Exercise Sheet 1

November 5, 2025

▼ Table of Contents

Practical Notes

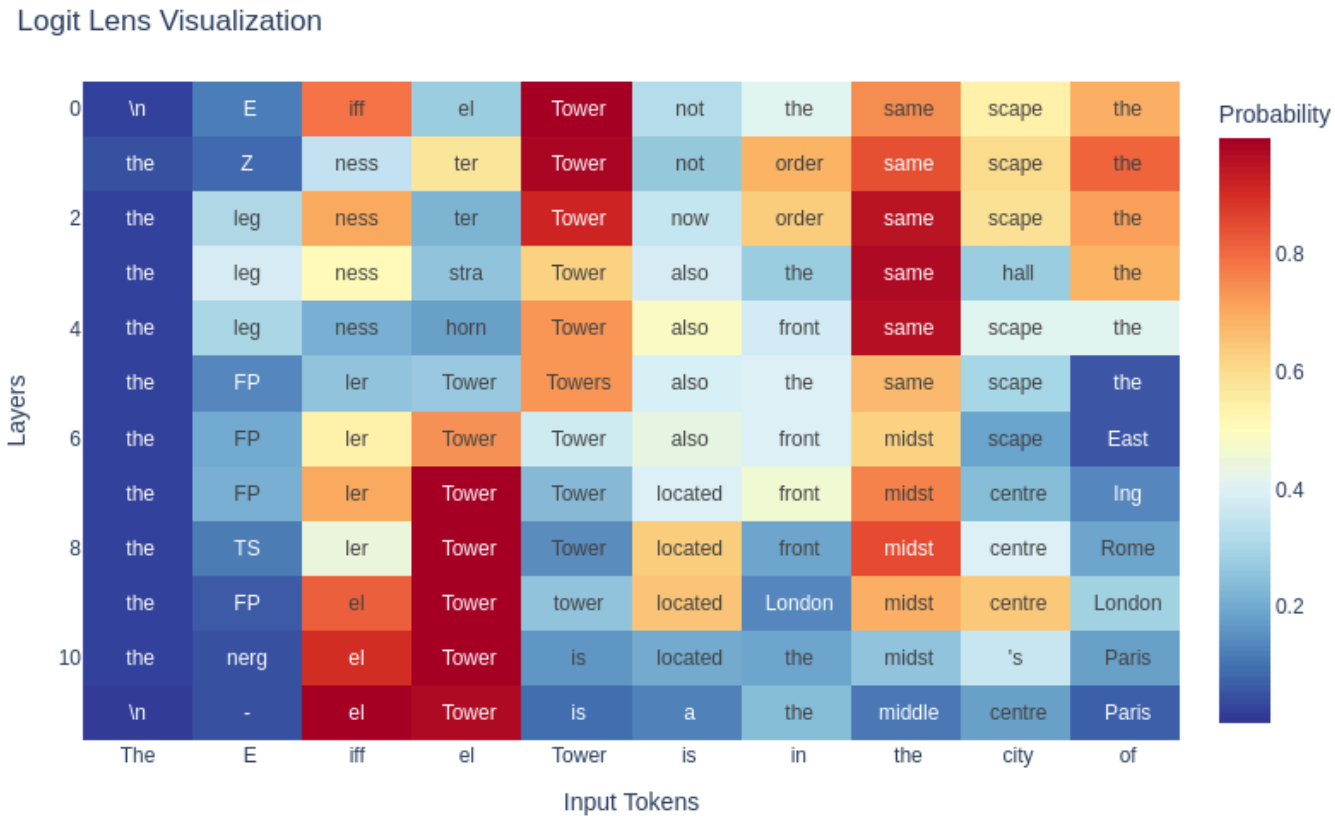
- Let's set November 17th as the deadline. This gives you the weekend to work on it and splits the two sheets approximately evenly across the three-week period.
- Use whatever tools work best for you—Google Colab, your university cluster, local setup, etc.
- These exercises require looking things up. Documentation, implementation details, examples online—not everything will be handed to you.
- Some experiments might not work perfectly on your first try. That's completely normal and part of the process.
- Do experiments! Yes, I had specific ideas when designing this sheet, but there's so much to discover! Document whatever you find interesting—there's no single "right answer" here.

Part 1: Getting Started with LogitLens

Last week, we started with LogitLens—a natural starting point for exploratory analysis of what language models "know" at different layers.

Exercise 1.1: Implement LogitLens

Task: Make LogitLens work. There are libraries that offer this functionality, but it's also not too difficult to implement from scratch. Whatever you do: make sure you understand what you're doing. You should be able to plot the top predictions at each layer ℓ when applying the unembedding matrix. This lets you visualize which tokens the model predicts at different layers. An example visualization looks like this:



Visualization Taken from nnsight

In this visualization, each row represents a layer, and you can see how the model's predictions change as information flows through the network. Early layers might predict generic or incorrect tokens, while later layers converge on the correct answer.

Note for from-scratch implementations: Don't forget the final layer normalization! Many decoder-only transformers apply LayerNorm or RMSNorm right before the unembedding. When implementing LogitLens, you should apply this same normalization to intermediate hidden states before projecting with the unembedding matrix. See the blog post's "Note on notation/implementation" for details.

Verification: Compare your results to examples you can find online to make sure it's working correctly. Start with a well-documented model like GPT-2 on simple English

prompts where you can sanity-check the predictions.

Exercise 1.2: LogitLens on Multilingual Models

Task: Make your LogitLens work with a "real-world" multilingual decoder-only model. I personally recommend "CohereForAI/aya-expanse-8b", as it is a multilingual decoder-only model with a standard architecture, but you are free to choose another model. Keep in mind, however, that the BLOOM family behaves strangely. You'll probably need to make small adjustments because layer/component naming conventions differ across models. Moreover, you'll need to find a setup where you don't run out of CUDA memory.

Part 2: Cross-Lingual Exploration

Now that we have LogitLens working on a multilingual model, let's think about what we could investigate. We have a tool that lets us peek inside the model at different layers—so what questions might be interesting to ask? One natural starting point: How does a multilingual model handle the same factual knowledge across different languages? Does it process "The Eiffel Tower is in the city of Paris" differently when the prompt is in English versus Arabic versus Chinese?

Exercise 2.1: Brainstorming

Before actually starting with experiments, let's think systematically about what we could vary. Take a moment to brainstorm (just briefly!): When considering different languages, what "dimensions of multilinguality" could we explore? What kinds of prompts could we test? Don't feel obligated to test everything, just think about what seems interesting.

Exercise 2.2: Experiments!

Task: Take a prompt like "The Eiffel Tower is in the city of" and apply LogitLens across different languages. Translate the prompt into different languages—and don't just stick to one language family (like Italian, French, and Spanish)! We live in a world of LLMs; you can get translated variants for (more or less) free. Try languages with different scripts.

Document everything you find interesting or surprising. Summarize your findings so we can discuss them in class. There's no single "right answer" here—different observations can lead to different insights. Your documentation will be the basis for our class discussion, so focus on what genuinely interests or puzzles you.

Part 3: A Puzzle About English

If you've been running the experiments above, you might have noticed something curious: even when the prompt is in another language, English tokens sometimes appear in the middle layers before the model settles on the correct answer in the target language.

What's more, the English token is often the English solution to the task we had in the other language. For example, when translating from French to Chinese—say, "tête" → "头" (head)—you might observe the English token "head".

Exercise 3.1: Experiment Design

Can you design an experiment yourself to investigate this systematically?

Think about:

- What do you have to control for? (Hint: How can you be sure English isn't appearing just because of the prompt or task structure?)
- What would you measure? (Presence of English tokens? At which layers? With what probability?)
- How would you set up the comparison? (What languages? What tasks?)

There's no single "right answer"—different approaches can reveal different insights.

If you need help: Don't get stuck! If you're unsure what I'm aiming for or if you feel lost, just text me. I didn't want to give answers away "for free" because it would be tooooo tempting to just look them up, but I definitely don't want you to waste time being confused.

Exercise 3.2: Experiments!

Task: Implement your idea from Exercise 3.1 and run the experiments. Your analysis should probably quantify: (a) how often and where English tokens appear and (b) how often and where language-specific tokens appear. The experiment should be quantitative, so you should have some "objective" results—not just cherry-picked examples. Track patterns systematically across multiple test cases. You can cheaply generate data using your favorite LLM. Or you can use an existing dataset.

Part 4: Reflection and Looking Ahead

In this exercise sheet, you've worked extensively with LogitLens and hopefully discovered interesting patterns. As I mentioned earlier, these observations can be foundations for further hypotheses and testing. However, LogitLens only tells us *what* the model would predict after a layer, not *how* we got there. We've also discussed Activation Patching and

Sparse Autoencoders. These are tools I've given you, and you're welcome to use them—but you don't have to.

Exercise 4.1: Designing a Causal Test

In Part 3, you likely observed that certain language-specific tokens emerge at particular layers. Does this mean language information is added at these layers? Are there possible alternatives? Could language-specific information be present earlier than when it shows up in LogitLens predictions?

Design an activation patching experiment to test when language information becomes causally important for generating language-specific output (but don't implement it!!!).

- How should the clean run and the corrupted run look?
- Do you use different prompts? Do you introduce noise?
- What would you patch, and at which layers?
- How would you measure whether your patching "worked"?

Exercise 4.2: Other Tests

Are there other ideas you have for what one could investigate? Maybe with linear probing, SAEs, or Patchscopes? This is completely optional—only if you think you have a cool idea you'd like to explore or sketch out.

Exercise 4.3: Feedback

It was genuinely hard for me to estimate how difficult these exercises will be for you. Everyone comes with different backgrounds—some of you might have extensive experience with Transformers and find the implementation straightforward, while others might be encountering these concepts for the first time. If you like, please give me feedback on whether it was easy, much too difficult, or whatever you experienced. Your feedback will have an effect on the next exercise sheet (if you and I are fast enough in reacting). Any other feedback is also appreciated.