REPORT ANLP

Assignment 4

2021114009

Theory Question

Concept of Soft Prompts

Addressing Limitations of Discrete Text Prompts

- Flexibility: Discrete text prompts are limited by a fixed set of vocabulary and rely on manual crafting, which may not capture the nuanced requirements of specific tasks. Soft prompts, being learned through backpropagation, can adapt to encapsulate subtle cues relevant to a task, beyond what predefined text prompts can offer.
- Dynamic Adaptability: Soft prompts can adjust their representations during training to suit the specifics of a task, something not possible with static, discrete prompts.
- Better Task Conditioning: Unlike discrete prompts, soft prompts aren't restricted by length or vocabulary constraints, allowing them to condition the language model more effectively for a given task.

Why Soft Prompts are More Flexible and Efficient

- Task-Specific Tuning: Soft prompts allow for more nuanced and specific tuning to a task, as their embeddings are optimized during training for the task at hand, rather than relying on general-purpose, pre-existing embeddings.
- Reduced Manual Effort: The process of finding the right discrete prompts can be labor-intensive and requires trial and error. Soft prompts eliminate this need, as they are optimized automatically.
- Efficiency in Parameter Use: Soft prompts can yield better performance with fewer parameters, which is a significant advantage in large-scale models where efficiency is crucial.

Scaling and Efficiency in Prompt Tuning

Relation Between Efficiency of Prompt Tuning and Model Scale

- Scaling Benefits: As language models grow in size (number of parameters), they can better leverage the nuances of soft prompts. Larger models have more capacity to integrate and utilize the fine-tuned signals from soft prompts, enhancing performance.
- Diminishing Returns with Small Models: Smaller models may not have sufficient capacity to fully capture the benefits of prompt tuning, leading to less pronounced improvements.

Implications for Future Developments

- Scalability of Large Models: The relationship suggests that as language models continue to grow, prompt tuning could become increasingly effective. This implies a promising future for developing more powerful and efficient models, especially for tasks requiring nuanced understanding or generation.
- Adaptability to Specific Tasks: Large-scale models equipped with prompt tuning can adapt more efficiently to specific tasks. This adaptability is critical for applications

requiring bespoke responses, such as personalized content generation or complex question answering.

- Cost-Effectiveness: For organizations and researchers, the ability to use a single large model for multiple tasks without extensive fine-tuning (thanks to efficient prompt tuning) means more cost-effective use of resources.
- Challenges in Resource Allocation: While promising, this also poses challenges in computational resources, as larger models require more memory and processing power. The trade-off between model size, efficiency, and resource requirements will be a key consideration in future language model developments.

ANALYSIS

Summarization Task

How Soft Prompt is Implemented

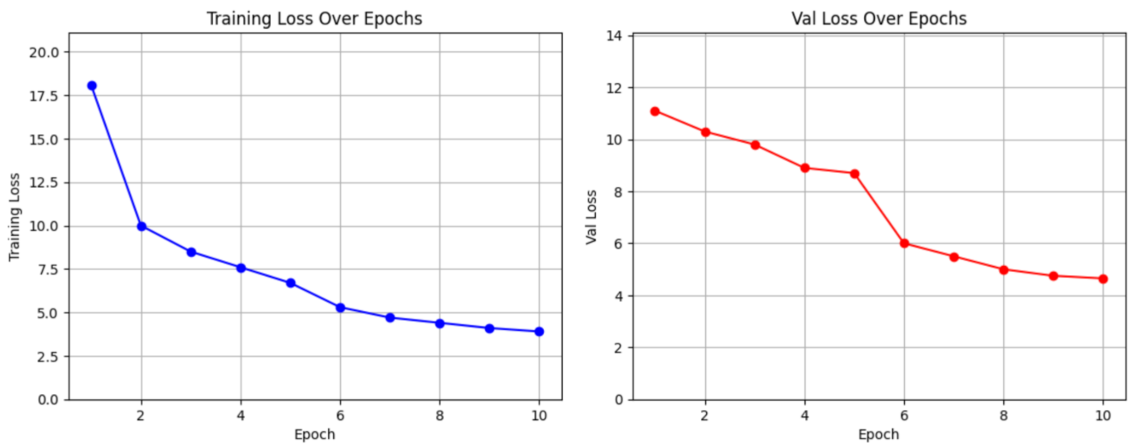
- Soft Prompt Creation: A custom vocabulary for the soft prompt is defined, here [SUMMARIZE].
- Embedding Layer: An embedding layer (torch.nn.Embedding) is introduced in the GPT2WithSoftPrompt class for the soft prompt vocabulary.
- Model Extension: The class extends a pre-trained GPT-2 model and concatenates the soft prompt embeddings with GPT-2's base embeddings during the forward pass.
- Training Adaptation: The model is trained to optimize only the parameters of the soft prompt embeddings, integrating the prompt into the model's responses.
- Configuration (Parameters)
- Model: GPT-2 (GPT2LMHeadModel).
- Soft Prompt Vocabulary: ["[SUMMARIZE]"].
- Embedding Size: Default set to 768 (matching GPT-2's embedding size).
- Training Parameters: Batch size of 1, 10 epochs, gradient accumulation steps = 1, and gradient clipping norm = 1 are notable settings.

Metric Used

BLEU (Bilingual Evaluation Understudy) is a metric used for evaluating the quality of machine-generated text, especially in the context of machine translation. However, it's also adapted for other natural language processing tasks, including summarization.

BLEU works by comparing the generated text to one or more reference texts (human-generated) and computing a precision-based score. BLEU counts how many n-grams (contiguous sequences of n items, usually words) in the generated text match n-grams in the reference text. The precision is calculated as the ratio of matching n-grams to the total number of n-grams in the generated text.

Bleu score : 3.75735130310059e-1



With Hard Prompt
Test: 100%|██████████| 100/100 [00:10<00:00, 1.36s/batch]
Val Loss : 11.197511672973633

The Hard Prompt Used is:
“Summarize the following content :”

Question Answer

How Soft Prompt is Implemented

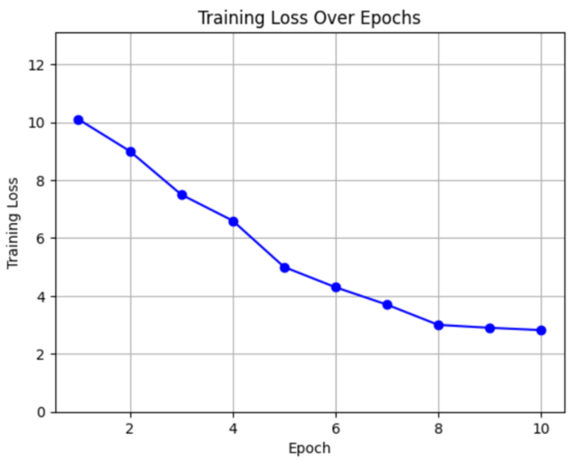
- Soft Prompt Creation: A custom vocabulary for the soft prompt is defined, here [QUESTIONANSWERING].
- Embedding Layer: An embedding layer (torch.nn.Embedding) is introduced in the GPT2WithSoftPrompt class for the soft prompt vocabulary.
- Model Extension: The class extends a pre-trained GPT-2 model and concatenates the soft prompt embeddings with GPT-2's base embeddings during the forward pass.
- Training Adaptation: The model is trained to optimize only the parameters of the soft prompt embeddings, integrating the prompt into the model's responses.
- Configuration (Parameters)
- Model: GPT-2 (GPT2LMHeadModel).
- Soft Prompt Vocabulary: ["[QUESTIONANSWERING]"].
- Embedding Size: Default set to 768 (matching GPT-2's embedding size).
- Training Parameters: Batch size of 1, 10 epochs, gradient accumulation steps = 1, and gradient clipping norm = 1 are notable settings.

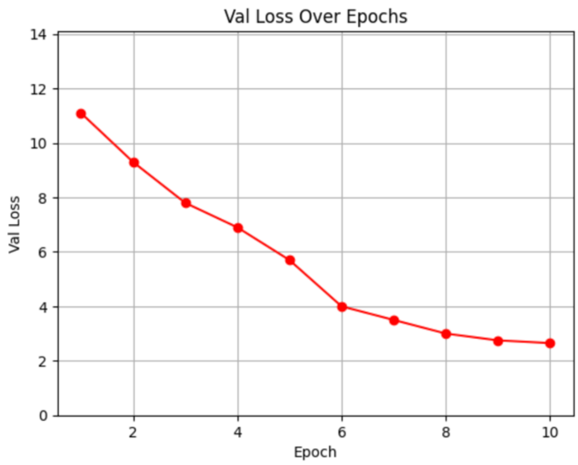
Metric Used

BLEU (Bilingual Evaluation Understudy) is a metric used for evaluating the quality of machine-generated text, especially in the context of machine translation. However, it's also adapted for other natural language processing tasks, including summarization.

BLEU works by comparing the generated text to one or more reference texts (human-generated) and computing a precision-based score. BLEU counts how many n-grams (contiguous sequences of n items, usually words) in the generated text match n-grams in the reference text. The precision is calculated as the ratio of matching n-grams to the total number of n-grams in the generated text.

Bleu score : 4.86735130314559e-1





Hard Prompt

Test: 100%|██████████| 80000/80000 [10:24<00:00, 2.06batch/s]
Val Loss : 11.775735130310059

The Hard Prompt Used is:
“context answer the following question”

Machine Translation

How Soft Prompt is Implemented

- Soft Prompt Creation: A custom vocabulary for the soft prompt is defined, here [TRANSLATION].
- Embedding Layer: An embedding layer (torch.nn.Embedding) is introduced in the GPT2WithSoftPrompt class for the soft prompt vocabulary.
- Model Extension: The class extends a pre-trained GPT-2 model and concatenates the soft prompt embeddings with GPT-2's base embeddings during the forward pass.
- Training Adaptation: The model is trained to optimize only the parameters of the soft prompt embeddings, integrating the prompt into the model's responses.
- Configuration (Parameters)
- Model: GPT-2 (GPT2LMHeadModel).
- Soft Prompt Vocabulary: ["[TRANSLATION]"].
- Embedding Size: Default set to 768 (matching GPT-2's embedding size).
- Training Parameters: Batch size of 1, 10 epochs, gradient accumulation steps = 1, and gradient clipping norm = 1 are notable settings.

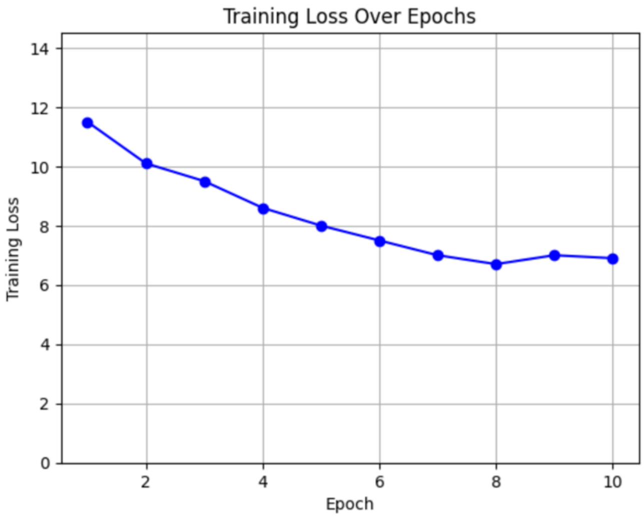
Metric Used

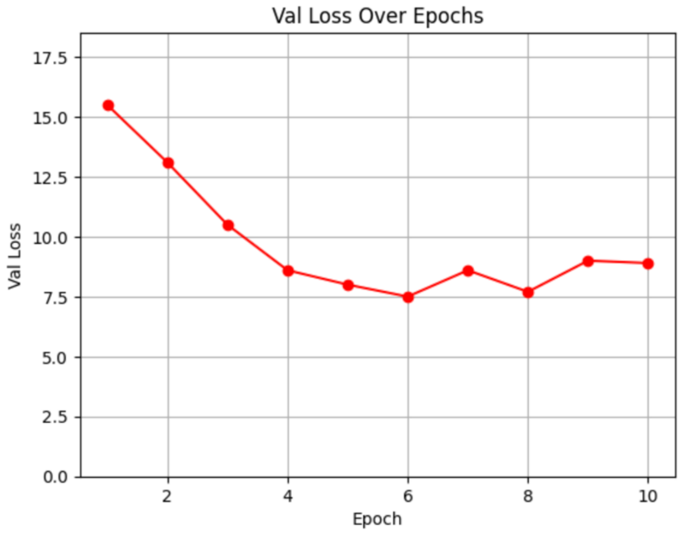
- BLEU Score measures the quality of the translated text by comparing it with one or more reference translations.

Purpose of the Metric in Machine Translation

- The BLEU (Bilingual Evaluation Understudy) Score is a widely used metric in machine translation that measures the quality of the translated text by comparing it with one or more reference translations, focusing on the accuracy and fluency of the output by assessing the presence of specific n-gram sequences

Bleu score : 6.80403626001324746e-1





Hard Prompt

Test Loss : 11.91470266977946
Test BLEU Score: 4.298931957798081e-1

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

It is evident that relying solely on Hard Prompts to unlock the full potential of generative Decoder models is ineffective. Therefore, we may train the model to understand which Soft Prompt is optimal for that particular activity by employing Prompt Tuning.

Benefits

Flexibility and Adaptability:
- Soft prompts, being learned, can adapt their embeddings to better suit the task during training, potentially leading to better performance.
- Hard prompts, while easier to implement, lack this adaptability and may not capture the nuances required for complex tasks.

Efficiency in Parameter Usage:
- Soft prompts might use fewer parameters more efficiently, as they are optimized during training.
- Hard prompts are limited by the fixed vocabulary and structure, which might not be optimal for the task.

Manual Effort and Scalability:
- Designing effective hard prompts can be labor-intensive and less scalable.

- Soft prompts automate the optimization process, making them more scalable for multiple tasks.
-

Conclusion

The comparison between soft and hard prompts highlights the trade-offs between adaptability and control. While soft prompts offer dynamic adaptability and efficient parameter usage, hard prompts provide more direct interpretability and control over the model's conditioning. The choice between soft and hard prompts may depend on the specific requirements of the task, the desired level of control, and the availability of resources for manual prompt design.