Automatic task recognition of tracr model

01.03.2025

Krzysztof Gwiazda

Supervised by Leonard Bereska

1. Introduction:

   A project to develop an automated interpreter for neural networks, specifically focusing on Tracr-compiled transformers where we have ground-truth knowledge of their internal mechanisms.

   I transformed tracr model to transformer lens model, performed simple task like reversing and then provide combination of parameters from cache. Lower results are obtained from attention patterns and positional embedding. The task of LLM was to describe inner mechanism basing on data and then predict a task.

   I didn't add input/output pairs in purpose to check whether LLM can predict task without this information.

2. Methods:

   2.1. Tasks

   I performed 3 tasks – sequence reversal, sequence sorting and checking a length of sequence. Each task was run 10 times.

   2.2. Model

   I used ChatGPT o1 model. I checked several models of ChatGPT – o3-mini, 4o, o1-small, GPT-4. Most of them couldn't perform the task. O1 was chosen basing on quality of responses and fact that it's a reasoning model. 4o model sometimes return good results but it was mostly due to guessing (model didn't possess enough information to predict it).

   2.3. Prompt

   I used prompt

   "Analyse this tracr model layer by layer. compare provided data with known patterns from other simple tasks and at the end predict top 1 task it probably preforms \n {attention patterns and positional embeddings}"

   I used several prompts, but it turned out that more complex prompts performed worse. Simple ones give more flexibility to model. Then model could perform its own chain of thoughts.

   2.4. Combination of cache parameters

   Considered data:

   hook_embed, hook_pos_embed, hook_resid_pre, attn.hook_q, attn.hook_k, attn.hook_v, attn.hook_attn_scores, attn.hook_pattern, attn.hook_z, hook_attn_out, hook_resid_mid, mlp.hook_pre, mlp.hook_post, hook_mlp_out, hook_resid_post

   Surprisingly more data didn't occur with better predictions and using all data resulted in 0% accuracy. In this experiment I provided only attention patterns and positional embedding to the LLM. Unfortunately, this is insufficient data for a lot of tasks (for example sorting).

3. Results
   3.1. Accuracy (10 iterations):
   Sequence reversal – 10/10.
   Sequence sorting – 0/10.
   Checking length of sequence – 2/10

   3.2. Sequence sorting task considerations
   Combination of cache parameters is insufficient to predict it correctly. In 9/10 it predicted correctly that the model is preforming permutation. LLM described correctly what this permutation is doing but LLM didn't realise this is sorting task. In one iteration LLM predicted wrongly that the model is parentheses closeness task. To make it more explicit I performed this task second time and added cache data to file (twice as much data). Then run LLM 10 times. 10 times it predicted permutation task.

   3.3. Checking length task also suffer from insufficient number of cache parameters. Similarly to sorting task it well predicted overall concept (aggregation – 8/10).

4. Results after adding input/output pairs:
   4.1. Accuracy (10 iterations):
   Sequence reversal – 10/10.
   Sequence sorting – 10/10.
   Checking length of sequence – 10/10

5. Future work
   5.1. It needs to be checked better combination of cache parameters or determined that it is not possible to recognise these tasks without input/output pairs in the prompt.
   5.2. Check performance on more tasks.

6. References

   6.1. D. Lindner, J. Schulz, A. Kulmizev, J. Uesato, A. Stuhlmüller, "Tracr: Compiled Transformers as a Laboratory for Interpretability", arXiv:2301.05062
   6.2. J. Dunefsky, N. Cammarata, D. Hendrycks, J. Steinhardt, "Transcoders Find Interpretable LLM Feature Circuits", arXiv:2406.11944
   6.3. K. Wang, A. Variengien, A. Conmy, B. Shlegeris, J. Steinhardt "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small", arXiv:2211.00593