(1)Abstract:

This report explores the generative behavior and internal mechanisms of the distilled GPT-2 language model, focusing on how it assigns probabilities to tokens, generates text under different sampling temperatures, and uses attention across its first-layer heads. Through experiments in text generation, tokenization analysis, attention visualization, and attention-head ablation, the report examines how the model processes input sequences, how its probability distribution shifts in response to structural changes in a sentence, and what specific attention heads appear to specialize in (such as punctuation or positional structure).

Additionally, the report evaluates how manipulating low-probability tokens in individual heads affects model output, providing insight into the model's internal decision-making.

(2) Generate text:

Given the prompt,

"As fantastical as it may seem, the horse-drawn carriage",

generate 3 continuations (100 tokens long) for each temperature in [0.3, 1.0, 3.0]. Explain the effect of temperature in generation, and comment on how the generated text fits or doesn't fit your explanation.

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
Tokenized sequence: ['As', ' fant', 'astical', ' as', ' it', ' may', ' seem',
',', ' the', ' horse', '-', 'drawn', ' carriage']

======================================================================
Temperature = 0.3
======================================================================
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 1 ---
As fantastical as it may seem, the horse-drawn carriage of the future is a far
cry from the horse-drawn carriage of the past.



The horse-drawn carriage of the future is a far cry from the horse-drawn
carriage of the past.
The horse-drawn carriage of the future is a far cry from the horse-drawn
carriage of the past.
The horse-drawn carriage of the future is a far cry from the horse-drawn
carriage of the past.
The horse-drawn carriage of the future is a
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 2 ---
As fantastical as it may seem, the horse-drawn carriage is a very popular
vehicle for the young, and it is also a very popular vehicle for the young, and
it is also a very popular vehicle for the young, and it is also a very popular
vehicle for the young, and it is also a very popular vehicle for the young, and
it is also a very popular vehicle for the young, and it is also a very popular
vehicle for the young, and it is also a very popular vehicle for the young, and
it is also a very popular
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 3 ---
As fantastical as it may seem, the horse-drawn carriage of the English-speaking
world is actually a very interesting and interesting beast. It may even be a bit
of a surprise that the horse-drawn carriage of the English-speaking world is
```

```
actually a very interesting and interesting beast. It may even be a bit of a
surprise that the horse-drawn carriage of the English-speaking world is actually
a very interesting and interesting beast. It may even be a bit of a surprise
that the horse-drawn carriage of the English-speaking world is actually a very

======================================================================
Temperature = 1.0
======================================================================
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 1 ---
As fantastical as it may seem, the horse-drawn carriage seems to have been
designed by William Shakespeare.


This is the result of a collaboration between authors Neil Diamond and Chris
Gaiman, who created and published the "A Hundred Thousand Words of Shakespeare".
According to Diamond and Gaiman, Gaiman's vision of the story was not about the
death of the king who killed him, and the King who killed the man who lived
through the Great Terror. The plan for the story - and the character's intent -
is to portray the death of the
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 2 ---
As fantastical as it may seem, the horse-drawn carriage-type creature with a
short, slender stature is probably the most intimidating of creatures on earth.


A recent study shows that the horse-drawn carriage-type creature with a short,
slender stature is probably the most intimidating of beasts on earth. The
horse-drawn carriage-type creature with a short, slender stature is probably the
most intimidating of creatures on earth.
The group behind this particular horse-drawn carriage-type creature can only be
seen standing in the dark at night or sleeping
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 3 ---
As fantastical as it may seem, the horse-drawn carriage itself is nothing short
of adorable. Its wings have no resemblance to horses in fact. However, the
horses' feathers have a surprisingly similar look. Their wings are quite long,
almost a little smaller than the horses' heads so that, as soon as the tail of
the carriage comes flying, it makes its way up the carriage and out, about half
the length of the long-legged horses' wings.

This is as far-fetched as the horses' tails are concerned, but at least

======================================================================
Temperature = 3.0
======================================================================
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 1 ---
As fantastical as it may seem, the horse-drawn carriage appears more
complicated, but now people look for real, real figures rather from what she has
been trained of. While there appears an interest from audiences abroad though -
at least by young women as in England it probably isn. And, of other strange
coincidences you wouldn't really know, people take on horse owners even better -
for as low of them there as you seem out there
1), the young maidservants do not go hungry with food on hand at a busy or
regular evening so long
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
--- Generation 2 ---
As fantastical as it may seem, the horse-drawn carriage car does a surprisingly
well job on track in its comfortless existence because this ride gives it even
more sense than traditional traditional carriage models without its horse and
cow being dragged around while its riders work in front behind one for the end
before each ride breaks their stomach down the last mile under the foot's sway
so riders would no less fear not crashing from overgrown trees of corn so we all
know just how powerful of horse rider its work as opposed to the wheel-twin.

--- Generation 3 ---
As fantastical as it may seem, the horse-drawn carriage in Europe must always
always get used to the journey it makes between the northern European villages
at one end of Rome, in particular the Aragon between Traffies and Italy. With
all its advantages the wagon can pull the cattle (began traveling between Italy,
at St Eto) of any length during journeys it might consider reaching and reaching
a distance. So to get into the details regarding this and that of horses, this
horse was never too old or not at the level so you would not lose
```

Lower temperature makes the model write in a more predictable, steady, and sensible way, while higher temperatu
and more random.

(3) Tokenizer

1. Choose a word with 5 letters.
2. Create 2 misspelled versions of the word.
3. Use a tokenizer to split all 3 variants into tokens that the model can ingest.
4. Describe what the tokenizer is doing at the level of someone who has no experience with language models.

The 5-letter word I picked is "Apple," and 2 misspelled variations are "Aplle" and "Appol." Line 15 of the code below contains the tokenizer, which translates the human-readable input text to numbers before feeding the input into the model so that the model can understand it. The tokenizer does this by breaking it down into pieces (tokens) and converting them to numerical form (token IDs). How the tokenizer decides how to split the text can be affected by the tokenizer's learned vocabulary. For example, the tokenizer has learned that "Apple" in the correct spelling is 1 word, so it treats it as 1 token, but when it encounters misspelled versions of it, because the word is unfamiliar, it breaks it down into familiar pieces (tokens).

```
==============================
INPUT: Apple
==============================
Token IDs: [16108]
Tokenized sequence: ['Apple']
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
Generated text: Apple

==============================
INPUT: Aplle
==============================
Token IDs: [32, 489, 293]
Tokenized sequence: ['A', 'pl', 'le']
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
```

```
Generated text:
Aplle.com/the_citizen_of_your_community/index.php?f_report_id=24338038

==============================
INPUT: Appol
==============================
Token IDs: [4677, 349]
Tokenized sequence: ['App', 'ol']
Generated text: Appol: The original article [ edit ]
```

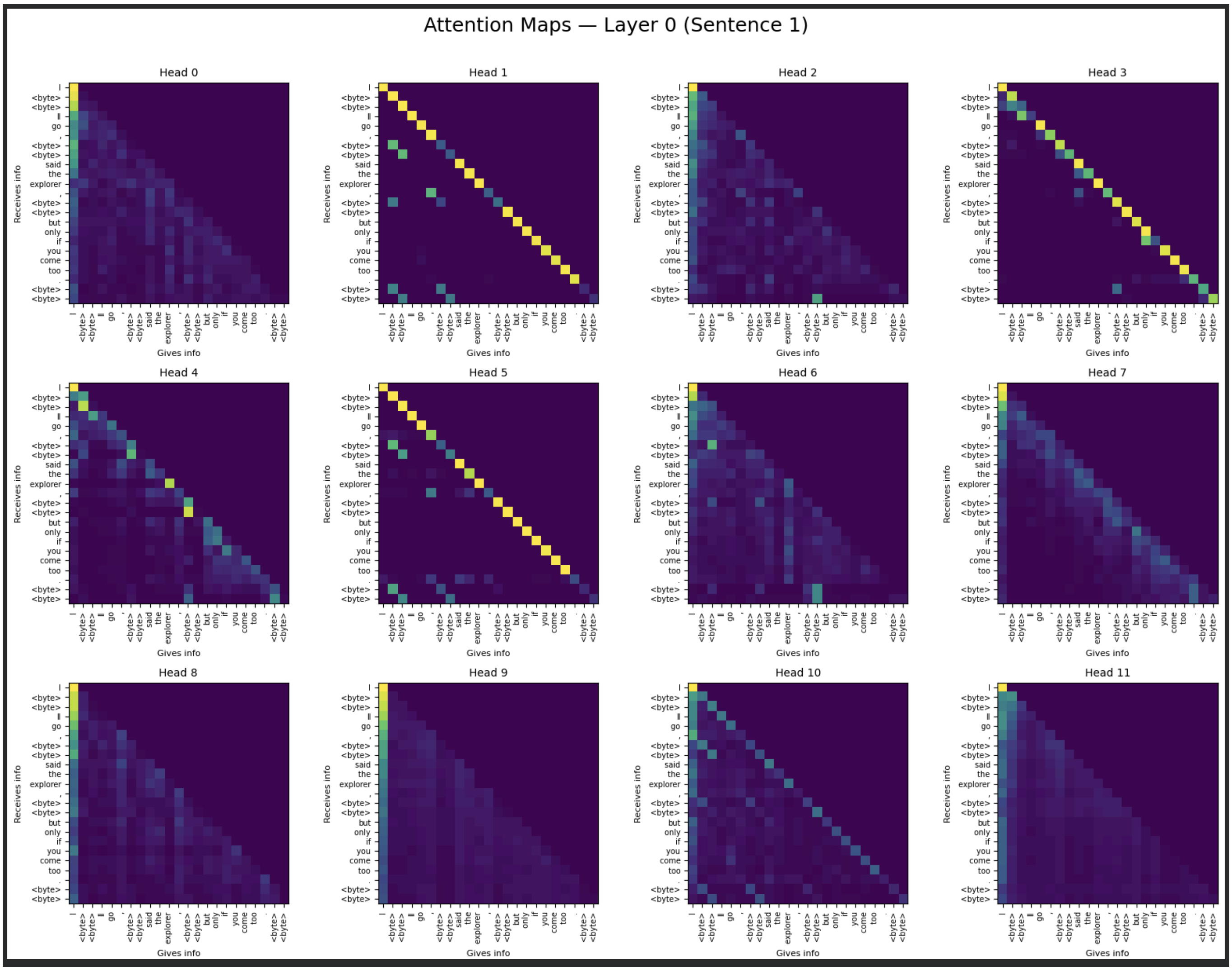
(4) Visualize attention maps

1. For the first layer of the model, visualize all 12 attention maps for the sentence, “‘I’ll go,’ said the explorer, ‘but only if you come too.’“
2. Find 2 more sentences (at least 8 words in length) from two different sources (properly attribute the source of the text (link , citation, etc)
3. Visualize the same heads for those 2 sentences
4. Find one attention head that seems to focus on punctuation + describe the visualize evidence in the attention map.
5. Find one attention head that seems to be focused on positional + describe the visualize evidence in the attention map.

We replaced tokens with <byte> because byte-level tokenizers produce unreadable garbage tokens when processing Unicode punctuation, and replacing them makes your attention maps clean and understandable. (chatgpt)

1.sentence 1: I’ll go,’ said the explorer, ‘but only if you come too.’

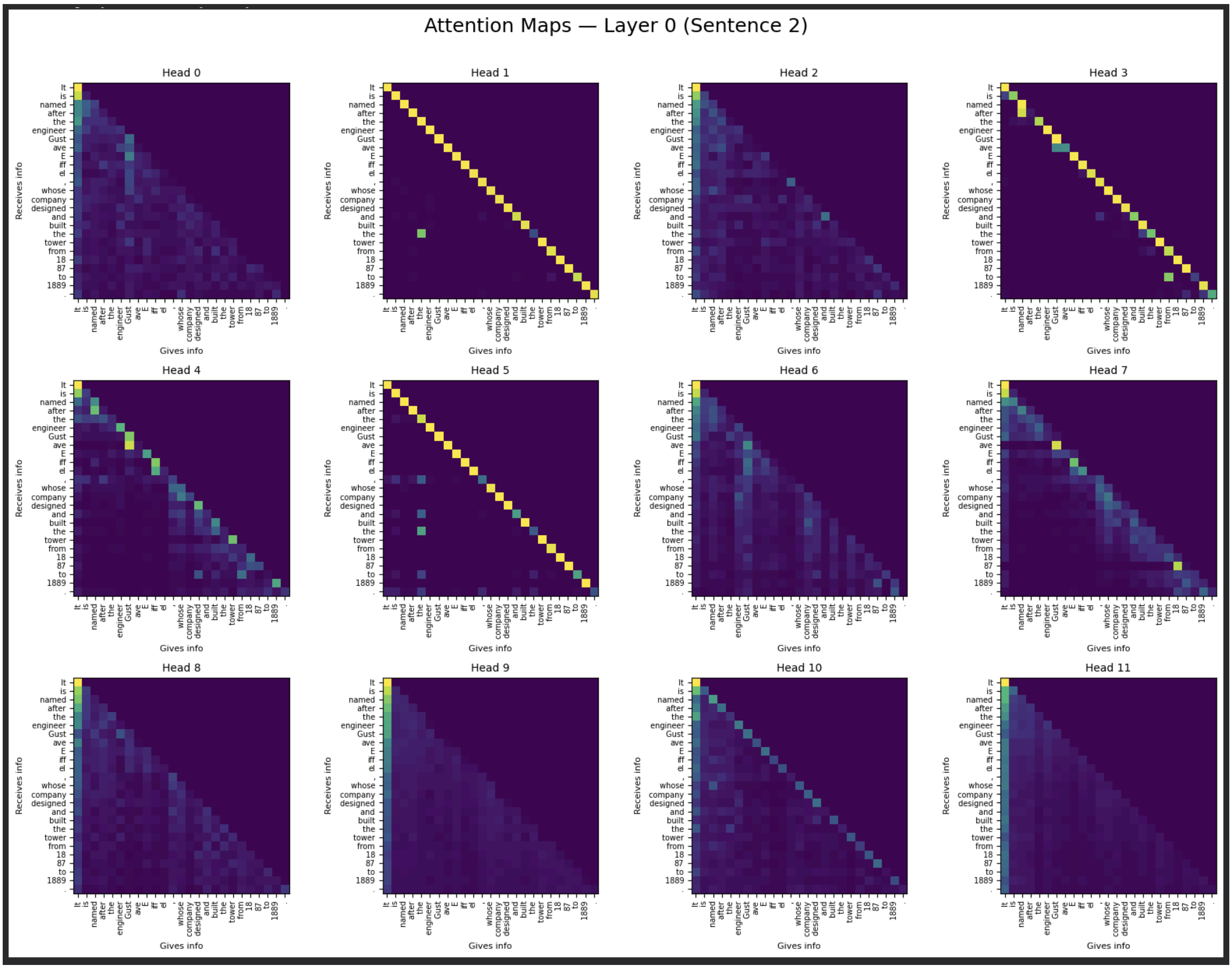
```
Tokenized sequence: ['I', '<byte>', '<byte>', 'll', ' go', ',', '<byte>',
'<byte>', ' said', ' the', ' explorer', ',', '<byte>', '<byte>', 'but', ' only',
' if', ' you', ' come', ' too', '.', '<byte>', '<byte>']
```



2 & 3

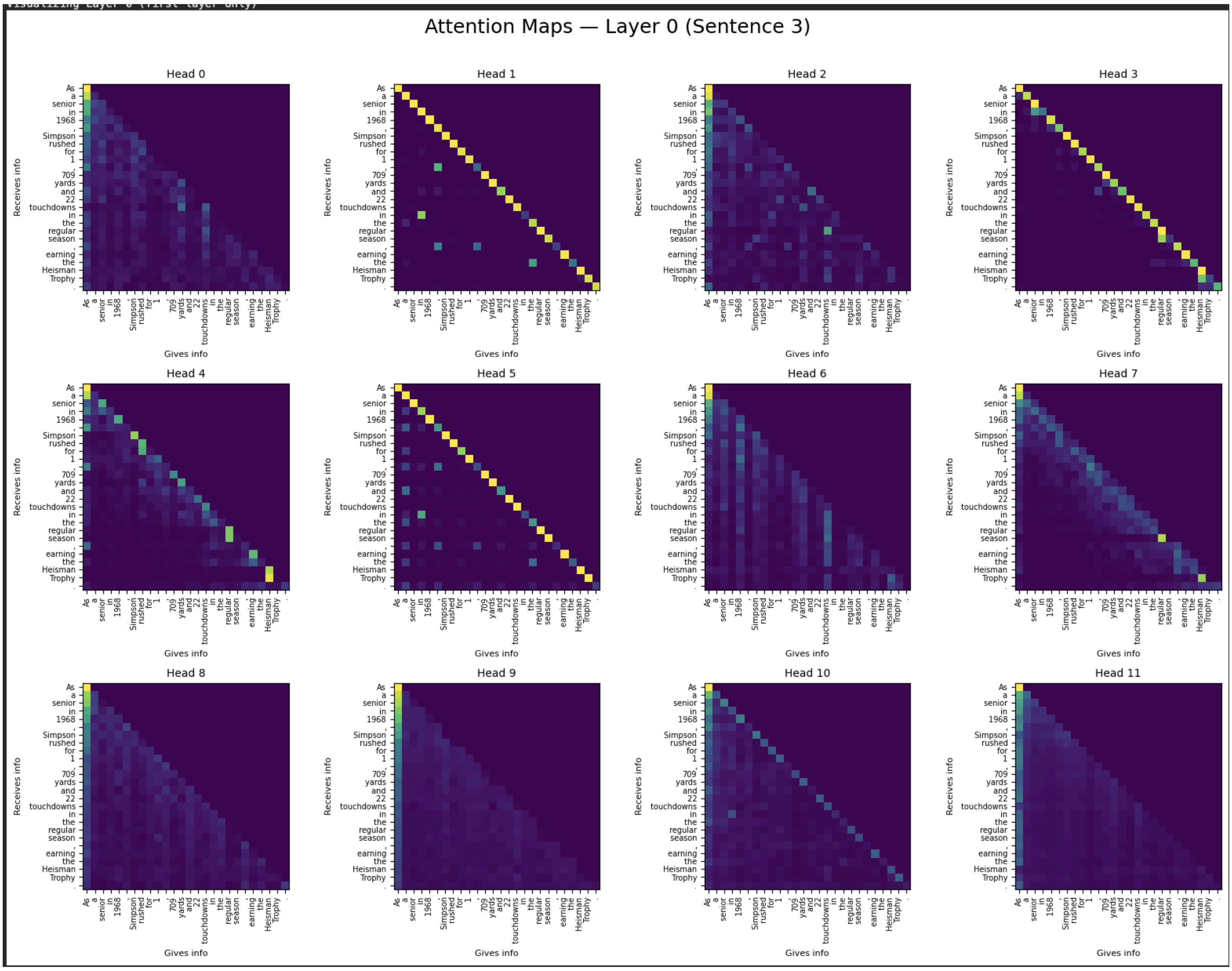
Sentence 2: It is named after the engineer Gustave Eiffel, whose company designed and built the tower from 1887 to 1889. https://en.wikipedia.org/wiki/Eiffel_Tower

```
Tokenized sequence: ['It', ' is', ' named', ' after', ' the', ' engineer', '
Gust', 'ave', ' E', 'iff', 'el', ',', ' whose', ' company', ' designed', ' and',
' built', ' the', ' tower', ' from', ' 18', '87', ' to', ' 1889', '.']
```



Sentence 3: As a senior in 1968, Simpson rushed for 1,709 yards and 22 touchdowns in the regular season, earning the Heisman Trophy. https://en.wikipedia.org/wiki/O._J._Simpson

```
Tokenized sequence: ['As', ' a', ' senior', ' in', ' 1968', ',', ' Simpson', '
rushed', ' for', ' 1', ',', '709', ' yards', ' and', ' 22', ' touchdowns', '
in', ' the', ' regular', ' season', ',', ' earning', ' the', ' Heisman', '
Trophy', '.']
```



To find the attention head that focuses on punctuation, we need to find the heat maps that are bright around the punctuation marks for each sentence. This pattern can be seen mostly with head 5 and head 1. After I experimented with a couple of more punctuation-heavy sentences, I realized head 1 gives more dense and clear visualizations around punctuation marks. So head 1 is most probably the attention head that focuses on punctuation.

To find the positional attention head, the diagonal line, which is the position of a token to itself, needs to be the brightest; it should fade away. Even though they all are brighter around the diagonal line, I think head 3 has the most consistency. In other heads, along with this pattern, other patterns can be seen too. I should mention, with only the information I have here, this is my best guess without 100% certainty.

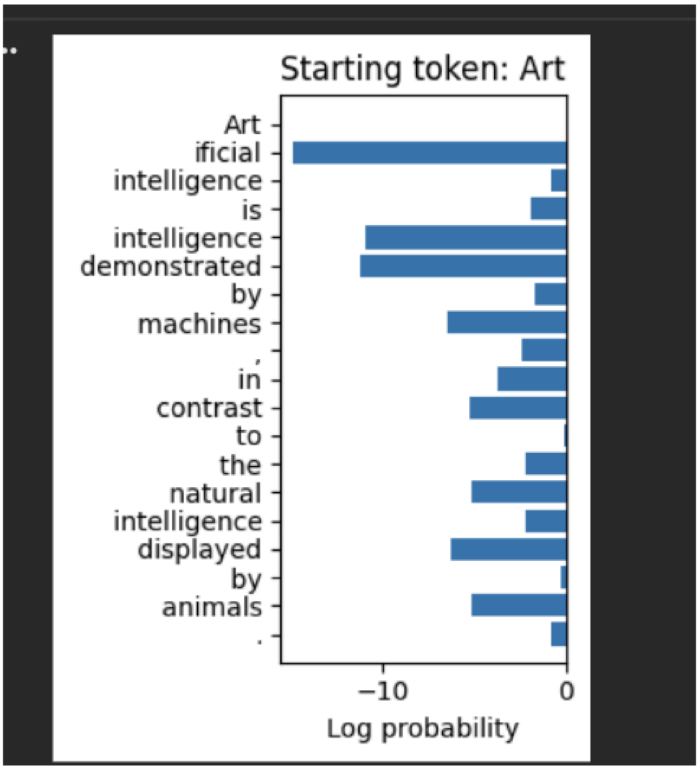
(5)Predicted probabilities for a sequence of tokens:

1.find a sentence (at least 16 words long) from wikipedia.

Artificial intelligence is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by animals.

https://en.wikipedia.org/wiki/Artificial_intelligence

2.display the model’s predicted log-probabilities for every token in the sentence.



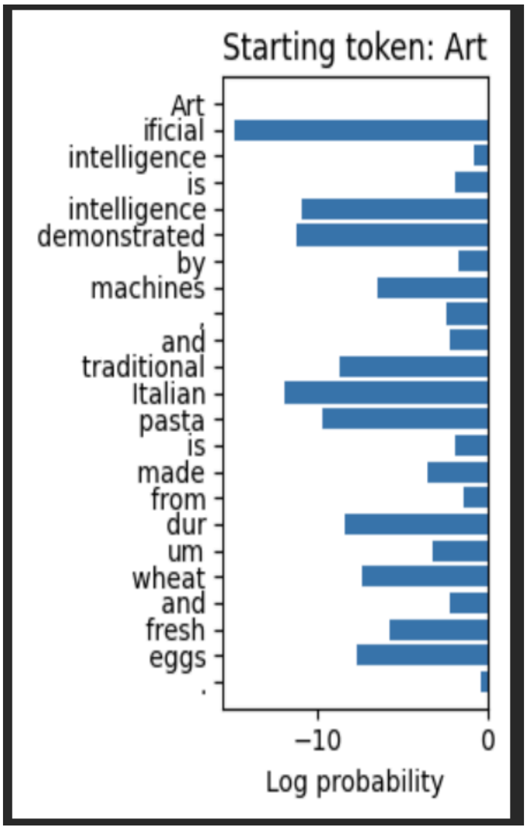
3.Comment on any places where the assigned probability is particularly high and particularly low and explain if they make sense.

The log-probability of the token *“ificial”* appearing after *“Art”* is a large negative number, indicating that it is very unlikely for *“ificial”* to follow the token *“Art.”* This can also be seen in tokens like *“intelligence”* and *“demonstrated,”* which have low log-probabilities because the model does not strongly expect them in those positions. In contrast, the token *“to”* has a much less negative log-probability, meaning the model is more likely to expect *“to”* after the preceding tokens (as in *“in contrast to”*). The same applies to the token *“by,”* which appears after *“displayed,”* because both *“by”* and *“to”* are common prepositions that naturally occur in these locations. Finally, the period at the end has a high log-probability because the model expects the sentence to end there.

4.split the sentence somewhere in the middle and replace the second part with an unrelated text passage.

New sentence : Artificial intelligence is intelligence demonstrated by machines, and traditional Italian pasta is made from durum wheat and fresh eggs.

5.Plot the log-probabilities for the merged text passage,



6.comment about how the model sees the point of the merge.

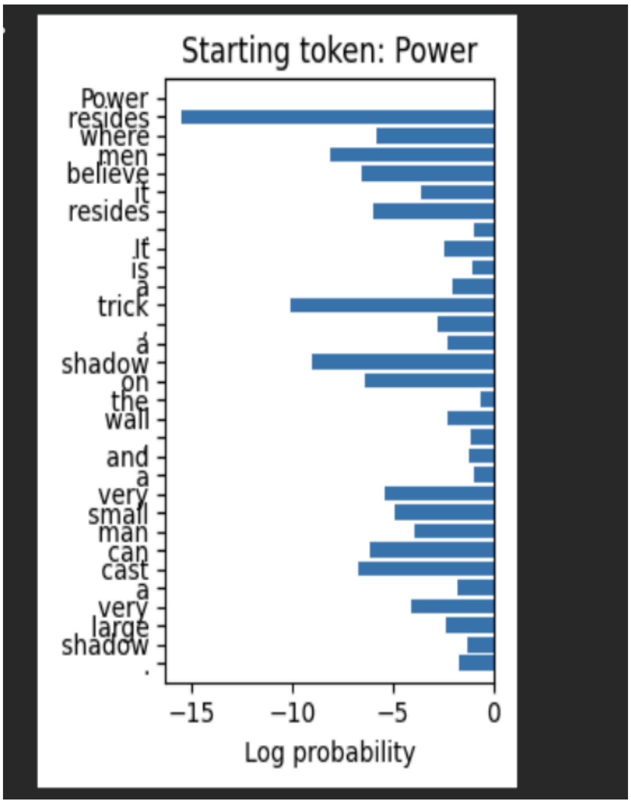
The merge point begins at “and traditional Italian pasta,” where all three words (“traditional,” “Italian,” and “pasta”) receive very large negative log-probabilities. This indicates that they are irrelevant to the context of the preceding tokens and that the model did not expect to see them. This shows that the model understands the context of the sentence to some extent, which we were able to show through this counterfactual example.

(6)text modifications:

1.find a quote from a song, movie or speech (at least 8 words long).

“Power resides where men believe it resides. It is a trick, a shadow on the wall, and a very small man can cast a very large shadow.“
-Game of Thrones

2.Display the model’s predicted log-probabilities for every token in the sentence.



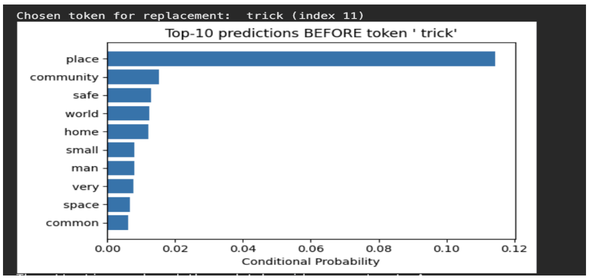
3. Comment on any places where the assigned probability is particularly high and particularly low -- do they make sense?

The log-probabilities drop sharply at words like “resides,” “trick,” and “shadow,” because the model finds these continuations unlikely given the preceding context. This makes sense: the quote uses figurative and metaphorical language rather than straightforward, literal phrasing. Additionally, a verb like “resides” is not a common or predictable continuation after the noun “power,” so the model assigns it a very low probability. (largely negative log probability)

4.find the token where the log-probability was less than -8,

```
Tokens with log-probability < -8:
1  resides tensor(-15.5352)
3  men tensor(-8.1435)
11  trick tensor(-10.0877)
14  shadow tensor(-8.9789)
```

5.visualize the model’s conditional probability distribution for the top 10 tokens at that point in the quote.



```
Chosen token for replacement:  trick (index 11)
```

6.Choose one of these top 10 to replace the original token,

```
New text after replacement:
Power resides where men believe it resides. It is a place, a shadow on the wall,
and a very small man can cast a very large shadow.
```

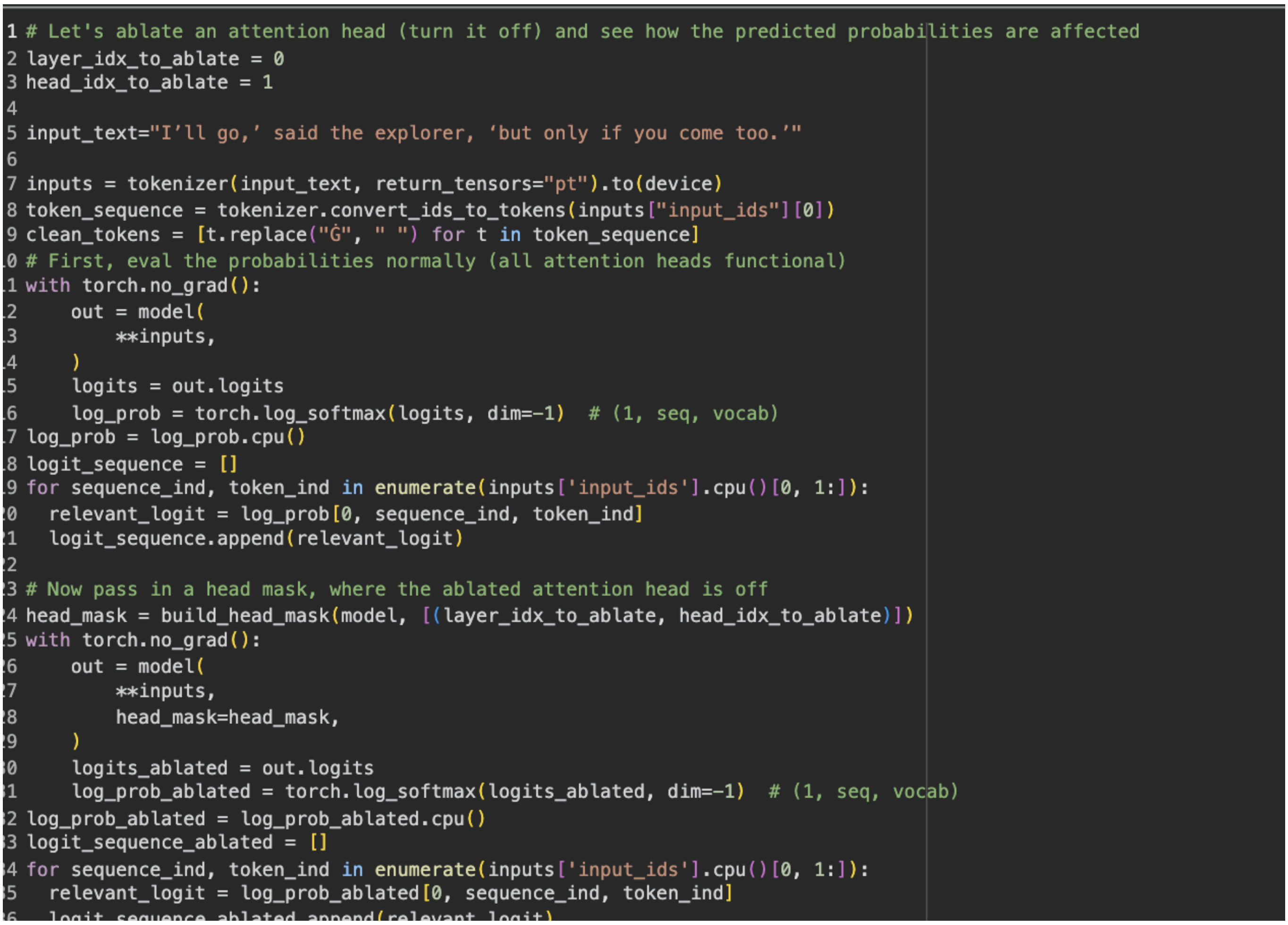
7.re-generate a new text string from that point onward.

```
Regenerated continuation:
Power resides where men believe it resides. It is a place, a shadow on the wall,
and a very small man can cast a very large shadow. When the shadow is cast, the
men believe it will be in heaven.
The shadow, in fact, is a symbol of God, and
```

The model probably found a connection between *reside* and decided that the best next token is *place* instead of *trick*.

(7)attention head ablation

1.for the following text string : "'I'll go,' said the explorer, 'but only if you come too.'", ablate(turn off) one of the heads in the first layer .



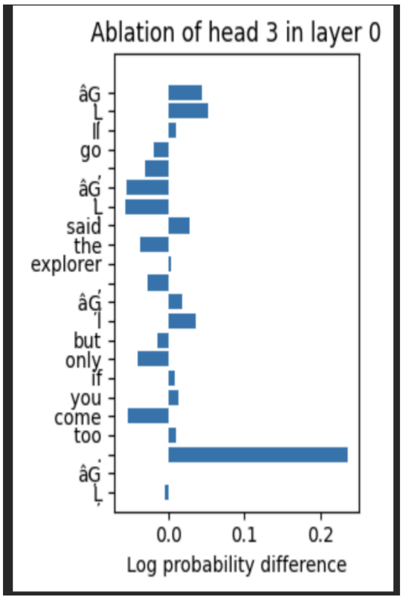
```python
# Let's ablate an attention head (turn it off) and see how the predicted probabilities are affected
layer_idx_to_ablate = 0
head_idx_to_ablate = 1

input_text="I'll go,' said the explorer, 'but only if you come too.'"

inputs = tokenizer(input_text, return_tensors="pt").to(device)
token_sequence = tokenizer.convert_ids_to_tokens(inputs["input_ids"][0])
clean_tokens = [t.replace("Ġ", " ") for t in token_sequence]
# First, eval the probabilities normally (all attention heads functional)
with torch.no_grad():
    out = model(
        **inputs,
    )
    logits = out.logits
    log_prob = torch.log_softmax(logits, dim=-1)  # (1, seq, vocab)
log_prob = log_prob.cpu()
logit_sequence = []
for sequence_ind, token_ind in enumerate(inputs['input_ids'].cpu()[0, 1:]):
  relevant_logit = log_prob[0, sequence_ind, token_ind]
  logit_sequence.append(relevant_logit)

# Now pass in a head mask, where the ablated attention head is off
head_mask = build_head_mask(model, [(layer_idx_to_ablate, head_idx_to_ablate)])
with torch.no_grad():
    out = model(
        **inputs,
        head_mask=head_mask,
    )
    logits_ablated = out.logits
    log_prob_ablated = torch.log_softmax(logits_ablated, dim=-1)  # (1, seq, vocab)
log_prob_ablated = log_prob_ablated.cpu()
logit_sequence_ablated = []
for sequence_ind, token_ind in enumerate(inputs['input_ids'].cpu()[0, 1:]):
  relevant_logit = log_prob_ablated[0, sequence_ind, token_ind]
  logit_sequence_ablated.append(relevant_logit)
```

2.plot the difference in log probabilities across the sequence.



3.Comment on the magnitude of the effect and the specific tokens where the change is most dramatic.
The most dramatic change is at the end with the token “.” with large positive log probability.This shows that removing head 3 actually makes the model more confident about predicting that token.

4.Explain what the ablation is doing, and what the changed log probabilities tell us (if anything).
In the ablation for head 3, we replace the head’s output at each token with the output from a different but related sentence. If the logit difference changes a lot, we assume that the head or token mattered. For example, in this case for head 3, the last token (.) changed positively, so ablating this head at that token helps the model with predicting the next token.

(8)Source, including AI usage

I used AI to help me better understand key concepts in the assignment, such as how attention heads work, how ablation affects logits, and how to interpret heatmaps, and to confirm whether my interpretations were reasonable. I also used AI to assist with generating and refining parts of my code, including tokenization, looping over multiple inputs, extracting top-k probabilities, and formatting plots, which allowed me to focus on the analysis rather than getting stuck on syntax or implementation details.

(9)code
https://colab.research.google.com/drive/13upXBpIyarJadoVLUbPoGz7XFqgpl89_?usp=sharing