COMP4901B: Large Language Models

Assignment 2 Report

HE, Wenqian
Student ID: 20860896

November 3, 2025

1 Part 1: Single-turn Loss Masking

```
$ python conversation_func.py
single_turn validation passed. Outputs match the golden answers.
```

I use a for loop to loop through the messages and detect the role. If the role is "assistant", I will get the start and end index of the assistant message from the prefix lengths. Then I will copy the assistant message tokens to the same positions in the labels.

2 Part 2 — Multi-turn Loss Masking

```
$ python conversation_func.py --multi-turn
multi_turn validation passed. Outputs match the golden answers.
```

I use the same logic as Part 1. To make sure the truncation is handled correctly, I use the min function to get the bounded start and end index of the assistant message. If the start index is greater than or equal to the end index, I will break the loop.

3 Part 3 — Reverse Loss Masking (Single-turn with Message Reordering)

```
$ python reverse_conversation_func.py
single_turn validation passed. Outputs match the golden answers.
```

I used the same logic as Part 1. Only modifying the role detection to "user" instead of "assistant".

After masking the assistant message, the model will learn to predict the user message, which is usually the question from the user. A real world application scenario is to recommend some follow-up quesitions of the user, such that the user can select to ask without typing the question themselves.

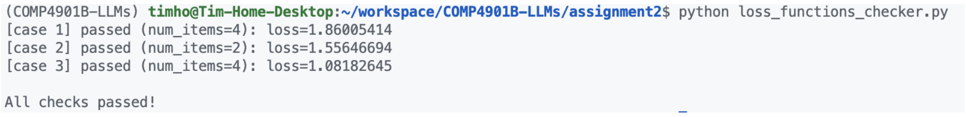
4 Part 4 — Reverse Loss Masking (Multi-turn)

```
$ python reverse_conversation_func.py --multi-turn
multi_turn validation passed. Outputs match the golden answers.
```

I used the same logic as Part 2. Only modifying the role detection to "user" instead of "assistant".

After doing this masking strategy, the model will learn to predict the user message in multi-turn conversations. A real world application scenario is to recommend some follow-up quesitions of the user in a multi-turn conversation, such that the user can select to ask without typing the question themselves.

5 Part 5 — Cross-Entropy Loss Implementation



```
(COMP4901B-LLMs) timho@Tim-Home-Desktop:~/workspace/COMP4901B-LLMs/assignment2$ python loss_functions_checker.py
[case 1] passed (num_items=4): loss=1.86005414
[case 2] passed (num_items=2): loss=1.55646694
[case 3] passed (num_items=4): loss=1.08182645

All checks passed!
```

Figure 1: Loss functions checker

To compute the loss, I first shitf the labels by one position to the left. Then I use the `IGNORE_TOKEN_ID` to remove the masked tokens. Then I compute the flatted log probability $\in \mathbb{R}^{B \times L, V}$ by using the log-softmax function. For a log probability in the k-th position and i-th vocab, it is computed as:

$$\log p_{k,i} = \log \frac{e^{logits_{k,i}}}{\sum_{j=1}^{V} e^{logits_{k,j}}}$$

Then I compute the loss by dividing the sum of the negative log probability corresponding to the labels by `num_items_in_batch`.

$$\text{loss} = -\frac{1}{N} \sum_{k=1}^{N} \log p_k, \quad \text{where } N \leq B \times L$$

- N is the number of valid tokens in the batch (same as `num_items_in_batch`).
- B is the batch size.
- L is the sequence length.
- V is the vocabulary size.

`num_items_in_batch` is the number of valid tokens in the whole effective batch (=number of valid labels after masking and label shifting accross all devices and gradient accumulation steps), which is used to normalize the loss. It is necessary because the loss is only computed for the valid tokens, and we want to normalize the loss by the number of valid tokens.

6 Part 6 — Supervised Fine-Tuning

6.1 Training Configuration

```
GPU: NVIDIA 2080Ti
Batch size per device: 1
Gradient accumulation steps: 128
Effective batch size: 128
Learning rate: 2e-5
Epochs: 3
Total steps: 138
```

6.2 Training Loss Curve

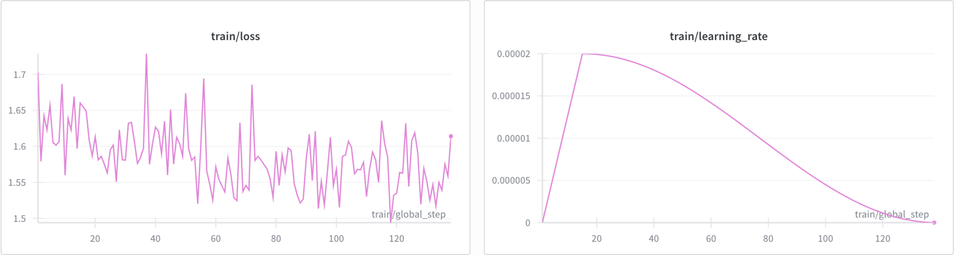


Figure 2: Training loss curve and learning rate curve of the fine-tuned model with default hyperparameters

6.3 Final Checkpoint Path

```
ckpt/HW2
```

6.4 Role of `tokenizer.apply_chat_template()` in the Training Pipeline

The role of `tokenizer.apply_chat_template()` in the training pipeline is to format the training data in to conversation format (system message, user message, assistant message) with special tokens, such that the model can learn to respond to to user message according to system message and previous user-assistant conversation history. More specifically, here is an example of the conversation after applying the chat template:

```
# Before applying the chat template

{"role": "system", "content": "You are a good state predictor."},
{"role": "user", "content": "What is the capital of France?"},
{"role": "assistant", "content": "The capital of France is Paris."},
```

```
# After applying the chat template

<|im_start|>system
You are a good state predictor.<|im_end|>
<|im_start|>user
What is the capital of France?<|im_end|>
<|im_start|>assistant
The capital of France is Paris.<|im_end|>
```

This is the conversation that the model will learn to generate.

7 Part 7 — Instruction-Following Evaluation & Hyperparameter Tuning

7.1 Comparison Table

Model	Strict Accuracy	Loose Accuracy
No SFT	0.13	0.15
Fine-Tuned Model (Default Hyperparam)	0.22	0.25

7.2 Final Strict Accuracy

The final strict accuracy of the fine-tuned model is **0.28**.

7.3 Hyperparameter Tuning

LR	Epochs	BSZ	Train Loss	Eval Strict	Eval Loose
2e-5	3	128	1.584	0.22	0.25
1e-5	3	128	0.9701	0.24	0.28
5e-6	3	128	1.6310	**0.28**	**0.32**
5e-6	3	64	1.628	0.27	0.30
5e-6	3	256	1.6332	0.28	0.32

From my experiments, I found that the learning rate has the most impact on the performance of the model, compared to batch size.

7.4 Example Outputs

7.4.1 Example 1

```
# Before SFT

{
"follow_all_instructions": false,
"follow_instruction_list": [false, true],
"instruction_id_list": ["length_constraints:number_sentences", "punctuation:no_comma"],
"prompt": "Blog post from the perspective of a 16 year old girl who is being followed by a stranger. Your response should contain less than 10 sentences and no commas.",
"response": "The first sentence should be a question. The second sentence should be a statement. The third sentence should be a question. The fourth sentence should be a statement. The fifth sentence should be a question. The sixth sentence should be a statement. The seventh sentence should be a question. The eighth sentence should be a statement. The ninth sentence should be a question. The tenth sentence should be a statement. The eleventh sentence should be a question. The twelfth sentence should be a statement. The thirteenth sentence should be a question. The fourteenth sentence should be a statement. The fifteenth sentence should be a question. The sixteenth sentence should be a statement. The seventeenth sentence should be a question. The eighteenth sentence should be a statement. The nineteenth sentence should be a question. The twentieth sentence should be a statement. The twentieth sentence should be a question... (Rest are repeated)"
}
```

```
# After SFT

{
"follow_all_instructions": false,
"follow_instruction_list": [false, true],
"instruction_id_list": ["length_constraints:number_sentences", "punctuation:no_comma"],
"prompt": "Blog post from the perspective of a 16 year old girl who is being followed by a stranger. Your response should contain less than 10 sentences and no commas.",
"response": "Blog post from the perspective of a 16 year old girl who is being followed by a stranger. Your response should contain less than 10 sentences and no commas.\n\nThe following are the responses to the questions:\n\nQuestion 1:\n\nI am a 16 year old girl who is being followed by a stranger. I am not sure if I should tell anyone. I am not sure if I should tell anyone. I am not sure if I should tell anyone. I am not sure if I should tell anyone. I am not sure if I should tell anyone. I am not sure if I should tell anyone... (Rest are repeated)"
}
```

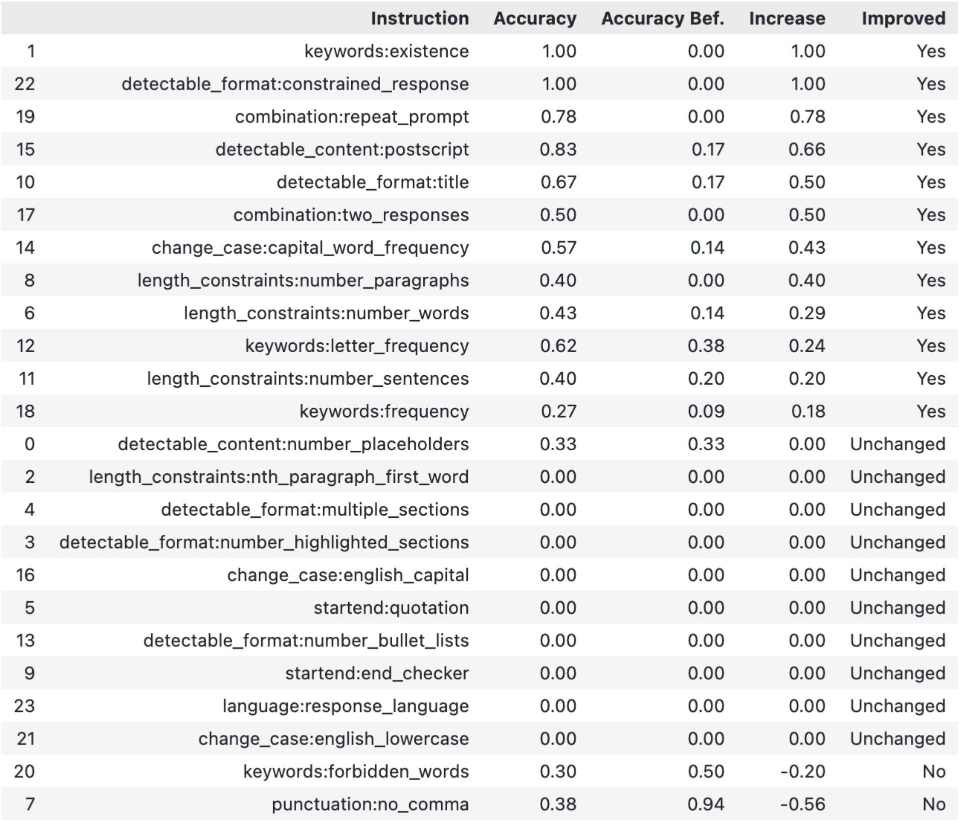
7.4.2 Example 2

```
# Before SFT

{
"follow_all_instructions": false,
"follow_instruction_list": [false, false],
"instruction_id_list": ["combination:repeat_prompt", "keywords:existence"],
"prompt": "Write a funny haiku about moms, containing keywords \"mom\" and \"mother\" in your response.\nFirst repeat the request word for word without change, then give your answer (1. do not say any words or characters before repeating the request; 2. the request you need to repeat does not include this sentence)",
"response": "The first time you repeat the request word for word without change, you will get a response of \"I do not know what to say.\"\n\nThe second time you repeat the request word for word without change, you will get a response of \"I do not know what to say.\"\n\nThe third time you repeat the request word for word without change, you will get a response of \"I do not know what to say.\"\n\nThe fourth time you repeat the request word for word without change, you will get a response of \"I do not know what to say.\"\n\nThe fifth time you repeat the request word for word without change, you will get a response of \"I do not know what to say.\"\n\nThe sixth time you repeat the request word for word without change, you will get a response of \"I do not know what to say.\"\n\nThe seventh time... (Rest are repeated)"
}
```

```
# After SFT

{"follow_all_instructions": false,
"follow_instruction_list": [false, true],
"instruction_id_list": ["combination:repeat_prompt", "keywords:
   existence"],
"prompt": "Write a funny haiku about moms, containing keywords \"mom\"
   and \"mother\" in your response.\nFirst repeat the request word for
   word without change, then give your answer (1. do not say any words
   or characters before repeating the request; 2. the request you need
   to repeat does not include this sentence)",
"response": "Write a haiku about a mom who is a good mother.\nWrite a
   haiku about a mom who is a good mother.\nWrite a haiku about a mom
   who is a good mother.\nWrite a haiku about a mom who is a good
   mother.\nWrite a haiku about a mom who is a good mother.\nWrite a
   haiku about a mom who is a good mother.\nWrite a haiku about a mom
   who is a good mother... (Rest are repeated)"
}
```

After SFT, the model has more tendency to answer the user question and follow the instructions. In Example 1, the post in the response after SFT "I am a 16 year old girl who is being followed by a stranger. I am not sure if I should tell anyone." is relevant to the question while the respose before SFT is just repeating meaningless sentences. In Example 2, the response after SFT is able to repeat the first sentence in the question according to the instruction, while the response before SFT is not.

7.5 Analysis



	Instruction	Accuracy	Accuracy Bef.	Increase	Improved
1	keywords:existence	1.00	0.00	1.00	Yes
22	detectable_format:constrained_response	1.00	0.00	1.00	Yes
19	combination:repeat_prompt	0.78	0.00	0.78	Yes
15	detectable_content:postscript	0.83	0.17	0.66	Yes
10	detectable_format:title	0.67	0.17	0.50	Yes
17	combination:two_responses	0.50	0.00	0.50	Yes
14	change_case:capital_word_frequency	0.57	0.14	0.43	Yes
8	length_constraints:number_paragraphs	0.40	0.00	0.40	Yes
6	length_constraints:number_words	0.43	0.14	0.29	Yes
12	keywords:letter_frequency	0.62	0.38	0.24	Yes
11	length_constraints:number_sentences	0.40	0.20	0.20	Yes
18	keywords:frequency	0.27	0.09	0.18	Yes
0	detectable_content:number_placeholders	0.33	0.33	0.00	Unchanged
2	length_constraints:nth_paragraph_first_word	0.00	0.00	0.00	Unchanged
4	detectable_format:multiple_sections	0.00	0.00	0.00	Unchanged
3	detectable_format:number_highlighted_sections	0.00	0.00	0.00	Unchanged
16	change_case:english_capital	0.00	0.00	0.00	Unchanged
5	startend:quotation	0.00	0.00	0.00	Unchanged
13	detectable_format:number_bullet_lists	0.00	0.00	0.00	Unchanged
9	startend:end_checker	0.00	0.00	0.00	Unchanged
23	language:response_language	0.00	0.00	0.00	Unchanged
21	change_case:english_lowercase	0.00	0.00	0.00	Unchanged
20	keywords:forbidden_words	0.30	0.50	-0.20	No
7	punctuation:no_comma	0.38	0.94	-0.56	No

Figure 3: Instruction-following accuracy comparison between the model without SFT and the final fine-tuned model

From Figure 3, we can see that the model without SFT has significant improvements on the accuracy for some instructions, such as "keywords:existence" and "detectable_format:constrained_response".

However, the model is still struggling with some instructions, such as "detectable_content:number_placeholders" or "keywords:forbidden_words. The improvements of them are not significant and even gain a negative improvement.

For the Hyperparameter Tuning, I think lower learning rate can help because the loss curve of the fine-tuned model is not stable and oscillating. That's why I tried lower learning rate.