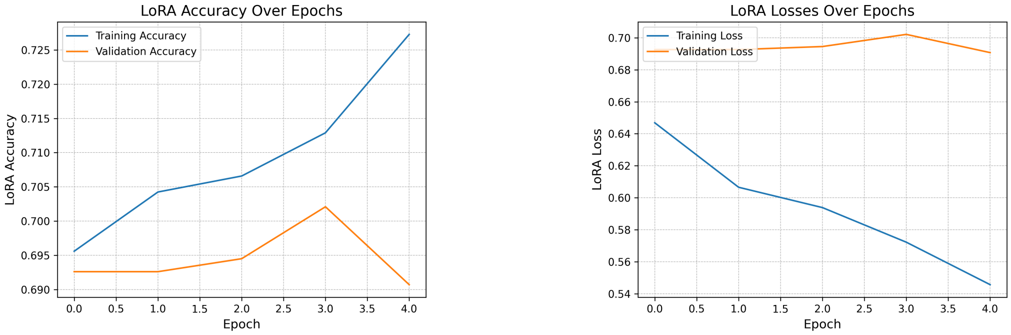


Figure 2: LoRA Training and Validation Accuracy Over 3 Epochs



(a) LoRA Accuracy Over Epochs

(b) LoRA Losses Over Epochs

Figure 3: Training and Validation metrics for LoRA

Conclusion

LoRA has proven to be a viable method for parameter-efficient model adaptation, offering a substantial reduction in the trainable parameter count without significantly compromising model performance. The fine-tuning resulted in a lightweight model that can be rapidly adapted to new tasks, facilitating efficient deployment in resource-constrained environments.

Problem 2: Knowledge Distillation

Knowledge Distillation (KD) aims to transfer the knowledge from a larger teacher model to a smaller student model. In this case, the fine-tuned GPT-2 model served as the teacher, and a newly created Recurrent Neural Network (RNN) model acted as the student.

Implementation of DistilRNN

A custom RNN architecture was defined in the `DistilRNN` class within the `model.py` file. This class was tailored to capture the essence of the linguistic patterns learned by the GPT-2 model, albeit in a more parameter-efficient manner.

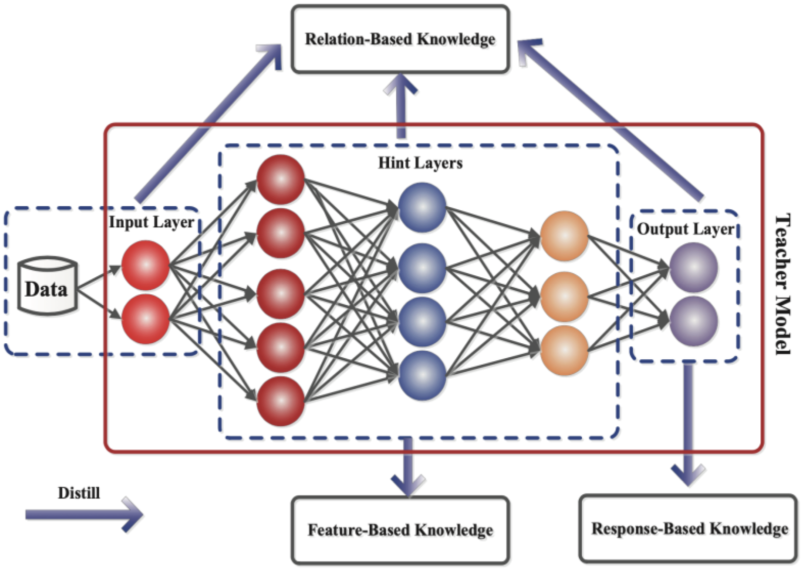


Figure 4: The different kinds of knowledge in a teacher model

Training Procedure

The RNN was trained through knowledge distillation over several epochs, using a combination of the Kullback-Leibler divergence loss for distillation and the Cross-Entropy Loss for the actual classification. This dual loss mechanism guided the student model to emulate the output distribution of the teacher while also performing the classification task effectively.

Hyperparameter Tuning and Results

Throughout the tuning process, it became evident that the model's performance saturated after 3 epochs. The final model was trained for 5 epochs, resulting in a slight improvement in training accuracy but no substantial gain in validation accuracy, suggesting early convergence of the student model. This observation prompted a halt in further training to prevent overfitting and to conserve computational resources.

Model Performance Analysis

The following plots illustrate the loss and accuracy of the student model over the training epochs:

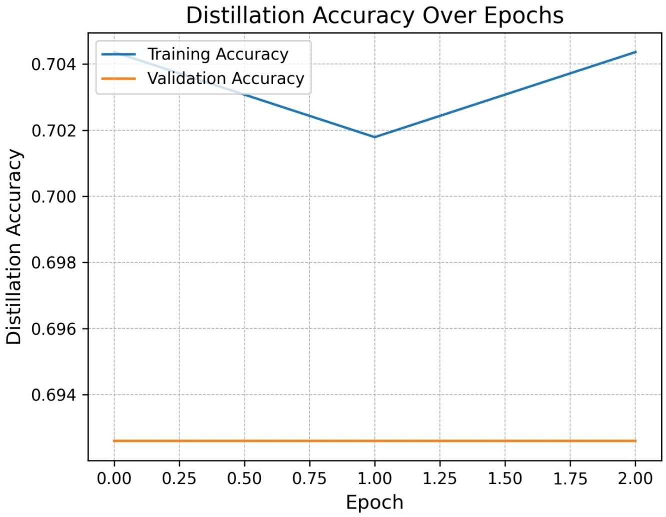


Figure 5: Distillation Training and Validation Accuracy Over 3 Epochs

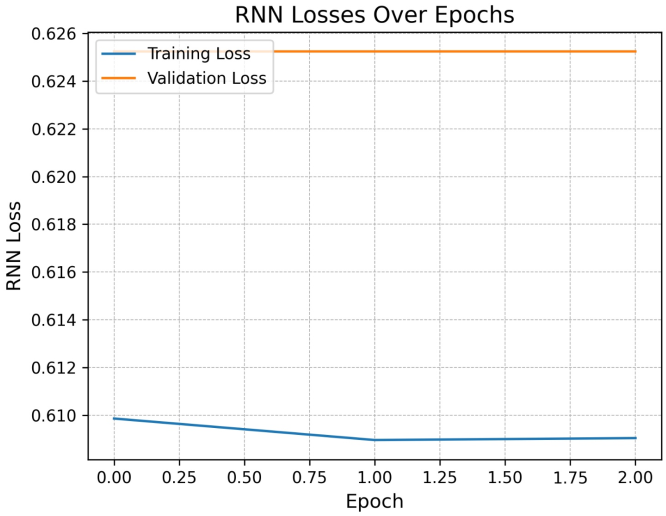
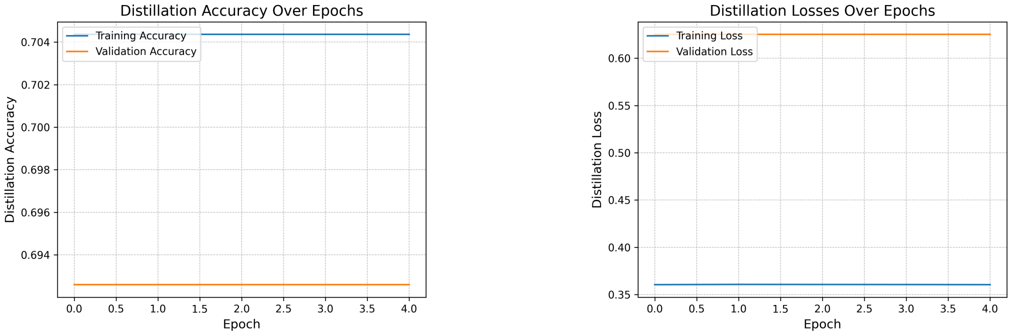


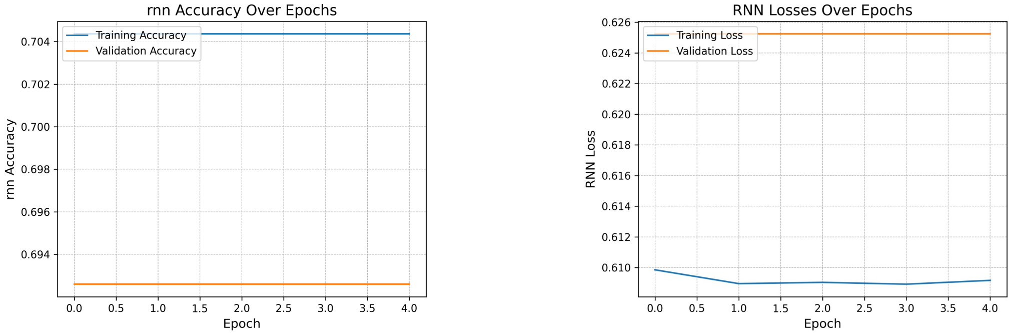
Figure 6: RNN Training and Validation Loss Over 3 Epochs



(a) Distillation Accuracy

(b) Distillation Loss

Figure 7: Training and Validation metrics for the student model during Distillation



(a) RNN Accuracy

(b) RNN Loss

Figure 8: Training and Validation metrics for the RNN model trained from scratch

As shown in Figures 7a and 7b, the student model's training loss decreased and plateaued, indicating a good fit to the training data. However, the validation loss remained relatively unchanged, and the accuracy on the validation set indicated that further training epochs did not yield improvements.

Comparatively, the RNN model trained from scratch, as depicted in Figures 8a and 8b, maintained stable training and validation metrics throughout the epochs. The consistency between the training and validation lines suggests that the RNN model was not overfitting, albeit the improvements in accuracy were marginal.

Conclusion

Knowledge Distillation was successfully applied to transfer the knowledge from a fine-tuned GPT-2 model to an RNN. The student model achieved similar validation performance compared to the teacher model, thereby confirming the effectiveness of the distillation process. This approach serves as a promising direction for deploying competent models in constrained environments without significant loss in performance.

References

- **Practical Tips for Finetuning LLMs Using LoRA** https://magazine.sebastianraschka.com
- **Knowledge Distillation: Principles, Algorithms, Applications** https://neptune.ai/blog/knowledge-distillation
- **Pretraining a 124-M Parameter GPT-2 Language Model** https://wandb.ai/bkkaggle