Easy Trafo* Mech Int[1]

*Transformer

Stefan Heimersheim, Jonathan Ng
SERIMATS scholars

Abstract

We created a simple web app allowing users to create some standard mechanistic interpretability plots (based on Stefan's explainer) for arbitrary prompts.

The web app computes residual stream patching, attention head output patching, and attention pattern visualizations. The currently-online version allows only 1 & 2 layer models, but in principle the code supports any models supported by TransformerLens.

The code is built on TransformerLens and CircuitsVis for the interpretability tools, with the web page built with Streamlit.

The web app code is available on GitHub. You can run a local version with `streamlit run Home.py`, which also allows you to select arbitrary models (comment them in at the top of the file).

Keywords: Mechanistic interpretability, ML safety

1. Introduction

The motivation for this project was my surprise how easy and relatively accessible it was to start researching Transformer Interpretability. A couple lines of code with TransformerLens gives you a model, you can look at all the weights, and arbitrarily mess with internal activations!
Then I realized it would be really cool to just have a web page where really anyone could try this out with minimal trivial inconveniences.

A secondary motivation was to finally establish the German abbreviation **Trafo** for **Transformer**.

2. Methods

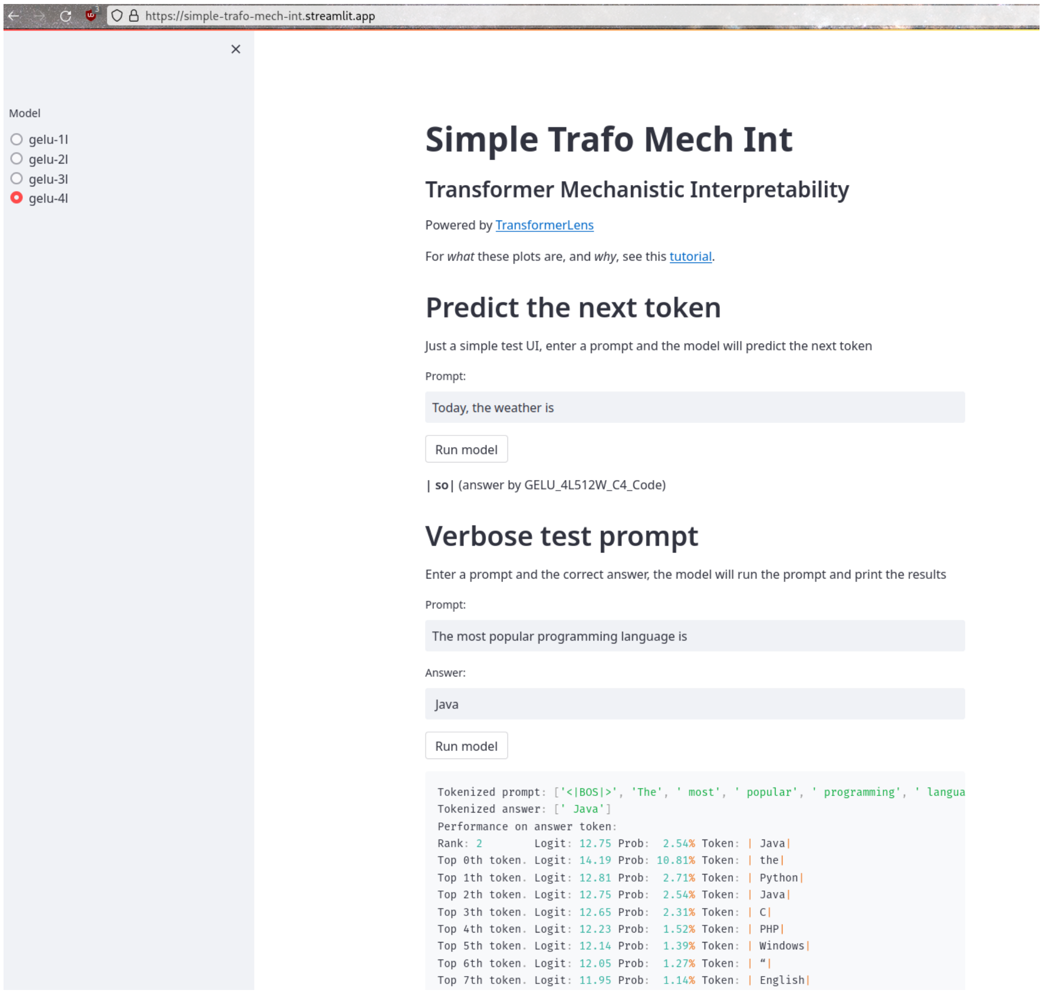
[1] Research conducted at the Apart Research Alignment Jam #4 (Mechanistic Interpretability), 2023 (see https://itch.io/jam/mechint)

Really we just wrote a web app that wraps simple TransformerLens functions. The full code is on GitHub, the patching functions are based on Stefan's tutorial text and notebook.

3. Results

Let's put a couple of screenshots here.

Overview



https://simple-trafo-mech-int.streamlit.app

Model

○ gelu-1l
○ gelu-2l
○ gelu-3l
● gelu-4l

Simple Trafo Mech Int

Transformer Mechanistic Interpretability

Powered by TransformerLens

For *what* these plots are, and *why*, see this tutorial.

Predict the next token

Just a simple test UI, enter a prompt and the model will predict the next token

Prompt:

Today, the weather is

Run model

| **so** | (answer by GELU_4L512W_C4_Code)

Verbose test prompt

Enter a prompt and the correct answer, the model will run the prompt and print the results

Prompt:

The most popular programming language is

Answer:

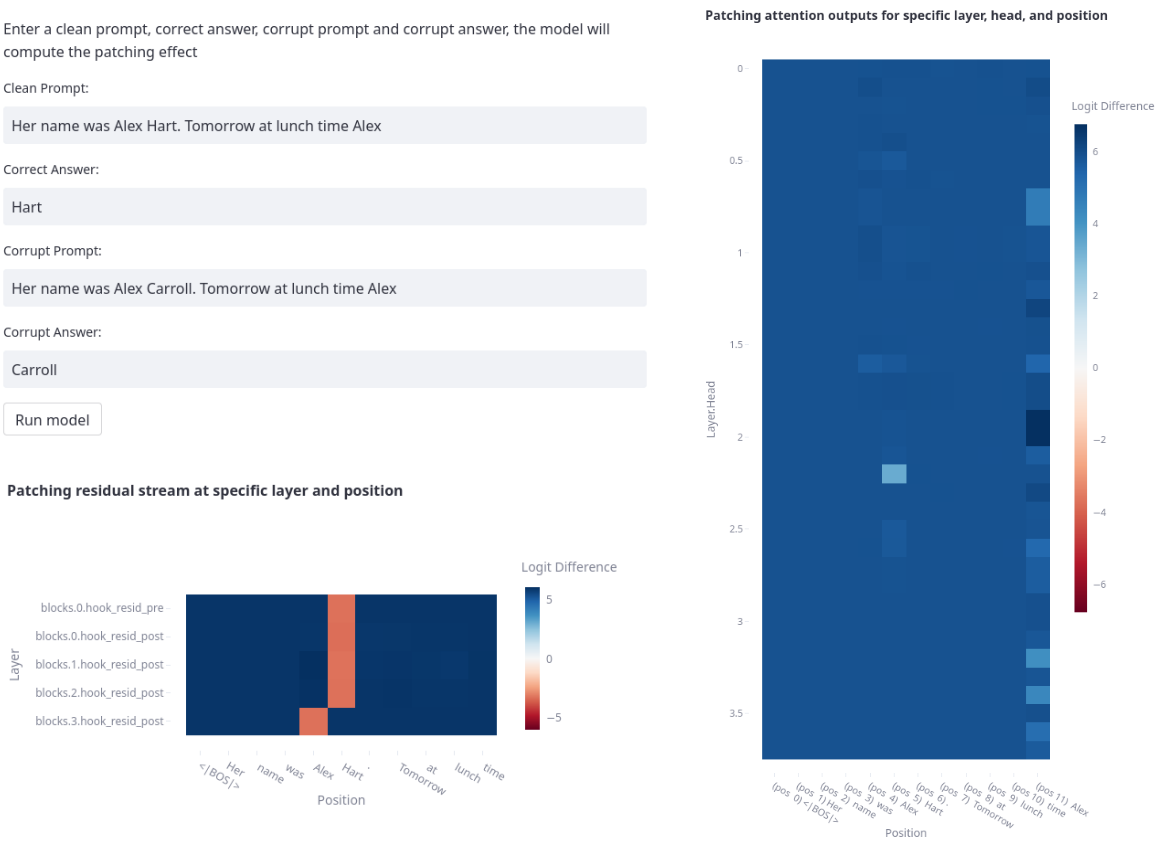
Java

Run model

```
Tokenized prompt: ['<|BOS|>', 'The', ' most', ' popular', ' programming', ' langua
Tokenized answer: [' Java']
Performance on answer token:
Rank: 2        Logit: 12.75 Prob:  2.54% Token: | Java|
Top 0th token. Logit: 14.19 Prob: 10.81% Token: | the|
Top 1th token. Logit: 12.81 Prob:  2.71% Token: | Python|
Top 2th token. Logit: 12.75 Prob:  2.54% Token: | Java|
Top 3th token. Logit: 12.65 Prob:  2.31% Token: | C|
Top 4th token. Logit: 12.23 Prob:  1.52% Token: | PHP|
Top 5th token. Logit: 12.14 Prob:  1.39% Token: | Windows|
Top 6th token. Logit: 12.05 Prob:  1.27% Token: | “|
Top 7th token. Logit: 11.95 Prob:  1.14% Token: | English|
```

The website with the model selector and simple prompt testing tools.

Residual stream and Attention head output patching



Enter a clean prompt, correct answer, corrupt prompt and corrupt answer, the model will compute the patching effect

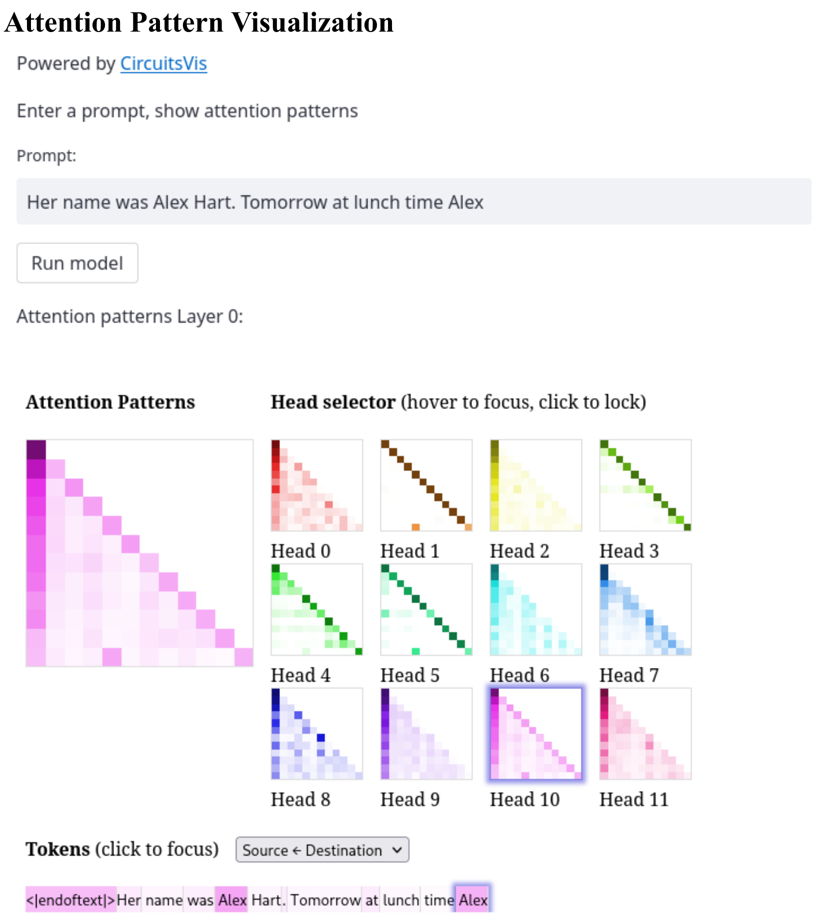
Clean Prompt: Her name was Alex Hart. Tomorrow at lunch time Alex

Correct Answer: Hart

Corrupt Prompt: Her name was Alex Carroll. Tomorrow at lunch time Alex

Corrupt Answer: Carroll

Residual stream patch with different last name, attention head result patch with different first name.



Attention Pattern Visualization

Powered by CircuitsVis

Enter a prompt, show attention patterns

Prompt:

Her name was Alex Hart. Tomorrow at lunch time Alex

Attention patterns Layer 0:

Tokens (click to focus) Source ← Destination

<|endoftext|>Her name was Alex Hart. Tomorrow at lunch time Alex

Attention Pattern visualization of GPT2-small. GPT2 runs locally but not on Streamlit due to storage limits.

4. Discussion and Conclusion

Discuss your results: These aren't research results, just a cool tool :)

We had to restrict the online app to 1 and 2 layer models, and it will also crash when too many users use it. The maximum number of users is approximately 1.

5. References

1. TransformerLens
2. CircuitsVis
3. AMFTC
4. Streamlit
5. Stefan's tutorial text and notebook