HW3 Text Generation with the Transformer Decoder

计11 张凯文 2020080094

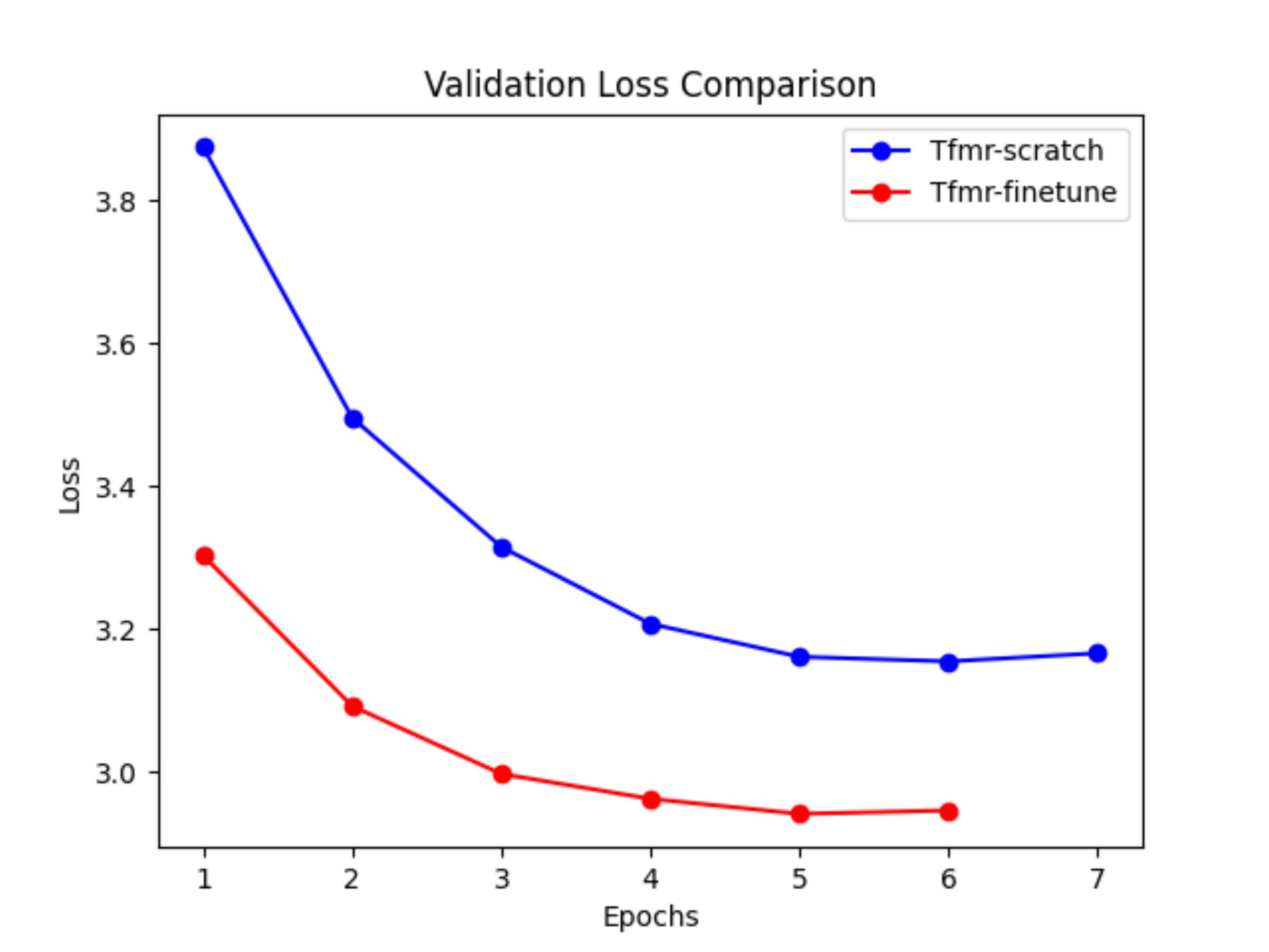
Project environment

```
Python 3.8
PyTorch 1.1
nltk 3.5
Macbook Pro Apple M1 chip
```

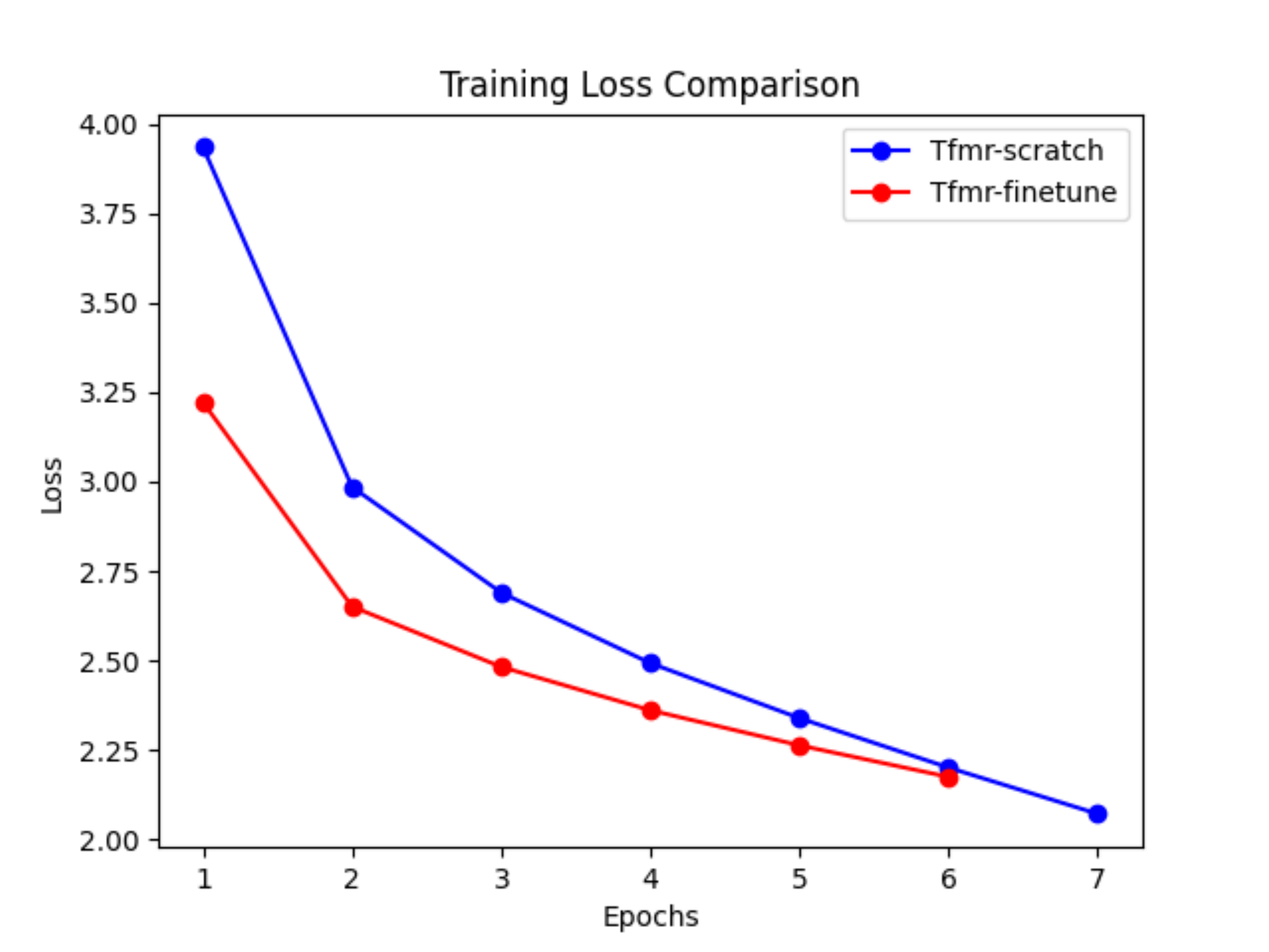
1. Tfmr-scratch vs Tfmr-finetune

Graphical Comparison

Validation Loss:



Training Loss:



Test Results

Model	Perplexity	forward BLEU-4	backward BLEU-4	harmonic BLEU-4
Tfmr-scratch	18.74	0.577	0.424	0.489
Tfmr-finetune	15.41	0.570	0.430	0.490

Analysis

Based on the graph and model performance data, it is evident that the pretrained model converges faster and possesses a smaller loss on both the training and validation sets, as well as a significantly lower Perplexity on the test set.

In terms of BLEU score, there is no significant difference between the fine-tuned pretrained model and the one with randomly initialized parameters. This may be because of the BLEU Score not fully reflecting the overall generative quality of the model. BLEU tends to focus on the degree of n-gram overlap, and its score is not perfect. It may not accurately assess the reasonableness of the sentence, semantic correctness, or the fluency of the text, and is merely based on superficial lexical matching.

Compared to random parameter initialization, the reasons for the better overall performance of the pretrained model include:

- The pretrained model has already learned the basic features of the data, which means the model does not need to learn all the features from scratch but can adapt to specific tasks by fine-tuning the existing features.
- When the amount of data is limited, training a model from scratch tends to lead to overfitting issues. Pretrained models can achieve better performance and faster convergence by utilizing generalized features learned from large datasets.
- With sufficient computational resources, pretrained models can learn deeper features in an environment trained with a large amount of data beforehand, and these deep learning features are more profound and abundant than those learned from scratch.
- During the fine-tuning process, pretrained models can effectively avoid problems such as getting trapped in local minima.

2. Decoding Strategy

Table:

Model	forward BLEU-4	backward BLEU-4	harmonic BLEU-4
Tfmr-scratch ($Random,\ \tau = 1$)	0.577	0.424	0.489
Tfmr-scratch ($Random,\ \tau = 0.7$)	0.816	0.382	0.521
Tfmr-scratch ($top - p = 0.9,\ Temperature = 1$)	0.705	0.410	0.518
Tfmr-scratch ($top - p = 0.9,\ Temperature = 0.7$)	0.886	0.308	0.457
Tfmr-finetune ($Random,\ \tau = 1$)	0.570	0.430	0.490
Tfmr-finetune ($Random,\ \tau = 0.7$)	0.805	0.386	0.522
Tfmr-finetune ($top - p = 0.9,\ Temperature = 1$)	0.688	0.413	0.516
Tfmr-finetune ($top - p = 0.9,\ Temperature = 0.7$)	0.879	0.315	0.464

Analysis:

1. **Decoding Strategy:**
   - "Top-p" sampling tends to produce more diverse and coherent text, as it restricts the generation to a subset of probable tokens. This is evident from the generally higher forward BLEU scores for "top-p" compared to "Random" decoding strategy.
   - "Random" sampling with lower temperature (τ = 0.7) seems to provide a better balance between forward and backward BLEU scores, leading to a higher harmonic BLEU-4 score in the "Tfmr-finetune" model. This suggests that a degree of randomness can help maintain a balance between fluency and diversity in the generated text.
2. **Temperature (τ):**
   - In both "Tfmr-scratch" and "Tfmr-finetune" models, a lower temperature increases the forward BLEU score, indicating that the generated text is more predictable and closely matches the reference text. However, it does not necessarily improve the harmonic BLEU-4 score, which considers both precision and recall in the context of forward and backward BLEU.

3. **Overall Performance:**
    - Comparing the "scratch" and "finetune" versions of the Transformer, we see that fine-tuning does not dramatically change the BLEU scores across different strategies and temperatures. This suggests that both models are likely well-trained and that fine-tuning offers marginal benefits within the context of these BLEU score metrics.
    - The best harmonic BLEU-4 score is achieved with $Tfmr - finetune(Random, \tau = 0.7)$, implying that for the fine-tuned model, a slightly deterministic approach (lower temperature with random sampling) balances well between precision and recall.

3. Random Sentence

Sentences were selected at random then fed into `Grammarly.com` to check for grammatical errors, the corresponding errors were then underlined in each sentence.

Tfmr-scratch:

Model	Random Sentences
Tfmr-scratch ($Random,\ \tau = 1$)	A street corner with a fire hydrant and a street light lit up. A traffic light sitting in the middle of a road. Five red and engines are seen in sun wandering accessories. A giraffe standing with a few people behind it. Two women standing next to each other on a bench. An old school bus is parked in just crowded wide garage. A brown bear holding a stuffed bear on orange cat while stroller. A group of zebras in a grassy field. A busy street corner with two people crossing the street. The yellow double decker tour tour bus is

	stopped at a bus stop.
Tfmr-scratch ($Random,\ \tau = 0.7$)	A blue and white bus parked in the middle of a city. A little girl sitting on top of a wooden bench. A carrot from an old old fashioned bus on a street. A red fire hydrant with a blue top on the sidewalk. A blue bus driving down a street next to a tall building. A green and blue bus parked outside in a parking lot. A bus is taking off from the side of a street. A bus riding down the road while people look at them. A man sitting on a bench with a dog looking out in the window. A herd of animals laying in a field of grass.
Tfmr-scratch ($top - p = 0.9,\ Temperature = 1$)	A giraffe leaning over a drink of water from a metal fence. Three giraffes stand together in front of a window at a zoo. A police officer is standing on a city street near a fire hydrant. A pair of dogs standing in a blue yard near a fire hydrant. The giraffe is eating off the side of the wall. The person is walking a bus and looking up at the bus stop. Two zebra and blue sheep are facing a fence standing near grass. A city street with a town at night lit up. A full view of a plane flying in the sky. Two giraffes standing in green field next to a tree.

Tfmr-scratch ($top-p=0.9,\ Temperature=0.7$)	A giraffe standing next to a tree filled with trees. A man sits on a bench with a dog on a leash. A red and white bus is parked near a curb. A woman sitting on a bench in a park with a dog sitting in her hand. A woman is walking down the street next to a bus. A large jetliner flying over a park with a bench. A woman is leaning on a bench next to a tree. A giraffe is eating leaves from a tree in a field. A bus is parked on the side of the road. A red fire hydrant sitting in the middle of a grassy area.

Tfmr-finetune:

Model	Random Sentences
Tfmr-finetune ($Random,\ \tau=1$)	A man and woman sitting on a bench in a very large field. A wooden bench sitting on top of a field with trees. A wooden bench in a park with a tree in the woods. A red and white bus driving down a street without an. A plane with a lot of people waiting at an airport. A stop light sitting at the end of the road. A black and white photo with a bus parked at the curb. A crowd of people standing on top of a white bus. A man riding a paddle board on the beach in a pasture. A wooden bench sitting inside of a tall tree

	with trees in the background herself.
Tfmr-finetune ($Random,\ \tau = 0.7$)	A couple of giraffes grazing in a field of grass. A tall giraffe standing in the middle of a green grassy area. A tall giraffe standing around in a field with trees in the background. A bench sitting in the grass with a tree in the background. A woman sits on a bench in a park with her backpack. An empty white bus parked in front of a car. A street scene with a dog laying on it. A man riding a bike near a bus while others watch. A red bus driving down a road next to a bus. A white and blue fire hydrant sitting in a field.
Tfmr-finetune ($top - p = 0.9,\ Temperature = 1$)	Two giraffes stand near rocks with trees in a wilderness area. A person taking a picture of the three wheelie metro train. A man sits on a bench with many bottles of water. A bench is perched on a metal platform with a goal of a bench. A black and white picture of the mountains which is near a blue sky. A bird perched on top of a ladder looking over the ocean. A lone giraffe standing on the roof of a lush green hillside. There is a orange and green double decker bus parked in the road. A woman sitting next to a girl sitting on a wooden bench. A woman is walking through a forest and

	jumping off the ocean.
Tfmr-finetune ($top - p = 0.9,\ Temperature = 0.7$)	A man is sitting on a bench with a dog. A giraffe standing in a field next to a tree. A man sits on a bench with a bag of shoes. A bench is sitting in the shade of trees in the woods. A black and white photo of a woman on a bench. A giraffe standing next to a tree in a field. A man sits on a bench with a dog on his leash. A red and white bus is parked near a bus stop. A woman sitting on a bench in a park with a dog in the back seat. A woman is sitting on a bench next to a bench.

Analysis:

1. **Decoding Strategy and Temperature Impact**:
   - Models using a "Random" strategy with a lower temperature ($\tau = 0.7$) generally produce sentences that are more coherent than those generated with a higher temperature ($\tau = 1$), which introduces more randomness.
   - "Top-p" sampling with a high temperature ($\tau = 1$) tends to generate more diverse sentences but may sometimes include irrelevant or nonsensical phrases. Reducing the temperature (to 0.7) with "top-p" sampling appears to strike a better balance between coherence and diversity.
2. **Best Sentence Generation**:
   - The "Tfmr-scratch" model with $top - p, \tau = 0.7$ and the "Tfmr-finetune" model with $top - p = 0.9, Temperature = 0.7$ seem to generate the most coherent and contextually appropriate sentences. The fine-tuned model appears to have a slight edge in consistency and grammatical correctness.

While "top-p" sampling with a controlled temperature seems to result in higher quality sentence generation in fine-tuned models, the metrics provided do not always align with qualitative assessments of sentence quality. A model's ability to generate contextually appropriate, diverse, and coherent text is often best judged by a combination of quantitative metrics and human evaluation.

4. Final Network

After fine-tuning and experiments, the hyperparameters with the best results are as follows:

- Decoding strategy: `Random`
- Temperature: `0.7`
- Other hyperparameters remain the same

Results on these hyperparameters:

Perplexity	forward BLEU-4	backward BLEU-4	harmonic BLEU-4
18.68	0.816	0.382	0.521

5. Questions

1. **Compare Transformer and RNN from at least two perspectives such as time / space complexity,**

   performance, positional encoding, etc.

   Time/Space Complexity:

   - **Transformers**: Enable parallel processing, leading to faster training but higher memory usage due to attention scores.
   - **RNNs**: Process data sequentially, resulting in slower training but typically use less memory.

   Performance:

   - **Transformers**: Excel in tasks requiring understanding of full context and long-range dependencies.
   - **RNNs**: Better for short-range dependencies, but may struggle with longer sequences due to vanishing gradients.

   Positional Encoding:

   - **Transformers**: Require positional encodings to maintain sequence order.

    - **RNNs**: Inherently consider sequence order, so no positional encoding is needed.

2. **Regarding the inference time complexity, answer the following question.**

    1. **During inference, we usually set use_cache in model_tfmr.py to True . What is the argument used for?**

        The `use_cache` argument in `model_tfmr.py` is typically set to `True` during inference to enable the model to reuse previously computed hidden states instead of recalculating them for each new token. significantly speeding up inference by avoiding redundant computations.

    2. **Denote the whole sequence as, please give the inference time complexity when decoding the token ,i.e., the -th loop in the inference function of model_tfmr.py when decoding the first example, and the whole time complexity for decoding the whole sequence . We denote the hidden state dimension as (so that the dimension of the intermediate state of the feed forward layer is), the number of heads in multi-head attention as , the number of hidden Transformer blocks as , the vocab size as .**

        The inference time complexity when decoding a token `lt` in a sequence `L` is generally O(n * d), where `n` is the number of attention heads, and `d` is the dimension of the hidden state. For decoding the entire sequence, the complexity is O(L * n * d), where `L` is the sequence length. The factor of 4d for the feed-forward layer's intermediate state does not change the overall complexity since the feed-forward layers operate in a constant time relative to the sequence length.

    3. **Based on your analysis of the question No 2. , in which case the self-attention module dominate the time complexity? And in which case the feed-forward layer is dominant?**

        The self-attention module dominates the time complexity when the sequence length is large because it involves computing attention scores for each pair of tokens, which is quadratic with respect to the sequence length (O(L^2 * d)). The feed-forward layer is dominant when the sequence length is short or comparable to the size of the hidden dimension because its complexity is linear with respect to the sequence length (O(L * d)).

3. **Discuss the influence of pre-training regarding the generation results, convergence speed, etc. Considering the experimental setup (the training task, data, pre-trained checkpoints, etc.), does the influence of pre-training meet your expectation?**

    - **Generation Results**: Pre-training on large datasets enables models to learn a larger representation of the language, which often results in higher quality and more coherent text generation. This is because the model has already learned common patterns in the language, which can be fine-tuned for specific tasks with less data.
    - **Convergence Speed**: Models that have been pre-trained typically converge faster during fine-tuning because they start from a knowledgeable state rather than learning from scratch. The initial weights provide a good starting point, which means less training is required to reach optimal performance.

    Regarding expectations, if the experimental setup involves a task similar to the pre-training regime, one can expect significant benefits from pre-training. For example, a model pre-trained on a diverse text corpus and then fine-tuned for a language generation task would likely meet high expectations in terms of output quality and training efficiency.

    However, if the fine-tuning task is very different from the pre-training data, the benefits may not be as pronounced, and further adjustments or continued pre-training with relevant data might be necessary. In summary, pre-training typically meets and often exceeds expectations in terms of both generated text quality and convergence speed, provided the tasks are well-aligned.