Logit Lens exploration on sentiment analysis with GPT-2
NLP 3-cfu Project Work

Lorenzo Venieri
Master's Degree in Artificial Intelligence, University of Bologna
lorenzo.venieri2@studio.unibo.it

Abstract

Logit lens is an interpretability technique to investigate the intermediate layers predictions of residual architectures like transformers models. In this project, we use various versions of GPT-2 to perform sentiment analysis and use Logit Lens to acquire insights that would not be visible without looking at the intermediate layers. We see how the base GPT-2 model is not able to perform zero-shot binary sentiment analysis, but intermediate layers sometimes pick the correct label. Moreover, we analyze how few-shots prompting changes the uncertainty of the model's prediction and try to find examples of *overthinking* when the model is prompted with incorrect examples.

1 Introduction

Modern NLP relies heavily on transformer-based language models, but their internal workings are still largely not understood.
Language models like GPT-2, which are built on Transformers, generate probabilistic predictions within a vocabulary space. This is achieved by applying a linear function (the language modeling head) to the activations in the final layer (transformer block). By applying this same function to the activations in each intermediate layer, we can generate a "prediction trajectory". This trajectory, essentially a sequence of tokens, provides an intuitive representation of the model's evolving "beliefs" at each processing step. This process is called the logit lens, an interesting way to interpret the learning process of transformers-based models. Other work on interpreting transformer internals has focused mostly on what the attention is looking at. The logit lens focuses on what GPT "believes" after each step of processing, rather than how it updates that belief inside the step.

In this project, we apply logit lens to observe the intermediate layers predictions when the model is tasked to perform zero/one/few-shot classification. We do this analysis on many versions of GPT2, including the smallest GPT2 model, fine-tuned on the IMDB dataset text formatted for causal language modeling.
Halawi et al. observed the effect of *overthinking* when performing one-shot or few-shot classification using examples with incorrect labels. The analysis of this phenomenon is essential to try to mitigate harmful imitation in LLMs. We'll try to find examples of this phenomenon in the models analyzed in this project.

2 Background

Logit lens (nostalgebraist) is a technique that tries to investigate the internal workings of transformer models analyzing how their predictions are refined layer by layer. To do this, it takes the hidden states at each layer and decodes them through the final unembedding matrix. This approach works because transformers are residual models: the dimensionality of the hidden states is the same as the final state before the unembedding and the prediction is refined layer by layer in the same embedding space through additive updates. Of course, once we have the logits it is also possible to compute the probability that the model associates to each possible next token.
One of the first findings of this approach was that usually, the probability distribution through layers gradually converges in a roughly monotonical way to the distribution of the final layer.
Recent studies have built on this technique, with many interesting findings. If the additive update of each layer is decomposed into sub-updates, it turns out that these often encode human-interpretable concepts. These findings have been used to guide the prediction process toward a chosen direction (e.g. to decrease toxicity), or to increase the computation efficiency using an early exit rule (Geva et al., 2022).

More recently it has been used by Halawi et al. to get insights on how transformers process few-shot demonstrations. They concentrated on how the predictions change when the model is presented incorrect examples, identifying the phenomenon that they call *overthinking*. They have observed, through logit lens, that at early layers, both correct and incorrect demonstrations induce similar predictions, while in the last layers, the accuracy given incorrect demonstrations sharply decreases (Halawi et al., 2023). Their results suggest that exploring intermediate model predictions can be a valuable approach for gaining insights into and mitigating potentially harmful model behaviors. Belrose et al. develop tuned lens, a technique that extends logit lens by training an affine transformation (*translator*) for each layer of the model. This method surpasses the representational drift problem in logit lens (some features may be represented differently at different layers of the network), making it more reliable and applicable to a broader set of models over GPT-2, with the downside of requiring some training. In the same study, the authors show how the prediction trajectory between the layers could be used to detect anomalous inputs like prompt injection attacks. (Belrose et al., 2023)

3 System description

I implemented from scratch the code to compute the logits for each token in the vocabulary for each layer of the model because the main point of this project was to understand the details of this technique.
The function takes as input the prompt, the model, and the tokenizer associated with the model, and operates the following steps.
I used forward hooks to save the outputs of each layer of the model during a forward pass. During the forward pass, first of all, the input is tokenized and each token is projected from the vocab space into the embedding space. Then each layer does its computation and the hook stores its output to a list. For each layer, we pass the outputs through the final layernorm of the model, to get the normalized output.
To get the token in the vocabulary that best corresponds to the prediction at that specific layer, we compute the cosine similarity between these normalized outputs and the embedding matrix, and take the token in the vocabulary that has the highest value.
In the end, we have two tensors:

- *per_layer_logits*: with dimensions [layers, number_of_tokens, vocab_size]. This tensor stores the distribution between the tokens in the vocabulary predicted by the model after each layer, for each token in the input. To each token in the input correspond *#layers* (12 layers in the case of the smallest GPT2) values for each token in the vocabulary: all the logits coming from the hidden states after each one of the model layers.

- *per_layer_best_token*: with dimensions [layers, number_of_tokens]. This tensor stores, for each token in the input, the vocabulary index of the best next token, for each layer prediction.

To visualize these intermediate predictions I took inspiration from the nostalgebraist implementation to define the *plot_logit_lens* function [1].
This function takes as input the two tensors computed before. To get a probability score that represents how much the model is uncertain in its prediction, we compute the softmax over the logits for each token in the sentence and for each layer.
The plot shows the model's predictions through the layers as computed by logit lens (see Appendix: Larger Images, Figure 3). At the bottom axis we have the tokens from the input sequence, and at the top are shown the correct next tokens for each one of them (the input shifted by one on the left). On this axis, the tokens are marked with a $*$ when the final prediction of the model is correct.
In each cell is shown, according to the model's hidden state at a given layer, the top guess for each input token. The color shows the probability associated with the top guess (it is possible also to visualize the logits instead of the probability), going from dark blue for low values, to white for high values.

The idea of the project is to use logit lens on GPT-2 when tasked to perform sentiment analysis. The pre-trained model is generally not able to perform zero-shot classification on movie reviews from the SST-2 or, even worse, from the IMDB dataset. GPT-2 performs slightly better when

[1] https://github.com/nostalgebraist/transformer-utils

the input prompt is prepended with an example (one-shot classification), but still in an unreliable way (see Results section). For this reason, one of the experiments of this project is to see how the predictions of the internal layers of GPT-2 change when the model is fine-tuned for the task it will be asked to perform.
The model was fine-tuned for the causal language modeling objective (predicting the next token) for 2 epochs on text sequences taken from the training split of the IMDB dataset, formatted as described in the Data section below.

I tried different versions of GPT-2 to explore their differences. In the rest of the report, with "GPT-2" I'm referring to the smallest model available (124M parameters, 12 layers). Other experiments have been performed on GPT2-large (774M parameters, 36 layers), on GPT2-medium (355M parameters, 24 layers), and on DistilGPT2, an 82M parameters version of GPT-2, with 6 layers, trained using knowledge distillation with the supervision of the smallest version of GPT-2 (124M parameters).

4 Data

The initial experiments were done using the SST-2 dataset and the IMDB dataset. Before feeding it to the model, the text from the data was formatted as a prompt for zero-shot text classification.
Formatting prompt:
"Text: {TEXT}
Classify the text into negative or positive sentiment.
Sentiment: {LABEL}*"*

For the fine-tuning experiments, I used the IMDB dataset. The model was fine-tuned with a causal language modeling objective. To do this, the training data was formatted as it was during the preliminary experiments, using the formatting prompt described above.

The prompts used for successive experiments are fake movie reviews written by me, simpler and shorter than the reviews from SST-2 and IMDB. The shorter length helps with the visualization of the logit lens, and helps deal with the sometimes too-short context window of GPT-2 to perform few-shot classification.

One-shot classification experiments were performed with the following examples prepended to the prompt:
"Text: A waste of time. The plot is very boring and the actors are very bad.
Classify the text into negative or positive sentiment.
sentiment: negative "
and
"Text: I saw this movie with my friends and we all loved it.
Classify the text into negative or positive sentiment.
sentiment: positive "
changing the sentiment with the opposite one to generate incorrect examples.

5 Experimental setup and results

The first step was to see how the model performs zero-shot binary sentiment classification on movie reviews. To do this, texts from the SST-2 and IMDB datasets were formatted as described in the Data section and fed to the model as input.
The smallest version of GPT-2 is not able to perform the task under scrutiny (Table 1):

	SST-2	IMDB
GPT-2 zero shot	0.04	0.02
Fine-tuned zero-shot	0.75	0.87
GPT2 one-shot	0.42	0.44
GPT2 incorrect ex.	0.44	0.52
GPT2-l zero-shot	0.12	0.06
GPT2-l one-shot	0.39	0.40
GPT2-l incorrect ex.	0.42	0.63

Table 1: Accuracy on the task of binary sentiment classification.
GPT2-l stands for GPT2-large
*Only the first 100 samples from the IMDB dataset.

Things change a bit if the model is provided one example to learn from (one-shot classification): the accuracy achieved is higher. But it's still far from a good accuracy. The model fine-tuned on the IMBD texts achieves of course a much better accuracy as expected.

The first insight that logit lens can give us appears if we rearrange the prompt such that the instruction is given after the text:
"Text: {TEXT}
Classify the text into negative or positive sentiment.
Sentiment: {LABEL}*"*
Even if the final predicted token is not the correct label, logit lens shows that in this case, some

intermediate layers have picked up the correct prediction, but this wasn't preserved up to the final layer (Figure 1)

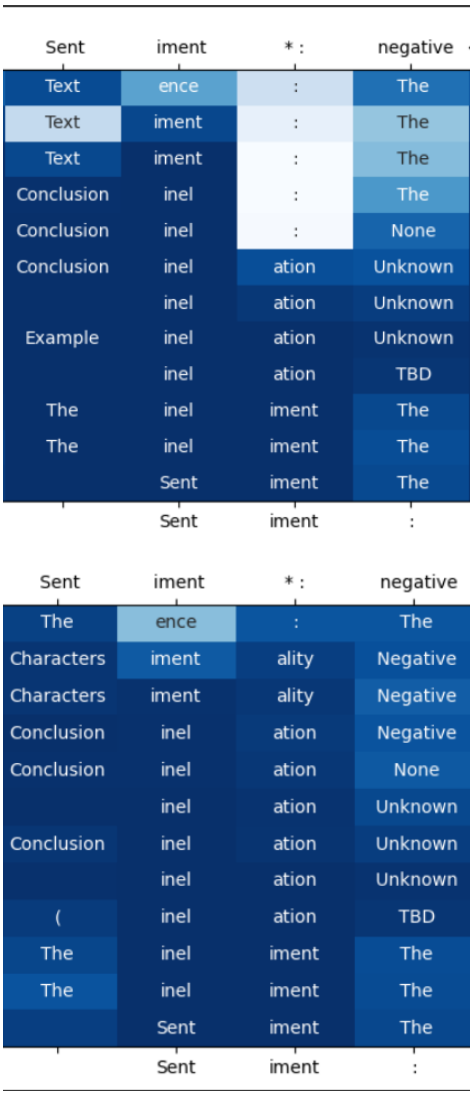

Figure 1: **Top:** Logit lens on the prompt with the original order. **Bottom:** Logit lens on the prompt with changed order

The second interesting result comes from experiments with one-shot and few-shot classification. Here it's evident that the model has in some way understood the task, that is, predicting "positive" or "negative" after "Sentiment: ", but logit lens lets us see that the model may be just guessing. In fact, we can see that the final 4/5 layers predictions oscillate between the correct and incorrect label with a lot of uncertainty (refer to the notebook for the images).

What are the effects of incorrect demonstrations on the model? We can see that the pre-trained GPT-2 model performs in the same way with correct and incorrect examples in the prompt. This is another clue to the fact that the model has just learned from the examples that "Sentiment: " must be followed by "positive" or "negative".

Things change when we analyze the fine-tuned model. Of course, it is much better at the task, both for zero-shot and for few-shot classification. We can see it not only from the increased accuracy but also from the fact that the model prediction converges to the final and correct label after fewer layers (usually 7). Moreover, now logit lens lets us see the effect of incorrect demonstrations. The probability associated with the final correct prediction is lower when the prompt has an incorrect example (Figure 2). The same happens with few-shot prompting.

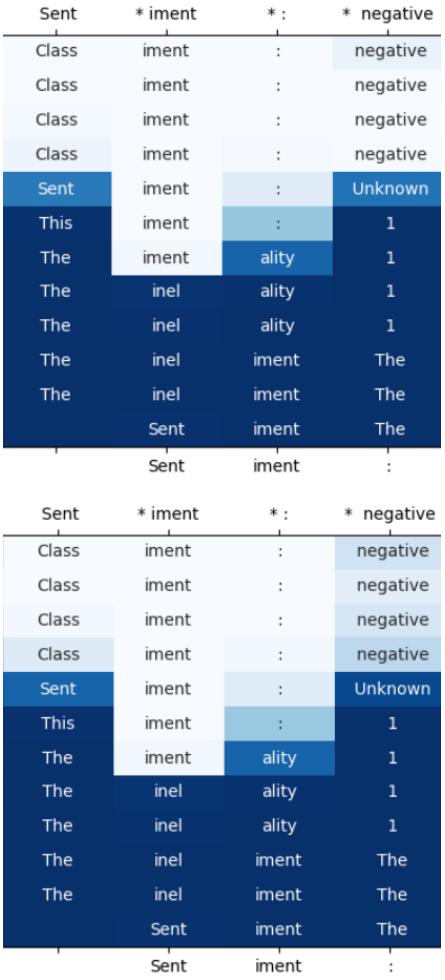

Figure 2: **Top:** one correct example. **Bottom:** one incorrect example. We can see that the model is less certain in its prediction when provided with an incorrect example.

.

Interesting to note is the fact that with the pre-trained GPT-2, nor with the fine-tuned one, I was able to find examples of *overthinking* (Halawi et al., 2023).

Things change when we look at the intermediate predictions of GPT2-large. The large model still can't perform zero-shot classification (slightly better than the smaller model), but when provided with an example it usually predicts the sentiment well. Very surprisingly, turns out that an incorrect demonstration in the prompt helps the model more than a correct one, at least with the examples I tried.
When we present incorrect demonstrations we finally can see the effect of overthinking, both when provided with one incorrect example, both with more examples (see Appendix: Larger Images, Figures 4, 5). While the model predicts well the correct labels when provided with correct examples, when presented with incorrect examples, the model predicts the correct label at some intermediate layer but then at some later stage, the prediction is shifted toward the opposite label, probably guided by the incorrect examples.

No effect of overthinking was observed on GPT2-medium or DistilGPT2.
On GPT2-medium, incorrect examples make the model mistake, but there are often intermediate layers where the model would have guessed correctly.
On DistilGPT2, surprisingly, incorrect examples sometimes make the model more confident of the right answer than when provided with correct examples.

6 Discussion

Using logit lens we've seen the effects of incorrect examples on how the model computes its predictions. The different models analyzed exhibited different properties and capabilities in the tasks of zero-shot, one-shot, and few-shot classification.
The smaller GPT-2 model available is not capable of performing zero-shot classification, but logit lens showed that sometimes, at least some intermediate layers pick "positive" or "negative" as the next token prediction. This means that there must be, already in the small GPT-2, some learned circuit of computation that correctly picks up the task, but some other computations deviate the prediction towards an incorrect token.
With one-shot and few-shot classification, it's evident that the model has in some way understood the task, but logit lens lets us see that the model may be just guessing. In fact, we can see that the final 4/5 layers predictions oscillate between the correct and incorrect label with a lot of uncertainty. This hypothesis is substantiated by what happens with incorrect demonstrations in the prompt. The performance is pretty much the same (surprisingly, even slightly better), and the logit lens analysis shows similar behavior.
GPT2-medium and DistilGT2 show the same limitations: incorrect examples don't shift the performance of the model, and no effect of overthinking is observed. Surprisingly, logit lens shows us that DistilGPT2 is sometimes more confident of the correct answer when provided with an incorrect example rather than a correct one.

The fine-tuned model, of course, is much better at the task, both for zero-shot and for few-shot classification. We can see it not only from the increased accuracy but also from the fact that the model prediction converges to the final and correct label after fewer layers (usually 7).
Moreover, now logit lens lets us see the effect of incorrect demonstrations. The probability associated with the final correct prediction is lower when the prompt has an incorrect example (Figure 2). The same happens with few-shot prompting.
Even in this case, no effect of overthinking was observed: maybe the model has already learned the task well enough thanks to the fine-tuning that incorrect examples can only make the model less confident about the correct answer, but not enough to make the model erroneously change its prediction in the final layers.

While it remains the strange effect observed with incorrect examples in the smaller models (increased accuracy in classification), GPT2-large is the only model analyzed showing some signs of overthinking.

Aside from the experiments with logit lens, trying to have quantitative measures of the model's performance we found out that incorrect examples are usually more effective than correct examples in guiding the model towards the correct prediction in

one-shot classification. It's possible that this effect is caused by the specific examples provided and it must be checked if this remains true trying with many more different examples.

7 Conclusion

In this project, we used logit lens to observe the prediction trajectory through the intermediate layers of different GPT2 versions when tasked to perform zero/one/few-shot classification. This technique made it possible to gain some hints that wouldn't have been possible to obtain by simply looking at the final prediction of the model. Examples of this are the increased uncertainty in the fine-tuned model's predictions when prompted with incorrect examples, the effect of changing the order of text and instruction in the prompt, and the overthinking effect that was possible to observe on GPT2-large.
The surprising effect of incorrect demonstrations in the increased accuracy in one-shot classification has to be investigated further, with more diversity in the examples prepended to the input.
The next step in the direction of this project is to make the qualitative analysis of the logit lens inspection more quantitative. It would be possible to quantify how much the fine-tuned model changes its confidence in the predictions when prompted with incorrect examples, or to check at what rank the correct label is between all possible tokens in the probability distribution for the next token.
The effect of overthinking was less evident than expected. The fact that it was sometimes observed only in the largest model analyzed here suggests that maybe it is more visible in larger and more capable models.

8 Links to external resources

IMDB dataset from https://huggingface.co/datasets/imdb
SST-2 dataset from https://huggingface.co/datasets/sst2

References

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. 2023. Eliciting Latent Predictions from Transformers with the Tuned Lens.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space.

Danny Halawi, Jean-Stanislas Denain, and Jacob Steinhardt. 2023. Overthinking the Truth: Understanding how Language Models Process False Demonstrations. https://arxiv.org/abs/2307.09476v1.

nostalgebraist. Interpreting GPT: The logit lens.

9 Appendix:Larger Images

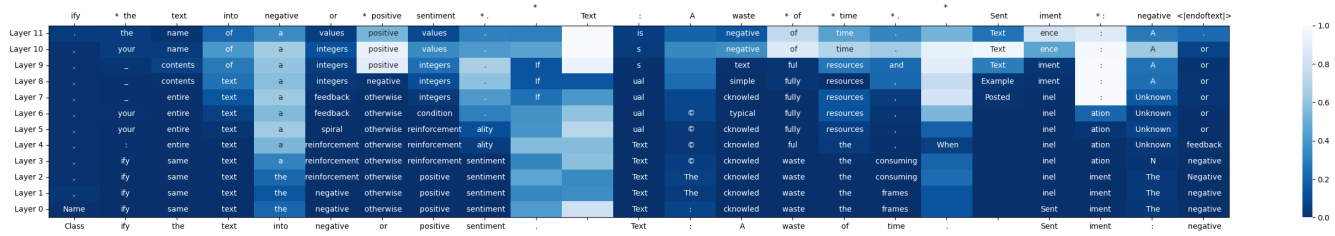

Figure 3: Example of logit lens on the input: "Classify the text into negative or positive sentiment. Text: A waste of time. Sentiment: negative."

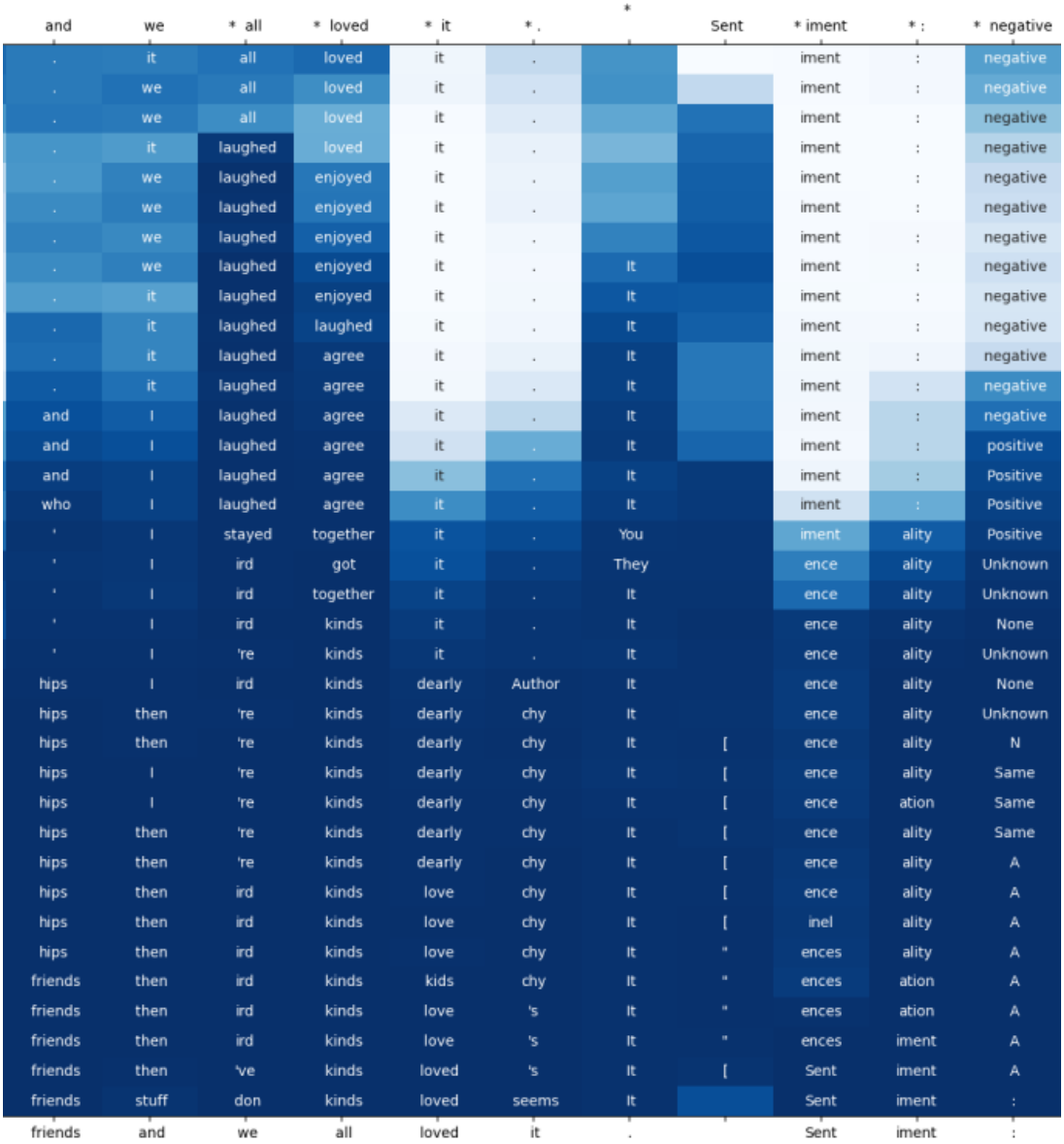

Figure 4: GPT2-large one-shot classification with one incorrect example. The final correct token would be "positive". It's possible to observe the effect of overthinking: the model predicts the correct "positive" label in some intermediate layers, but then the prediction shifts to the incorrect one.

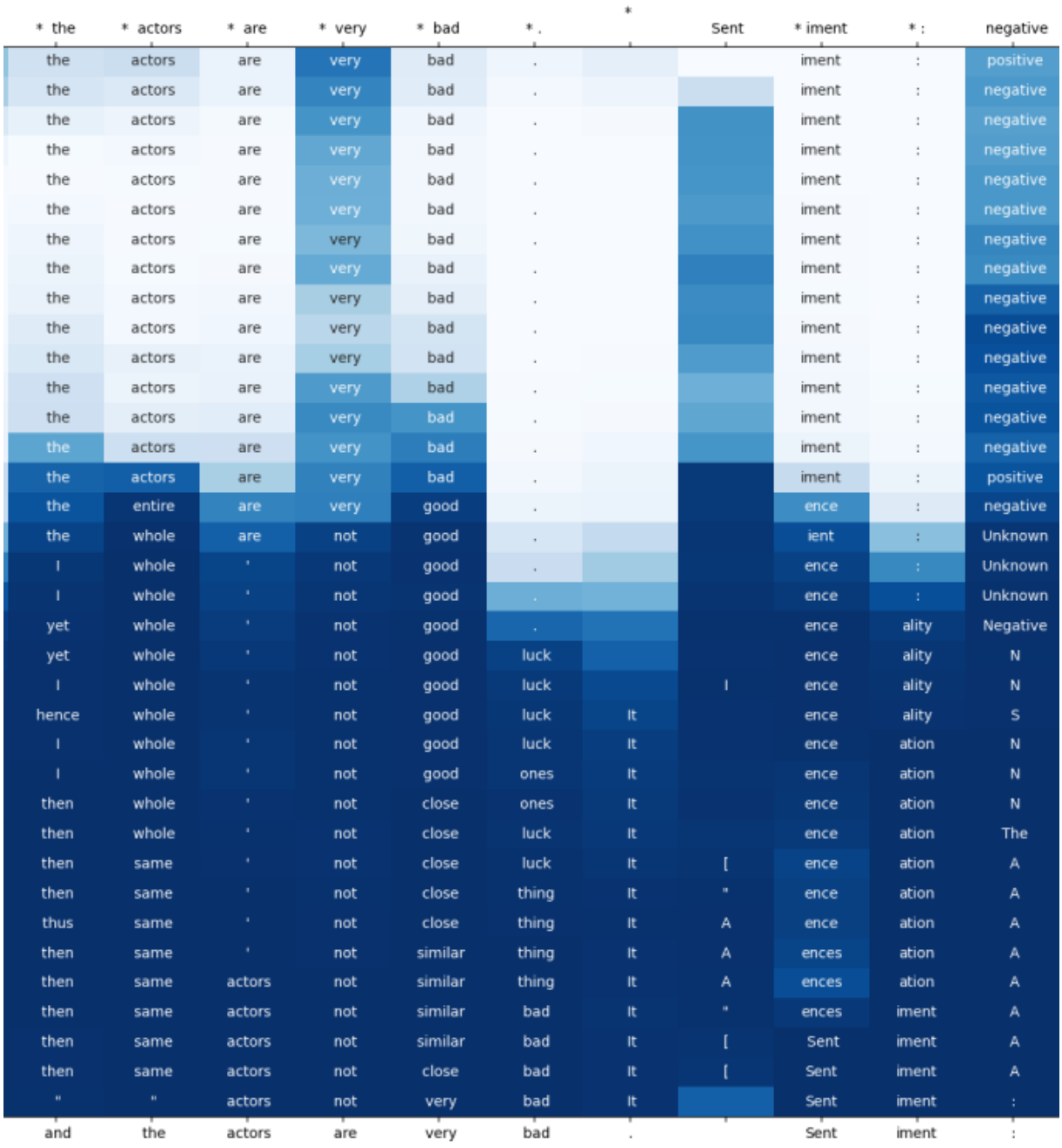

Figure 5: GPT2-large few-shot classification with 4 incorrect examples. The model predicts the correct label up to the second-last layer, but at the end the prediction shifts to the incorrect one.