Report on HW №2 for «Research in LLM»

Вадим Мельников

26 февраля 2026 г.

1 Task

In this homework we will be trying to teach Qwen 2.5, 1.5B parameters variant, to solve simple chess puzzles. More specifically, given the board and some board description, the model's task is to find a mate in 1 on the board (it is guaranteed that such move exists).

For this task, an open database by Lichess with over 5 million puzzles will be used. Of those puzzles, around 270k are mate-in-1 puzzles of various difficulty. The entirety of the further report basically answers the question "why not use the entire database?".

Stockfish is unanimously the best software for chessboard analysis. For comparison, the latest Stockfish model has around 140M parameters, a tenth of the model we're about to train. It is needless to say that Stockfish is able to solve all of the mate-in-1 chess puzzles in our dataset.

LLMs are famous for not being able to play chess, and this is true even for the latest models. In an attempt to find out whether SOTA LLMs are capable of solving mate-in-1 puzzles, several were fed into Claude Opus 4.6 non-thinking mode in the same format as the LLM we're about to train. The model didn't manage to solve a single puzzle despite having several orders more parameters than Stockfish. That's one of the many reasons why mate-in-1 puzzles were selected rather than the entire Lichess database.

2 Training setup

Training is painfully performed on a single RTX 3060 GPU with 12 Gb of video memory.

3 Training

The training itself is performed using GRPO provided by `unsloth` library in Python, making it easy to actually use GRPO.

3.1 Prompting

Below is a template of the user prompt provided to the LLM:

```
You are a chess engine. Your task is to find the single move that delivers checkmate.

Board representation in FEN format: [Board representation]

Board representation in visual format. Capital letters represent white pieces,
lowercase letters represent black pieces.

[ASCII representation of the border]

White pieces: [List of white pieces in format "(Chess piece name) on (Square)"]

Black pieces: [List of black pieces in format "(Chess piece name) on (Square)"]

Side to move: [White or Black]

Briefly identify which of your pieces can give check and which checking move
leaves the king with no escape in block surrounded with <think> tags. Then
provide the move in UCI format (e.g. e2e4) in <answer> tags.
```

The system prompt from the task description was also used.

Parameter	**Value**
Optimizer	Paged AdamW, 8 bit
Epoch size	5000 steps
Batch size	2
Gradient accumulation steps	1
# generations	8
Prompt length	800 tokens
Max completion length	224 tokens
KL regularization coefficient	0.1

Рис. 1: Hyperparameters

Action	**Reward**
Mating move found	4.0
Checking move found	1.5
Any other legal move	-1.0
Illegal move	-1.0
Gibberish (non-UCI move format or anything else)	-2.0
Correct formatting	0.0
Wrong formatting (no think-answer structure)	-2.0

Рис. 2: Reward Model

3.2 Hyperparameters

The first problem is choosing hyperparameters such that the training is as fast as possible while still successfully working on the training setup. Some of the more hyperparameters are displayed on the figure 1.

Looking at the hyperparameters, effective batch size is 16, which is, to our knowledge, the largest one can use on 3060 GPU without encountering CUDA OOM.

Counterintuitively, one of the important hyperparameters is prompt length – quite a lot of training runs turned out to be ineffective due to the beginning of the prompt being left out, leading to the model not knowing what to do.

3.3 RL

The hardest and most important part of finetuning an LLM with GRPO is choosing the reward model and preventing the model from hacking the rewards. The final reward model is provided on figure 2.

We will now answer a few questions about why the reward model is exactly as provided and not something else.

It may be counterintuitive as to why there is no reward for at least making a legal move, as it seems that the model starts to understand how chess works. However, the model surprisingly managed to find whole two ways of hacking such rewarding strategy:

1. In the vanilla version of having just this kind of reward, the model starts performing extremely passive moves like moving a pawn in front of the king (pawns on f-h ranks are quite often right in front of the king after castling)

2. When attempting to introduce a diversity reward (i.e. penalize the model for producing the same moves over several inputs in the batch), the model begins to either

   (a) start moving more pawns (which is hard to fight because of the batch size)

   (b) picking the first move among the list of legal moves

This kind of reward hack is present both when the legal move reward is positive and negative.

It is important to emphasize that the model has no idea how to play chess. It perfectly understands both the rules (e.g. how the pieces move) and the technicalities like notation, but has no capacity to apply those concepts to an actual game. This kind of behavior goes so far that the non-finetuned model produces impossible moves (one of the funny ones was `h8h9` with no reasoning at all) and illegal moves (e.g. `a1f8`).

That being said, as we're not able to reward the model when it performs a legal move, we have to introduce a reward for checking the opponent to have at least some kind of learning curve, but at the same time not making the model too lazy for it to start reward hacking.

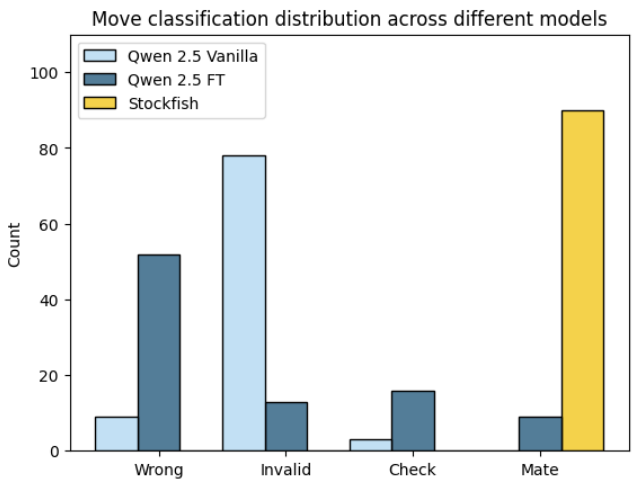

Рис. 3: Comparison between vanilla/finetuned Qwen 2.5, 1.5B and Stockfish

One of the side effects of disabling the legal move reward is enabling the model to use the `<think>` block for actual reasoning: when being rewarded for legal moves, the model tends to use the reasoning block for trivial information like `"I need to mate the opponent."`.

Other attempts to force the model to learn to play chess before trying to solve mate-in-1 puzzles have failed.

4 Results

To compare the resulting model quality with its vanilla variant and Stockfish, a dataset of 90 mate-in-1 puzzles of varying difficulty was selected (difficulty 1–9; difficulty is chosen with respect to ELO rating of the puzzle). The comparison is available on figure 3. Each move that the models produced was assigned one of four categories:

- Wrong – a legal move that does not check or mate the opponent;
- Invalid – an illegal move;
- Check – a move that checks the opponent;
- Mate – a move that mates the opponent.

There are two ways to interpret the results. On one hand, the performance is clearly horrible for the finetuned version of Qwen LLM. On half of the puzzles the model is straight up wrong, and on another ≈ 15 it produces something invalid.

On the other hand, the vanilla Qwen performs significantly worse – it didn't mate a single time and checked the opponent just a couple of times. Given that statistically there are approximately 30-35 legal moves in a chess position on average, those checks might just be accidental. A spike in the illegal moves category also highlights that the vanilla Qwen model basically has no capability of reasoning and doesn't properly follow the system prompt.

The way we interpret it is somewhere in between. It is clear that GRPO worked well in terms of teaching the model to play chess and threatening the opponent. It is also clear that it wasn't enough – while it can be said that the model understands what it is to check the opponent, it heavily struggles with actually mating the opponent.

The results of the work done are not a clear success or failure, but they rather highlight the limits of small language models and pure GRPO approach.

5 Further research

One of the ideas for further investigation is training the model to actually play chess before teaching it to solve mate-in-1 puzzles. This could be done by splitting the training process into two steps:

1. Train the model to perform simple tasks given the chessboard (like listing legal moves, making a move with a diversity reward etc.)
2. Actual training

Another way of improving the model quality is enhancing reasoning. Currently, the model learns what reasoning is and how to reason from scratch. An SFT phase would provide a head start in reasoning as a concept and reasoning about mating specifically. Despite the bigger LLMs not being able to consistently solve mate-in-1 chess puzzles, we're sure that they are able to generate simple mate-in-1 puzzles alongside with reasoning about their solution.

6 Final words

Ладно, тут теперь на русском. К сожалению, каких-либо впечатляющих результатов в результате работы достичь не удалось. Поэтому я принял решение продолжить работать над этой задачей и после дедлайна, и надеюсь, что в период дорешки удастся сдать что-то хоть сколько нибудь достойное.