ANLP - Assignment - 3

Saketh Reddy Vemula | 2022114014 | ANLP Assignment-3 Report

8.1 Theory Questions

1. ***Concept of Soft Prompts**: How does the introduction of "soft prompts" address the limitations of discrete text prompts in large language models? Why might soft prompts be considered a more flexible and efficient approach for task-specific conditioning?*

   Discrete text prompts suffers following limitations, and they are addressed in `soft-prompts` as mentioned:

   1. Human-readable format prompts can `limit effectiveness`. Each task requires careful design, leading to the creation of unique prompts or even seperate models for different models. Soft prompts doesn't contain any human-readable constraints. Soft prompts consist of `learnable tensors` that can be optimized through backpropagation. This allows for fine-tuning without altering the core model parameters, `enhancing efficiency` by focusing only on the prompt embedding.
   2. Discrete text prompts may not contain all the information required for a specific task. Soft prompts can encapsulate information from large datasets, allowing them to leverage insights from thousands or millions of examples, which is something discrete prompts struggle with due to their inherent limitations. This allows for `Condensed Information`.
   3. `Fixed vocabulary` can limit the expressiveness and adaptability of Discrete text prompts. Soft prompts instead learn tensors that can are not even present in the vocabulary of the model.
   4. Discrete text prompts require `careful expert knowledge`. We require to test various prompts before deciding on the best performing prompt. While discrete prompts can be interpreted easily, they may not always yield optimal performance across varying tasks.

   Soft-prompts are more flexible and efficient for task-specific conditioning.

   1. **Adapting Across Tasks**:
      Soft prompts are designed as
      `learnable vectors` that can be fine-tuned for various tasks without the need for extensive modifications to the underlying model. This allows a single model to handle multiple tasks efficiently, enabling `quick transitions` between different applications without requiring unique prompts for each task.
   2. **Reduced need of manual design**:
      Soft prompts can be
      `optimized automatically during training`. This reduces the time and effort associated with prompt engineering, allowing for more streamlined model deployment.
   3. **Dynamic Modification**:
      Soft prompts can be easily adjusted or modified in response to different requirements or contexts. This flexibility is crucial in scenarios where task specifications may change frequently or where
      `rapid adaptation` is necessary.

4. **Continuous Learning**:
   Since soft prompts are not constrained by human-readable formats, they can `encapsulate complex relationships` and patterns learned from large datasets. This enables them to perform well across diverse tasks without being limited by the explicit instructions that characterize discrete prompts

5. **Minimal Parameter Tuning**:
   Soft prompts involve tuning a small number of learnable parameters (often just a few hundred) while keeping the main model parameters frozen. This contrasts with traditional fine tuning, which requires adjusting millions of parameters, making soft prompting significantly less resource-intensive.

6. **Multi-Task Capability**:
   Soft prompts facilitate mixed-task inference by allowing different prompts to be used with a single frozen model. This capability enables efficient batch processing of inputs from various tasks without the overhead of deploying separate models for each task.

2. ***Scaling and Efficiency in Prompt Tuning****: How does the efficiency of prompt tuning relate to the scale of the language model? Discuss the implications of this relationship for future developments in large-scale language models and their adaptability to specific tasks.*

   a. **Efficiency of Prompt Tuning**:

   - Scales well with larger models, as it fine-tunes only a small set of parameters (prompt tokens), keeping the majority of the model fixed.
   - Reduces the computational cost compared to full model fine-tuning, especially on large models.

   b. **Implications for Large Models**:

   - **Adaptability**: Makes large models more adaptable to specific tasks without retraining the whole model.
   - **Resource Savings**: Conserves computational resources, making fine-tuning feasible on large-scale models even with limited resources.
   - **Scalability**: As language models grow, prompt tuning remains efficient, allowing rapid adaptation across multiple domains with minimal overhead.

   c. **Future Developments**:

   - Likely increase in adoption of prompt-based approaches for task specialization.
   - Facilitates deployment of large models in low-resource environments.
   - May encourage research into more efficient and dynamic prompt tuning methods to improve task performance with minimal adjustments.

3. ***Understanding LoRA****: What are the key principles behind Low-Rank Adaptation (LoRA) in fine-tuning large language models? How does LoRA improve upon traditional fine-tuning methods regarding efficiency and performance?*

   Key principles of **Low-Rank Adaptation (LoRA)** in fine-tuning large language models:

   - **Low-Rank Decomposition**: LoRA constrains weight updates during fine-tuning to low-rank matrices. Instead of updating all model parameters, LoRA adds low-rank matrices to specific weights, reducing the number of trainable parameters. Instead of updating the entire weight matrix, LoRA introduces low-rank matrices `A` (random Gaussian) and `B` (zeros) to approximate weight updates.

- **Parameter Efficiency**: By freezing the pre-trained model weights and only learning a small number of additional parameters (the low-rank matrices), LoRA reduces the computational and memory overhead.
- **Delta Weight (`delta_W`) Initialization**: Set to zero initially, ensuring minimal changes to the pre-trained model.
- **Rank Constraint (`r`)**: The rank `r` of the weight matrices is set to control the amount of learned parameters, making the model parameter-efficient.

Improvements over Traditional Fine-Tuning methods regarding efficiency and performance:

- **Parameter Efficiency**: LoRA fine-tunes only the low-rank matrices (`A` and `B`), significantly reducing the number of trainable parameters compared to full model fine-tuning.
- **Faster Training**: Since fewer parameters are updated, LoRA speeds up the training process without compromising much performance.
- **Memory Efficient**: Requires less memory, allowing fine-tuning on large models with limited hardware resources.
- **No Additional Inference Cost**: Unlike some other fine-tuning methods, LoRA does not introduce extra inference costs when switching between tasks, as the low-rank updates can be applied efficiently.
- **Task Modularity**: LoRA allows easy swapping of the learned task-specific parameters (i.e., `A` and `B`), making it flexible for multi-task setups without retraining the base model.

4. ***Theoretical Implications of LoRA****: Discuss the theoretical implications of introducing low-rank adaptations to the parameter space of large language models. How does this affect the expressiveness and generalization capabilities of the model compared to standard fine-tuning?*

1. Parameter Efficiency

- **Standard Fine-Tuning**: Involves adjusting all parameters of the model.
- **LoRA**: Introduces low-rank matrices `A` and `B` such that the weight updates can be expressed as `Delta W` = `A` `B`
- **Implication**: This significantly reduces the number of parameters being updated, leading to faster convergence and lower computational costs.

2. Expressiveness

- **Rank Constraint**: The low-rank approximation limits the expressiveness of the model in capturing complex patterns compared to full-rank updates.
- **Expressive Subspace**: LoRA can be thought of as projecting the parameter space into a lower-dimensional subspace, which might help focus on the most relevant features during training.

3. Generalization Capabilities

- **Regularization Effect**: By restricting the parameter updates to a low-rank space, LoRA acts as a form of regularization, which can improve generalization to unseen data.
- **Reduced Overfitting**: With fewer parameters being tuned, the risk of overfitting on small datasets is minimized compared to standard fine-tuning.

- **Statistical Bias-Variance Trade-off**: Low-rank adaptations can decrease the variance of the model while maintaining bias, leading to better generalization.

4. Adaptation to Tasks

- **Task-Specific Fine-Tuning**: LoRA allows models to adapt to specific tasks without altering the entire parameter space, maintaining the core knowledge while enabling specialized learning.
- **Multi-Task Learning**: Enables better transfer learning across tasks since the low-rank updates can be shared or adjusted without overhauling the model.

5. Gradient Descent Dynamics

- **Learning Dynamics**: The optimization landscape is altered; low-rank adaptations can lead to smoother optimization paths and can mitigate issues such as saddle points due to the reduced parameter space.
- **Theoretical Convergence**: Low-rank adaptations can potentially lead to faster convergence rates in the training process, theoretically underpinned by the structure of the loss landscape.

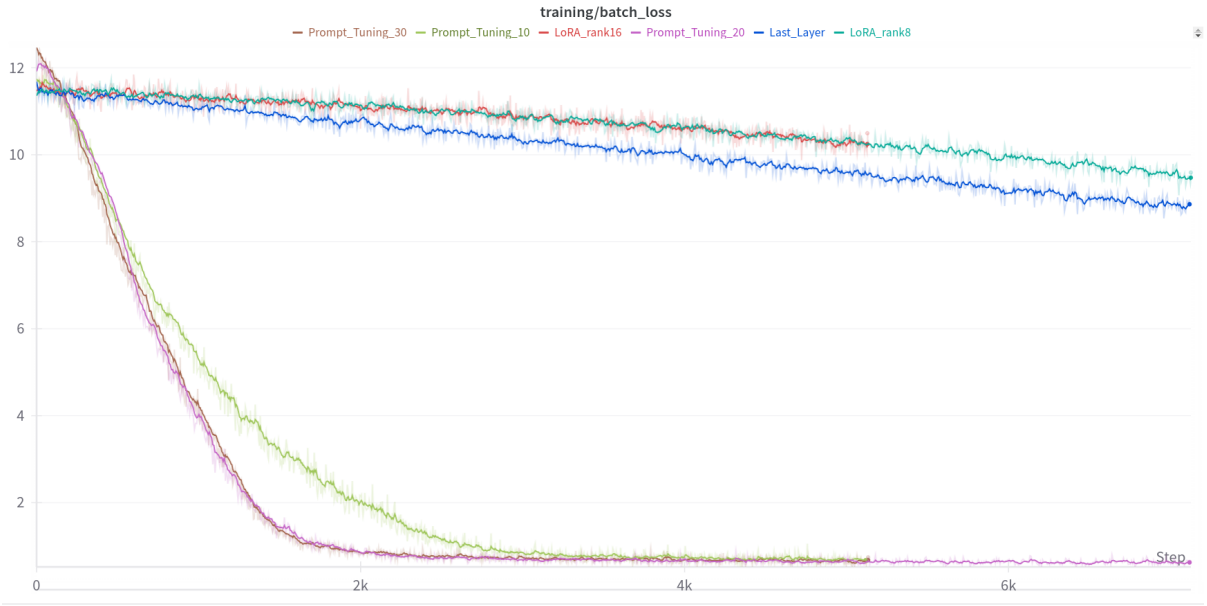
6. Computational Complexity

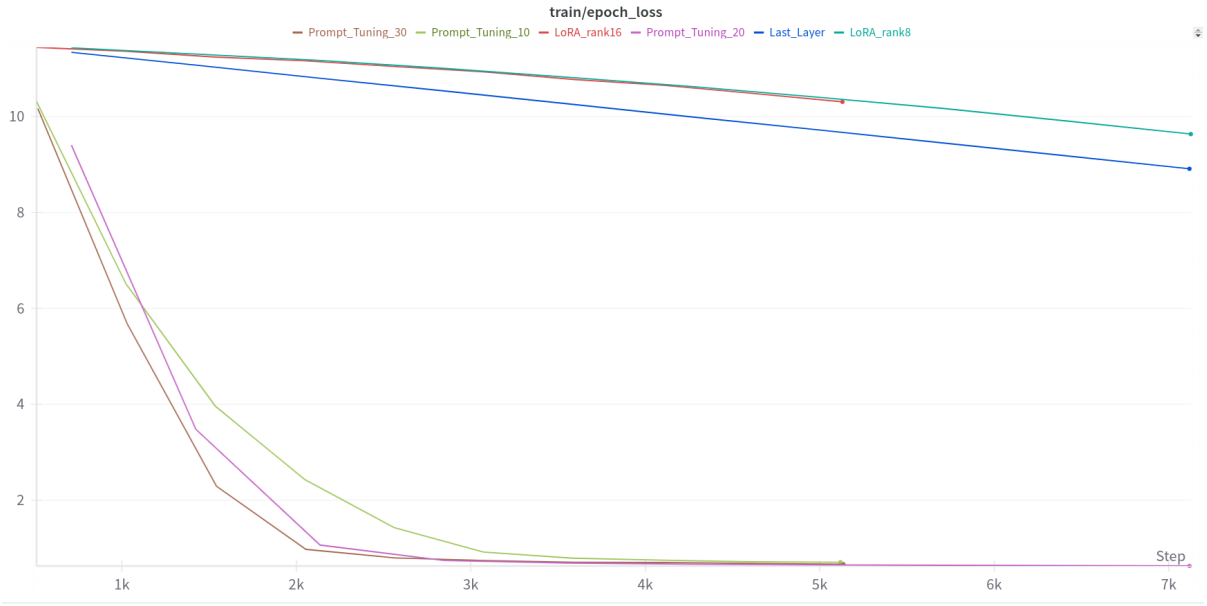
- **Reduced Complexity**: The complexity of computing gradients and performing updates is reduced, allowing for larger models to be effectively trained on smaller datasets or with fewer resources.

Therefore, ntroducing low-rank adaptations to LLMs allows for a more efficient, focused approach to model fine-tuning, enhancing generalization while maintaining expressiveness in specific task contexts. This trade-off is mathematically supported by the constraints and properties of low-rank matrices, enabling robust learning without the pitfalls associated with full parameter optimization.

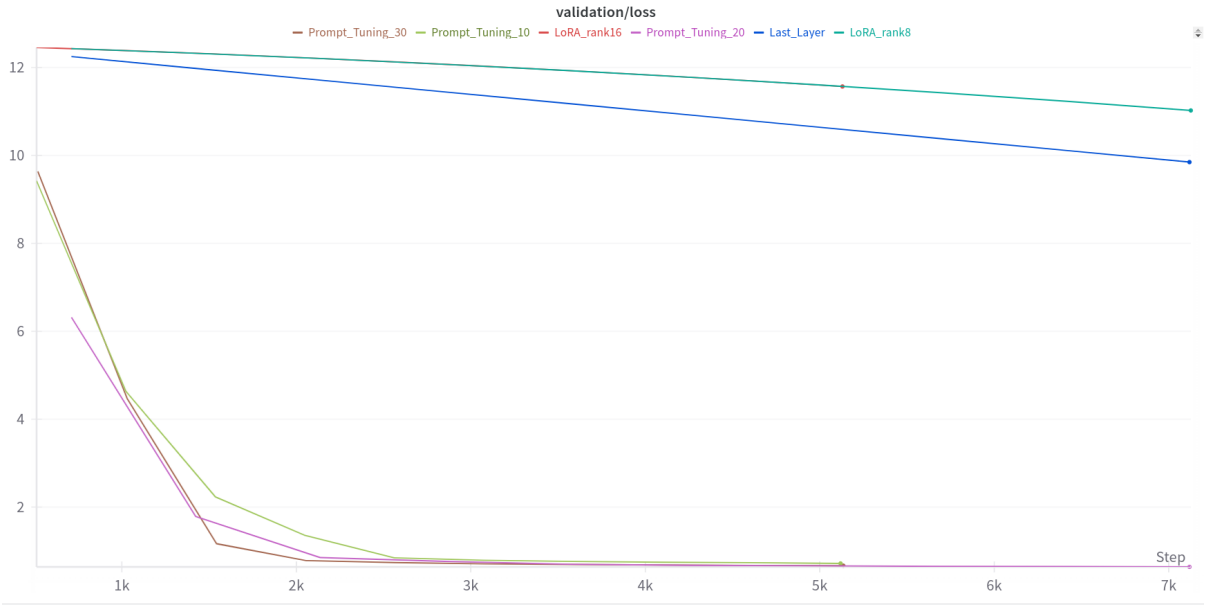
Prompt-Tuning vs Last Layer Fine Tuning vs LoRA

Training Loss:





Evaluation Loss:



Observation:

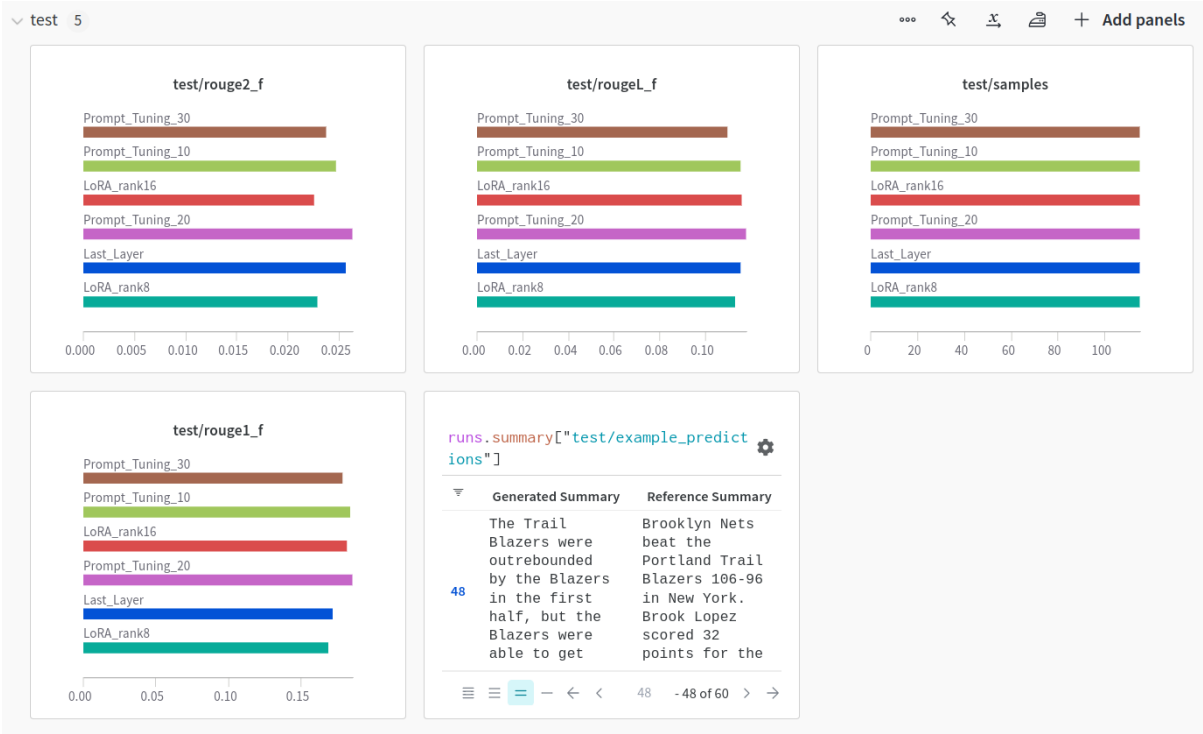
1. **Prompt Tuning (Ranks 10, 20, 30)**:
    - These methods display a `rapid decrease` in validation loss initially, especially within the first 1k steps, indicating `efficient learning`.
    - Among the prompt tuning methods, Prompt_Tuning_30 (with 30 soft prompts) achieves the lowest validation loss, followed by Prompt_Tuning_10 and Prompt_Tuning_20. This suggests that increasing the number of prompts may contribute positively to the model's learning.
2. **Last Layer Fine-Tuning**:
    - This method shows a `more gradual decrease` in validation loss compared to prompt tuning but remains consistent.
    - Although it doesn't reach the lowest loss values seen in prompt tuning, its stability across more steps suggests a steady but slower learning curve, which might imply `better generalization with further training steps`.
3. **LoRA Fine-Tuning (Rank 8 and Rank 16)**:
    - LoRA_rank8 and LoRA_rank16 perform similarly to last layer fine tuning, but with slightly higher validation loss than Traditional Fine Tuning
    - Both LoRA with rank8 and rank16 performs almost equally. Despite having only 0.36% trainable parameters compared to traditional fine tuning, LoRA gives a efficient solution,

Key points that can be made:

- **Prompt Tuning** methods are the fastest to converge, with Prompt_Tuning_30 achieving the lowest validation loss. This makes it potentially the best option among the three for this task, balancing speed and effectiveness.
- **Last Layer Fine-Tuning** is more gradual in its decrease, suggesting it may generalize well if given more training. However, it doesn't reach the low validation loss levels achieved by prompt tuning.

- **LoRA Fine-Tuning** achieve almost similar performance wrt Last Layer Fine Tuning.

Metrics:



Performance Summary

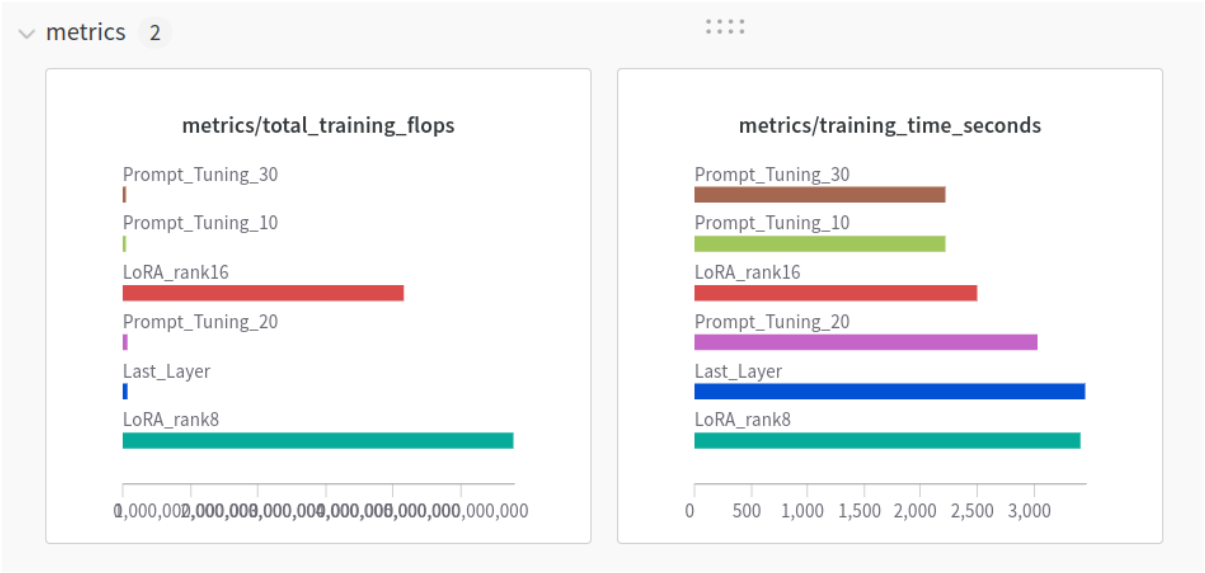
1. **Prompt Tuning (10, 20, 30 soft prompts)**:
   - **ROUGE-1**: Ranges from around 0.17841 to 0.18565.
   - **ROUGE-2**: Scores are relatively lower, between 0.023 and 0.026.
   - **ROUGE-L**: Scores range from around 0.109 to 0.1179
   - **Trend**: Among prompt tuning options, using 20 soft prompts yields the highest scores across all ROUGE metrics, suggests that not much not less number of prompts works well. 10 might be too less, and 30 migth lead to shorter actual input. Since, the more the number of prompt tokens lesser is the space for input.
2. **Last Layer Fine-Tuning**:
   - **ROUGE-1**: Around 0.1719, which is competitive with prompt tuning using soft prompts.
   - **ROUGE-2**: Slightly higher than prompt tuning with 10, 30 prompts, around 0.0256.
   - **ROUGE-L**: Approximately 0.11549, on par with the best prompt tuning configuration.
   - **Trend**: Last layer fine-tuning seems to achieve comparable results to the best prompt tuning setup, showing that adjusting only the last layer is effective for summarization on this small dataset.
3. **LoRA Fine-Tuning (Rank 8 and 16)**:

- **ROUGE-1**: Scores range between 0.16 and 0.18.
- **ROUGE-2**: Ranges from 0.0225 to 0.0228.
- **ROUGE-L**: Around 0.1157 to 0.1159.
- **Trend**: LoRA with rank 16 performs slightly better than rank 8 across all ROUGE metrics, though it does not outperform the best prompt tuning or last layer fine-tuning configurations.

Conclusion

- **Best Performance**: The highest ROUGE scores are achieved by either using 20 soft prompts in prompt tuning or performing last layer fine-tuning, with each method yielding similar results.
- **LoRA Limitations**: LoRA tuning performs well but does not reach the top scores achieved by prompt tuning with 20 soft prompts or last layer fine-tuning, suggesting it may not be as effective on this task with these configurations.

Training Time and Compute:



1. Training Time

- **Prompt Tuning**:
  - *Prompt Tuning 10*: Shortest training time.
  - *Prompt Tuning 20 and 30*: Moderate and relatively high training time respectively.
- **LoRA**:
  - *Rank 8*: Longest training time, taking over 3000 seconds.
  - *Rank 16*: Significantly less time than rank 8 but still higher than prompt tuning (expect 20 soft prompts).
- **Last Layer**: Moderate training time, similar to Prompt Tuning 20 but longer than Prompt Tuning 10.

- Prompt tuning generally requires less training time, especially with fewer prompts. LoRA with rank 8 is the most computationally demanding in terms of training time.

2. Compute (Total Training FLOPs)

- **Prompt Tuning**: Lowest compute requirements across all variations (Prompt Tuning 10, 20, and 30), as indicated by the minimal bar lengths.
- **LoRA**:
  - *Rank 8*: Highest compute requirement, as reflected by the significantly larger bar in FLOPs.
  - *Rank 16*: Higher than prompt tuning but considerably less than rank 8.
- **Last Layer**: Low compute requirements, comparable to prompt tuning and significantly less than LoRA rank 8.

Summary: Prompt tuning and last layer fine-tuning require much less compute than LoRA, especially at rank 8.

3. Number of Added Trainable Parameters

- **Last Layer**:
  - Total Parameters: 124,439,808
  - Trainable Parameters: 45,686,784
  - **Percentage of Trainable Parameters**: 36.71%
- **LoRA**:
  - *Rank 8*:
    - Trainable Parameters: 1,179,648
    - Total Parameters: 125,619,456
    - **Percentage of Trainable Parameters**: 0.94%
  - *Rank 16*:
    - Trainable Parameters: 2,359,296
    - Total Parameters: 126,799,104
    - **Percentage of Trainable Parameters**: 1.86%
- **Prompt Tuning**: Trainable parameters depend on the number of soft prompts but remain lower than both Last Layer and LoRA in terms of percentage of the model.

Summary:

- **Last Layer Fine-Tuning** has the highest percentage of trainable parameters, which may contribute to its relatively high effectiveness in tuning.
- **LoRA** introduces far fewer trainable parameters, especially at rank 8, which could explain its lower performance on ROUGE scores despite increased compute and training time.

Overall Comparison

- **Efficiency**: Prompt tuning is the most efficient in terms of training time and compute, particularly with a small number of soft prompts (e.g., Prompt Tuning 10).
- **Performance vs. Efficiency**: Last layer fine-tuning strikes a balance by having a high number of trainable parameters, moderate compute requirements, and competitive performance on ROUGE scores.

Generation Examples:

1. Last Layer:

```
Generated: The ACLU of Michigan filed the lawsuit on behalf of Shebshana Hebs
Reference: The federal government will give Shoshana Hebshi $40,000 as compen
--------------------------------------------------------------------------------
Generated: In the blog, Ms Bailey said she had been working for Mr Key for a
Reference: Amanda Bailey, 26, says she doesn't regret going public with her s
The waitress revealed in a blog how John Key kept pulling her hair.

--------------------------------------------------------------------------------
rouge1:
precision: 0.2398
recall: 0.1802
fmeasure: 0.1719
rouge2:
precision: 0.0416
recall: 0.0266
fmeasure: 0.0257
rougeL:
precision: 0.1718
recall: 0.1195
fmeasure: 0.1155
```

2. LoRA rank 8:

```
Generated:  became a rallying cry for the national movement for missing and m
Reference: Jurors have started deliberations in the case against Pedro Hernan
--------------------------------------------------------------------------------
Generated: The sexy lingerie collection is available in a wide range of colou
Reference: This is Emma Louise Connolly's second campaign for Ann Summers.
The erotic bridal range is priced between £14 and £85.

--------------------------------------------------------------------------------
rouge1:
precision: 0.2151
```

```
recall: 0.1857
fmeasure: 0.1684
rouge2:
precision: 0.0350
recall: 0.0250
fmeasure: 0.0228
rougeL:
precision: 0.1545
recall: 0.1225
fmeasure: 0.1130
```

3. LoRA rank 16:

```
Generated: The Liverpool captain has been linked with a move to Manchester Ur
Reference: Liverpool lost 2-1 to Aston Villa in the FA Cup semi-final at Wemb
Steven Gerrard's dream of making the FA Cup final were shattered.
--------------------------------------------------------------------------------
Generated:  on the banks of the Ganges River in India, where they are greeted
Reference: Uruma Takezawa, a Japanese photographer, has just won the third an

--------------------------------------------------------------------------------
rouge1:
precision: 0.1926
recall: 0.2262
fmeasure: 0.1817
rouge2:
precision: 0.0270
recall: 0.0279
fmeasure: 0.0226
rougeL:
precision: 0.1294
recall: 0.1430
fmeasure: 0.1157
```

4. Prompt Tuning with 20 soft prompts:

```
Generated: This story was originally published on CBC News....
Reference: Loretta Reinholdt, 54, and Andy Wasinger, 46, were on on a hired s
--------------------------------------------------------------------------------
Generated:  man to cut the grass. She also saw a man standing by. She said sh
Reference: Five members of Czech Roma family on trial for people trafficking.
Prosecutors say victims were lured to UK under promise of better life but wer
--------------------------------------------------------------------------------
```

```
rouge1:
precision: 0.1635
recall: 0.2619
fmeasure: 0.1856
rouge2:
precision: 0.0217
recall: 0.0383
fmeasure: 0.0263
rougeL:
precision: 0.1058
recall: 0.1669
```