1) Concept of soft prompts
Overcoming the Constraints of Discrete Text Prompts

Adaptive Versatility: Discrete text prompts, constrained by a fixed vocabulary and the need for manual construction, often fall short in capturing the nuanced requirements of specific tasks. Soft prompts, on the other hand, harness the power of backpropagation to dynamically adjust and incorporate subtle task-related cues, surpassing the limitations imposed by predefined text prompts.

Dynamic Adjustability: Soft prompts possess the unique ability to modify their representations throughout the training process, allowing them to tailor their responses to the intricacies of a given task. This dynamic adaptability stands in stark contrast to the static nature of discrete prompts, offering a distinct advantage in meeting the varied demands of different tasks.

Task-Driven Conditioning: Unlike discrete prompts, soft prompts operate without constraints on length or vocabulary, enabling them to more effectively condition the language model for a specific task.

The Superiority of Soft Prompts in Flexibility and Efficiency

Precision in Task Refinement: Soft prompts excel in nuanced and task-specific tuning by optimizing their embeddings during training for the targeted task. This contrasts with the reliance of discrete prompts on general-purpose, pre-existing embeddings.

Streamlined Workflow: The manual process of identifying suitable discrete prompts is labor-intensive and involves trial and error. In contrast, soft prompts eliminate this need by automatically optimizing themselves, reducing the manual effort required for effective task conditioning.

Optimal Parameter Utilization: Soft prompts showcase the ability to achieve superior performance with a more efficient use of parameters, a crucial advantage in the realm of large-scale models where efficiency is a paramount consideration.

2) Scaling and Efficiency in Prompt Tuning

The Interplay Between Prompt Tuning Efficiency and Model Scale

Benefits of Scale Expansion: As the size of language models increases, signified by a higher number of parameters, there emerges a heightened ability to harness the subtleties embedded in soft prompts. Larger models possess a greater capacity to assimilate and effectively utilize the finely-tuned signals from soft prompts, thereby elevating overall performance.

Limited Gains for Smaller Models: Conversely, the advantages of prompt tuning may not be as pronounced for smaller models due to their limited capacity. Smaller models might struggle to fully capture and leverage the benefits of prompt tuning, resulting in less substantial performance improvements.

Implications for Future Advancements

Scalability's Impact on Model Effectiveness: The observed relationship indicates that, with continued growth in language model size, the efficacy of prompt tuning is likely to increase. This suggests a promising trajectory for the development of more robust and efficient models, particularly beneficial for tasks demanding nuanced comprehension or generation.

Task-Specific Adaptability: The integration of prompt tuning in large-scale models enhances their adaptability to specific tasks. This adaptability proves crucial for applications requiring tailored responses, such as personalized content generation or intricate question answering.

Economic Use of Resources: The ability to employ a single large model across multiple tasks without extensive fine-tuning, facilitated by efficient prompt tuning, translates into a more cost-effective utilization of resources for organizations and researchers.

Resource Allocation Dilemmas: While the prospects are encouraging, the shift towards larger models introduces challenges in computational resources. Larger models demand increased memory and processing power, necessitating a careful consideration of the trade-offs between model size, efficiency, and resource requirements in the ongoing development of language models.

ANALYSIS:

1) **SUMMARISATION:**

   Creation of Soft Prompt: A customized vocabulary, denoted as [SUMMARIZE], is established to serve as the soft prompt.
   Incorporation of Embedding Layer: Within the GPT2WithSoftPrompt class, an embedding layer (utilizing torch.nn.Embedding) is introduced specifically for the soft prompt vocabulary.
   Extension of the Model: This class extends a pre-trained GPT-2 model and, during the forward pass, concatenates the soft prompt embeddings with the base embeddings of GPT-2.
   Adaptation During Training: The model undergoes training with a focus on optimizing solely the parameters associated with the soft prompt embeddings, seamlessly integrating the prompt into the model's generated responses.

   Configuration Parameters

Base Model: GPT-2 (GPT2LMHeadModel).
Soft Prompt Vocabulary: The designated soft prompt vocabulary is ["[SUMMARIZE]"].
Embedding Size: The default size is set to 768, aligning with GPT-2's embedding dimensions.
Training Parameters: Notable settings include a batch size of 1, 10 epochs, 1 gradient accumulation step, and a gradient clipping norm of 1.

Utilized Metric

ROUGE (Recall-Oriented Understudy for Gisting Evaluation):

ROUGE-N: Measures overlap of n-grams between the generated summary and the reference summary.

Soft Prompt

Epoch 1/5: 100%|| 1000/1000[10:16<00:00, 3.74batch/s]

Validation: 100%|| 130/130 [00:10<00:00, 10.17batch/s]

Val Loss : 10.00
Epoch 2/5: 100%| 1000/1000 [10:17<00:00, 3.72batch/s]

Validation: 100%|| 130/130 [00:10<00:00, 10.17batch/s]

Val Loss : 8.78
Epoch 3/5: 100%| 1000/1000 [10:17<00:00, 3.69batch/s]
Train : % Validation: 100%|| 130/130 [00:10<00:00, 9.84batch/s]

Val Loss : 8.54
Epoch 4/5: 100%| 1000/1000 [10:18<00:00, 3.64batch/s]

Validation: 100%|| 130/130 [00:10<00:00, 11.40batch/s]

Val Loss : 8.43
Epoch 5/5: 100%| 1000/1000 [10:17<00:00, 3.69batch/s]

Validation: 100%|| 130/130 [00:10<00:00, 11.35batch/s]

Val Loss : 8.38
Test: 100%|| 110/110 [00:10<00:00, 10.04batch/s]
Test : % Exact Match: 16.5811
Test Loss : 8.48

With Hard Prompt

Test: 100%|██████████| 100/100 [00:10<00:00, 1.36s/batch]

Test Loss : 11.19
Rogue score : 0.27
The Hard Prompt Used is:
“Summarize the following content :”

2) Question answering

Creation of Soft Prompt Vocabulary: A distinct vocabulary for the soft prompt is established, denoted as [QUESTIONANSWERING].
Introduction of Embedding Layer: Within the GPT2WithSoftPrompt class, an embedding layer (utilizing torch.nn.Embedding) is incorporated specifically for the soft prompt vocabulary.
Model Extension: Extending a pre-trained GPT-2 model, the class combines the soft prompt embeddings with GPT-2's base embeddings during the forward pass.
Adaptation during Training: The model undergoes training with a focus on optimizing solely the parameters associated with the soft prompt embeddings, seamlessly integrating the prompt into the model's generated responses.

Configuration Parameters

Base Model: GPT-2 (GPT2LMHeadModel).
Soft Prompt Vocabulary: The designated soft prompt vocabulary is ["[QUESTIONANSWERING]"].
Embedding Size: The default size is set to 768, aligning with GPT-2's embedding dimensions.
Training Parameters: Notable settings include a batch size of 1, 10 epochs, 1 gradient accumulation step, and a gradient clipping norm of 1.

Utilized Metric

Exact Match Percentage: This metric evaluates the correspondence between the sets of output tokens and target summary tokens, calculating the percentage of output tokens that exactly match the target tokens.

Objective of the Metric in Question Answering

This metric serves to evaluate the model's proficiency in capturing crucial information from the target summary, with a specific focus on the accuracy of information reproduction rather than considerations of stylistic nuances or paraphrasing elements.

Soft Prompt

Epoch 1/5:   0%| | 0/1000 [10:00<?, ?batch/s]
Epoch 1/5: 100%|██████████| 5000/5000 [11:05<00:00, 7.66batch/s]
Train : % Exact Match: 2.2589
Validation: 100%|██████████| 5000/5000 [10:19<00:00, 25.48batch/s]

Val : % Exact Match: 2.7082
Val Loss : 11.2295
Epoch 2/5: 100%|| 5000/5000 [11:00<00:00, 8.25batch/s]
Train : % Exact Match: 1.9713
Validation: 100%|| 5000/5000 [10:18<00:00, 26.34batch/s]
Val : % Exact Match: 2.2713
Val Loss : 10.746776478767394
Epoch 3/5: 100%|| 5000/5000 [10:59<00:00, 8.35batch/s]
Train : % Exact Match: 1.9270
Validation: 100%|| 5000/5000 [10:19<00:00, 26.17batch/s]
Val : % Exact Match: 2.3199
Val Loss : 10.5061
Epoch 4/5: 100%|| 5000/5000 [10:59<00:00, 8.36batch/s]
Train : % Exact Match: 2.0257
Validation: 100%|| 5000/5000 [10:19<00:00, 25.85batch/s]
Val : % Exact Match: 2.3833
Val Loss : 10.3091
Epoch 5/5: 100%|| 5000/5000 [10:59<00:00, 8.39batch/s]
Train : % Exact Match: 1.9569
Validation: 100%|| 5000/5000 [10:19<00:00, 25.97batch/s]
Val : % Exact Match: 3.0007
Val Loss : 10.0445

Hard Prompt
Validation: 100%|| 500/500 [10:24<00:00, 2.06batch/s]
Val : % Exact Match: 8.4267
Val Loss : 11.2963

The Hard Prompt Used is:
“Answer the Following Question”

3) Machine translation

Implementation of Soft Prompt in Machine Translation

Creation of Soft Prompt: A unique vocabulary for the soft prompt is specified, denoted as [TRANSLATION].
Introduction of Embedding Layer: Within the GPT2WithSoftPrompt class, an embedding layer (using torch.nn.Embedding) is implemented for the soft prompt vocabulary.
Extension of the Model: The class extends a pre-trained GPT-2 model, incorporating the soft prompt embeddings by concatenating them with GPT-2's base embeddings during the forward pass.

Training Adaptation: During training, the model is optimized exclusively for the parameters associated with the soft prompt embeddings, seamlessly integrating the prompt into the model's generated responses.

Configuration Parameters

Base Model: GPT-2 (GPT2LMHeadModel).
Soft Prompt Vocabulary: The designated soft prompt vocabulary is ["[TRANSLATION]"].
Embedding Size: The default size is set to 768, aligning with GPT-2's embedding dimensions.
Training Parameters: Notable settings include a batch size of 1, 10 epochs, 1 gradient accumulation step, and a gradient clipping norm of 1.

Utilized Metrics

BLEU Score: This metric evaluates the quality of the translated text by comparing it with one or more reference translations. It assesses accuracy and fluency by examining the presence of specific n-gram sequences.
Percentage of Exact Match: This metric compares the sets of output tokens and target summary tokens, calculating the percentage of output tokens that exactly match the target tokens.

Objective of Metrics in Machine Translation

The BLEU Score serves as a widely employed metric in machine translation, focusing on assessing the accuracy and fluency of the output by comparing it with reference translations. The Percentage of Exact Match provides additional insight by measuring the alignment between the output tokens and the target summary tokens, emphasizing precision in trans**l**

Soft Prompt

Epoch 1/5: 100%|██████████| 4800/4800 [11:02<00:00, 7.69batch/s]
Validation: 100%|██████████| 600/600 [10:02<00:00, 23.98batch/s]
Val Loss : 13.1125
Epoch 2/5: 100%|██████████| 4800/4800 [11:00<00:00, 7.94batch/s]
Validation: 100%|██████████| 600/600 [10:02<00:00, 23.55batch/s]
Val Loss : 11.8731
Epoch 3/5: 100%|██████████| 4800/4800 [11:00<00:00, 7.95batch/s]
Validation: 100%|██████████| 600/600 [10:02<00:00, 23.63batch/s]:
Val Loss : 11.7912
Epoch 4/5: 100%|██████████| 4800/4800 [10:59<00:00, 8.00batch/s]
Validation: 100%|██████████| 600/600 [10:02<00:00, 23.99batch/s]
Val Loss : 11.7822
Epoch 5/5: 100%|██████████| 4800/4800 [10:59<00:00, 8.04batch/s]

Validation: 100%|██████████| 600/600 [10:02<00:00, 23.41batch/s]
Test Loss : 8.9886
BLEU SCORE : 6.80

Hard Prompt

Validation: 100%|██████████| 150/150 [10:08<00:00, 1.79batch/s]
Test : % Exact Match: 26.18131868131868
Test Loss : 11.91470266977946
Test BLEU Score: 4.298931957798081e-1

The Hard Prompt Used is:
“Translate the following sentence from english to german :”

Utilizing Hard Prompts proves ineffective in fully harnessing the potential of generative Decoder-only models. The adoption of Prompt Tuning, on the other hand, allows the model to learn the most suitable Soft Prompt for a specific task.

Benefits:

Flexibility and Adaptability:

Soft prompts, being learned entities, can dynamically adjust their embeddings during training, potentially enhancing overall performance.
Hard prompts, although simpler to implement, lack this adaptability and may struggle to capture the intricate nuances essential for complex tasks.
Efficiency in Parameter Usage:

Soft prompts demonstrate the potential to use fewer parameters more efficiently, given their optimization during training.
Hard prompts, constrained by fixed vocabulary and structure, may not optimize well for certain tasks.
Manual Effort and Scalability:

Crafting effective hard prompts can be labor-intensive and lacks scalability.
Soft prompts automate the optimization process, making them more scalable across multiple tasks.
Interpretability and Control:

Hard prompts offer increased interpretability and control since they are manually designed.

Soft prompts are less interpretable due to their learned nature.

Conclusion:

The comparison between soft and hard prompts underscores the trade-offs between adaptability and control. While soft prompts provide dynamic adaptability and efficient parameter usage, hard prompts grant more direct interpretability and control over the model's conditioning. The choice between soft and hard prompts hinges on specific task requirements, desired control levels, and available resources for manual prompt design.