سیده شکیبا انارکی فیروز ۹۹۴۴۲۰۴۷
تمرین پنجم درس پردازش زبان ها

۱) دو سناریوی آموزش مدلی مانند BERT را در نظر بگیرید که در اولی وزن های اولیه ی آموزش برابر با وزن های یک مدل BERT از قبل آموزش دیده شده pre-trained بر روی مجموعه تسک های مختلف میباشد. و در دومی وزن های اولیه مقادیری تصادفی می باشند. توضیح دهید هر کدام از مدل ها در فرآیند آموزش به طور تقریبی چگونه عمل خواهند کرد و همینطور انتظار داریم عملکرد مدل ها پس از فرآیند آموزش چگونه باشد؟

When initializing with pre-trained BERT weights, the model already contains useful linguistic knowledge from the pre-training on large corpora. During fine-tuning on a specific task:

- The earlier layers tend to preserve the general language understanding capabilities learned during pre-training, while the later layers adapt to encode task-specific features.
- The last two layers undergo the largest changes compared to the pre-trained model, encoding features crucial for the target task performance.

When initializing with random weights, the model starts from scratch without any prior knowledge, So:

- All layers must learn the linguistic knowledge and task-specific features from scratch during fine-tuning.
- This results in much lower performance compared to pre-trained initialisation, and in some cases even underperforms the pre-trained BERT without any fine-tuning.

Fine-tuned BERT initialized with pre-trained weights significantly outperforms randomly initialized BERT on most tasks, indicating the pre-trained weights provide a strong starting point for the model to learn the target task.

https://aclanthology.org/D19-1445.pdf
https://stackoverflow.com/questions/68058647/initialize-huggingface-bert-with-random-weights
https://arxiv.org/pdf/2110.03848.pdf

۲) یکی از چالش های قابل توجه در فرآیند آموزش یک مدل شبکه عصبی بخصوص در فرآیند Fine-Tuning، Catastrophic Forgetting میباشد. این چالش را به طور کامل توضیح دهید و روشی ارائه دهید که بتوانیم این چالش را کاهش دهیم و یا برطرف کنیم.(منابعی که در آن ها جست و جو کردید را ارائه دهید.)

One issue when you fine tune a model is "Catastrophic Forgetting". This occurs when the model forgets the previous knowledge acquired during the pre-training. This results in incorrect output for previously trained knowledge, since fine tuning has affected the weights of the model.

The main reason for this is Single Task Fine Tuning in which you adjust the model to make it more suitable for the specific behaviour you need for one task, ignoring how it may adversely affect other tasks. While the model is now fine-tuned for the domain that you added, this single fine-tuning process might have adjusted the weights of the model in such a way that it stops performing well on general language comprehension or other tasks.

There are a couple of ways you can avoid Catastrophic Forgetting.

Fine tune on Multiple Tasks

Training the model on multiple tasks together. That way training instructions for one task does not affect the model weights impacting other areas. This is difficult due to the need for compilation of large amount of instruction sets which is directly proportional to number of tasks you are fine-tuning on.

Parameter Efficient Fine Tuning (PEFT)

With "Parameter Efficient Fine Tuning", you only train with a small number of weights avoiding a full change to the model. This not only preservers the core model weights but is also efficient and can be completed with low computation overheads. There are different PEFT approaches you can consider.

Reparameterization: where you selectively change the existing weights.

Additive: where you add delta set of weights not affecting the existing weights. This adds to the existing model as opposed to changing it.

Prompt tuning: where you add a soft prompt or trainable tokens to your prompt. This is done at the inference time.

https://www.linkedin.com/pulse/catastrophic-forgetting-side-effect-fine-tuning-large-karan-sehgal-/jjkqe
https://www.youtube.com/watch?v=-RgJx07jjCk
https://www.youtube.com/watch?v=f-z9whEdqFY
https://arxiv.org/pdf/1911.00202

۳)در رابطه با Transfer Learning و تفاوت های آن با Fine-Tuning تحقیق کنید و توضیح دهید در چه شرایطی از هرکدام استفاده میشود؟

Transfer learning is a broader concept that encompasses any scenario where a model developed for one task is repurposed on a second related task. It involves taking a pre-trained model, typically trained on a large dataset like ImageNet, and adapting it to a new but related problem. For instance, a model trained to recognize a wide array of animals might be adapted to specialize in identifying different breeds of dogs. In this case, the original model's learned features, such as edges and textures, can be leveraged for the new task, thereby reducing the need for extensive training data and computational resources.

Use case: When limited labeled data or computational resources are available, and tasks share similarities.

Fine-tuning, on the other hand, is a specific form of transfer learning. It involves taking a pre-trained model and continuing the training process on a new, typically smaller, dataset. This allows the model to adjust its weights and biases to better suit the new task. Fine-tuning often involves adjusting the learning rate to prevent overwriting the previously learned features too rapidly. For example, a model trained to perform general object recognition might be fine-tuned on a dataset of medical images to specialise in identifying specific pathologies.

Use case: When task-specific data is available and computational resources allow full retraining.

As a summary , Transfer learning is when a model developed for one task is reused to work on a second task. Fine-tuning is one approach to transfer learning where you change the model output to fit the new task and train only the output model.

https://www.linkedin.com/pulse/difference-between-fine-tuning-transfer-learning-devansh-kumar-bay3e/

https://www.geeksforgeeks.org/what-is-the-difference-between-fine-tuning-and-transfer-learning/

https://stats.stackexchange.com/questions/343763/fine-tuning-vs-transferlearning-vs-learning-from-scratch

۴) راجع به تاثیرات روش های مختلف masking و همینطور تعیین میزان توکن های قابل mask بر روی فرآیند آموزش و عملکرد مدل های MLMs توضیح دهید.

روش رندوم و روش مبتنی بر (part of speech)

Random Masking: Tokens are randomly selected for masking. This is the standard approach used in models like BERT.
Advantages:
- Simplicity: Easy to implement and understand.
- Uniform Learning: Ensures all parts of the input sequence have an equal chance of being masked and learned.

Disadvantages:
- Context Ignorance: Does not consider linguistic or syntactic information, which might limit the learning of more complex language structures.

Part-of-Speech (POS) Based Masking: Tokens are selected for masking based on their part of speech. For instance, masking nouns, verbs, or adjectives with different probabilities.
Advantages:
- Targeted Learning: By focusing on specific parts of speech, the model can learn more nuanced representations of those categories, which can be beneficial for certain tasks.
- Enhanced Context Understanding: Helps the model to better understand and predict syntactic and semantic structures.

Disadvantages:
- Complexity: Requires additional preprocessing to determine the part of speech for each token.
- Bias: Might introduce bias if certain parts of speech are overly masked or ignored.

Proportion of Tokens Masked
Low Masking Proportion (e.g., 10%):
Advantages:
- Stable Training: Lower chance of disrupting the context too much, leading to more stable training.
- Rich Context: The model has more context to use for predicting the masked tokens, which can improve prediction accuracy.

Disadvantages:
- Slower Learning: The model might take longer to learn robust representations since fewer tokens are masked in each training instance.
- Overfitting Risk: With less masking, the model may overfit to the context provided by the unmasked tokens.
- High Masking Proportion (e.g., 50%):

Advantages:
- Faster Learning: More tokens are masked, which can speed up the learning process as the model needs to predict more missing information.

- Robust Representations: The model learns to infer missing tokens from less context, which can lead to more robust representations.

Disadvantages:

- Context Loss: Too many masked tokens can result in a loss of context, making it harder for the model to learn effectively.
- Unstable Training: High masking proportions can lead to unstable training as the model might struggle to predict tokens with insufficient context.

https://medium.com/@ngiengkianyew/implementation-of-a-simple-masked-language-model-6a3bc18912c8

۵) عملکرد معماری های CLM, MLM, Seq2Seq را با یکدیگر مقایسه کنید و مزایا و معایب هرکدام را بیان کنید. و از هرکدام نمونه هایی را مثال بزنید.

Causal Language Modeling (CLM)

CLM is an autoregressive method where the model is trained to predict the next token in a sequence given the previous tokens. CLM is used in models like GPT-2 and GPT-3 and is well-suited for tasks such as text generation and summarization. However, CLM models have unidirectional context, meaning they only consider the past and not the future context when generating predictions.

Masked Language Modeling (MLM)

MLM is a training method used in models like BERT, where some tokens in the input sequence are masked, and the model learns to predict the masked tokens based on the surrounding context. MLM has the advantage of bidirectional context, allowing the model to consider both past and future tokens when making predictions. This approach is especially useful for tasks like text classification, sentiment analysis, and named entity recognition.

Sequence-to-Sequence (Seq2Seq)

Seq2Seq models consist of an encoder-decoder architecture, where the encoder processes the input sequence and the decoder generates the output sequence. This approach is commonly used in tasks like machine translation, summarization, and question-answering. Seq2Seq models can handle more complex tasks that involve input-output transformations, making them versatile for a wide range of NLP tasks.

Architecture	Advantages	Disadvantages
Seq2Seq	- Versatile for many tasks - Effective handling of long-range dependencies	- Complex training - High latency in inference
MLM	- Captures bidirectional context - Effective for many downstream tasks via transfer learning	- Not inherently generative - High pretraining cost
CLM	- Strong generative capabilities - Simpler training process	- Limited to unidirectional context - Higher risk of overfitting specific patterns

/https://heidloff.net/article/causal-llm-seq2seq

https://ritikjain51.medium.com/llms-model-architectures-and-pre-training-objectives-39c4543edef0

/https://www.linkedin.com/pulse/demystifying-large-language-models-architecture-sai-panyam

https://medium.com/@tom_21755/understanding-causal-llms-masked-llm-s-and-seq2seq-a-guide-to-language-model-training-d4457bbd07fa

۶) با توجه به عمکرد مدل های MLM(پیشبینی توکن های mask شده) چگونه میتوان از آن ها برای تولید یک دنباله ای از متن استفاده کرد؟

Masked Language Models (MLMs) like BERT are primarily designed for understanding and representing the context and relationships between words in a sentence, rather than generating full sequences of text. However, they can be adapted and fine-tuned for text generation tasks as well.

1. Fine-tuning for Language Generation: While MLMs are pre-trained on the task of predicting masked tokens, they can be fine-tuned on a language modelling objective (predicting the next word given previous words) using a task-specific dataset. This allows the model to learn patterns for generating coherent text sequences. Models like GPT-2 and GPT-3 use this approach.

2. Iterative Text Generation: MLMs can be used iteratively to generate text by predicting one masked token at a time and feeding the predicted token back into the input sequence. This process is repeated to generate the next token until the desired text length is reached.

3. Encoder-Decoder Architecture: MLMs can be combined with a separate decoder model in an encoder-decoder architecture, where the MLM encodes the input context, and the decoder generates the output text sequence. Models like BART and T5 use this approach.

4. Conditional Text Generation: MLMs can be fine-tuned for conditional text generation tasks, where the model generates text based on a given prompt or context. This is useful for applications like dialogue systems, story generation, or summarization.

It's important to note that while MLMs can be adapted for text generation, they may not perform as well as models specifically designed for this task, like GPT-3 or other large language models trained on massive text corpora. Additionally, careful fine-tuning and task-specific data are crucial for achieving high-quality text generation with MLMs.

https://medium.com/analytics-vidhya/fine-tune-a-roberta-encoder-decoder-model-trained-on-mlm-for-text-generation-23da5f3c1858

https://www.techtarget.com/searchenterpriseai/definition/masked-language-models-MLMs

/https://www.scaler.com/topics/nlp/masked-language-model-explained

7)Please provide a comprehensive explanation addressing the rationale behind this masking strategy. Your response should cover the following aspects:

1. 80% Masked with `[MASK]` Token

Why are 80% of the masked tokens replaced with the `[MASK]` token?

Replacing 80% of the masked tokens with the `[MASK]` token primarily serves to provide the model with clear and unambiguous signals about which tokens it should predict during training. This focused approach ensures that the model dedicates a significant portion of its capacity to learning the relationships between the masked token and its context.

Impact on Training and Learning:

- **Clear Learning Signal:** By frequently using the `[MASK]` token, the model receives a strong, consistent signal about where the gaps in the input are, allowing it to learn to leverage the surrounding context effectively.
- **Contextual Understanding:** This high percentage encourages the model to develop a deep understanding of contextual dependencies, as it must rely on the surrounding tokens to infer the masked word.
- **Efficiency:** It ensures that a substantial amount of training time is spent on the primary task of predicting missing words, which is central to the model's pre-training objective.

2. 10% Replaced with Random Words

Why are 10% of the masked tokens randomly replaced with other words from the vocabulary?

Introducing noise by replacing 10% of the masked tokens with random words helps in making the model more robust and less likely to overfit to the `[MASK]` token.

Impact on Robustness and Handling Novel Input:

- **Robustness to Noise:** By occasionally seeing incorrect or unexpected tokens, the model learns to handle noisy data and becomes more resilient to variations and errors in input during real-world applications.
- **Generalization:** This practice encourages the model to generalize better by exposing it to a wider variety of token-context pairings, reducing the likelihood of overfitting to the specific training data.
- **Error Tolerance:** In real-world scenarios, inputs may contain errors, typos, or previously unseen words. Training with random word replacements prepares the model to manage such inconsistencies more effectively.

3. 10% Left Unchanged

Why are the remaining 10% of the masked tokens left as is, unchanged?

Leaving 10% of the masked tokens unchanged serves to regularize the training process and prevent the model from becoming overly dependent on the `[MASK]` token.

Impact on Generalization and Avoiding Overfitting:

- **Regularization:** By not masking some tokens, the model is encouraged to pay attention to and learn from tokens that are always present, promoting a more balanced learning process.
- **Avoiding Overfitting:** If all masked tokens were replaced with the `[MASK]` token, the model might overfit to the training strategy rather than learning to predict tokens in general contexts. Leaving some tokens unchanged helps mitigate this risk.
- **Versatility:** This approach ensures that the model can still perform well even when the `[MASK]` token is not present during fine-tuning or downstream tasks, as it has learned to make predictions based on a variety of contexts.

As you can see, the output of the evaluation is quite poor. Why? Because we started training the MLM from scratch. If we want to achieve an acceptable performance similar to a pre-trained BERT model, we need to perform several steps.

Question

What steps can you take to improve the performance of your Masked Language Model (MLM)?

There are so many actions that we can take. Here are some samples in different aspects:

Data Preparation and Pre-processing

a. Data Quality:

- **High-Quality Corpus:** Ensure the pre-training corpus is diverse and high-quality, covering a wide range of topics and language structures.
- **Data Cleaning:** Remove noise, duplicates, and irrelevant data to improve the overall quality of the training data.

b. Data Augmentation:

- **Synonym Replacement:** Use techniques like synonym replacement to augment the data and introduce variability.
- **Back-Translation:** Translate sentences to another language and back to the original to create paraphrased versions.

Masking Strategy Adjustments

a. Dynamic Masking:

- Implement dynamic masking where the masking pattern changes each epoch, providing the model with varied learning experiences.

b. Masking Proportion:

- Experiment with different proportions of masked tokens (e.g., 15% instead of 20%) to find the optimal balance for the specific dataset.

Model Architecture

- **Deeper Networks:** Increase the number of layers (depth) to allow the model to learn more complex representations.
- **Wider Networks:** Increase the hidden size (width) to enable the model to capture more information.

Training Techniques

a. Learning Rate Schedules:

- **Warmup and Decay:** Use learning rate warmup followed by a decay schedule (e.g., linear or cosine decay) to stabilize training.

b. Regularization:

- **Dropout:** Apply dropout to prevent overfitting by randomly dropping units during training.
- **Weight Decay:** Use weight decay (L2 regularization) to penalize large weights and promote simpler models.

c. Mixed Precision Training:

- Implement mixed precision training to speed up training and reduce memory usage, allowing for larger batch sizes.

Using Task-Specific Fine-Tuning Strategies

- **Layer-Wise Learning Rate Decay:** Apply different learning rates to different layers during fine-tuning, typically with lower rates for the earlier layers.

Incorporating External Knowledge

- **Knowledge Graphs:** Integrate information from knowledge graphs or external databases to provide additional context and improve model understanding.