AI_HW4_repo_111550076

1. Describe your understanding and findings about the attention mechanism by exBERT.

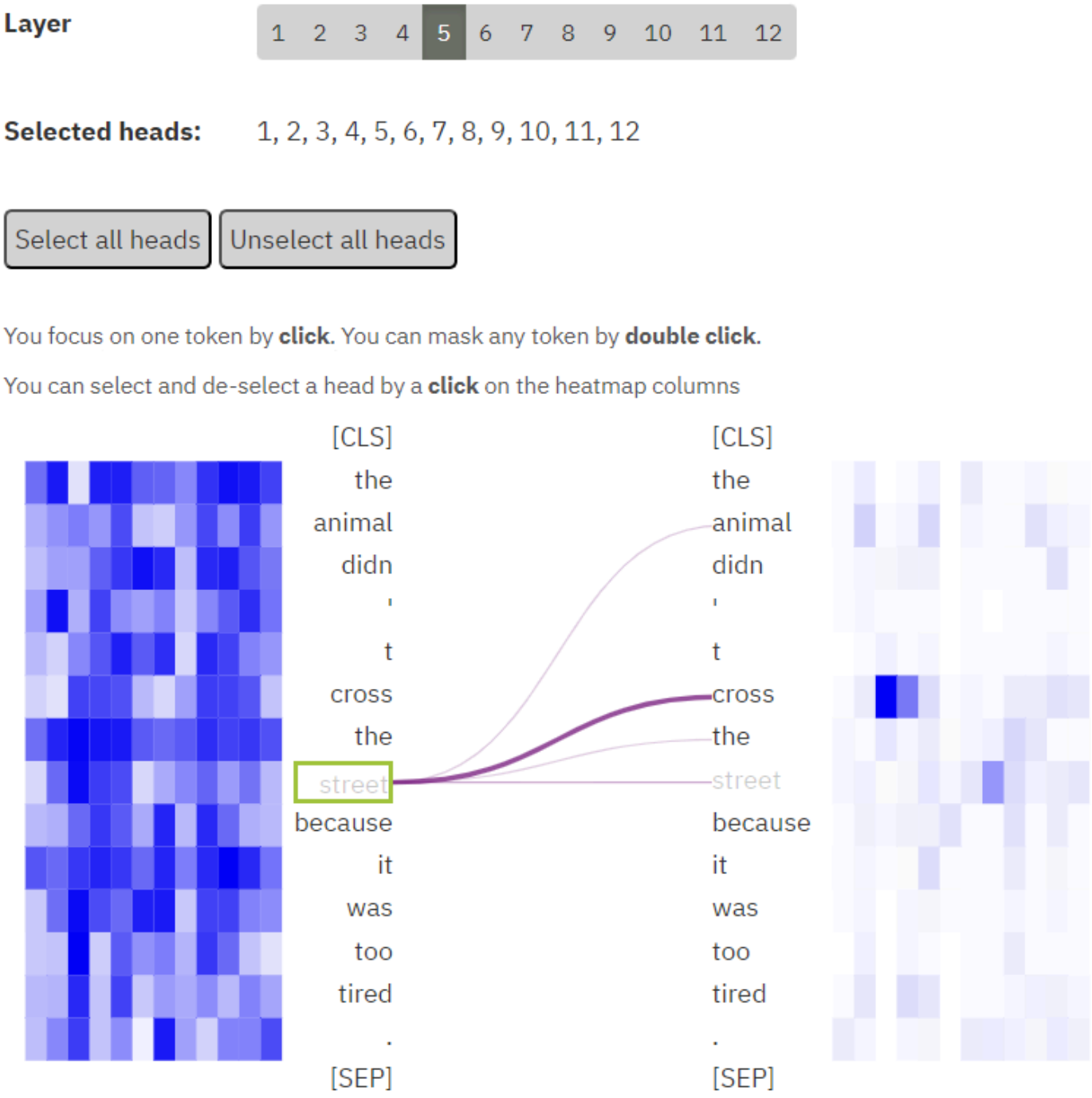
BERT

`BERT` (Bidirectional Encoder Representations from Transformers) is a Transformer-based NLP model that excels in understanding and generating text. It is pre-trained using two tasks: a masked language model and next sentence prediction. For example, in the sentence "The woman reads a [MASK] and enjoys a cup of [MASK]," BERT is trained to predict words like "book" and "tea," which helps it grasp the contextual meaning and relationships of words within a sentence.

Next sentence prediction is a task where the model receives two sentences, say sentence A and sentence B, and it must determine whether sentence B logically follows sentence A as its continuation. This task helps the model understand the logical and relational context between sentences.

Example of BERT using exBERT

In this section, I utilize the bert-base-cased model within exBERT to analyze BERT's attention mechanisms. By examining the sentence "The animal didn't cross the street because it was too tired," and specifically focusing on the masked word "street," I selected layer 5 and all heads for a detailed investigation. The analysis revealed that the words "animal," "cross," are relevant to "street," with "cross" showing the highest relevance. This demonstrates how the attention mechanism can pinpoint connections between words in a sentence.

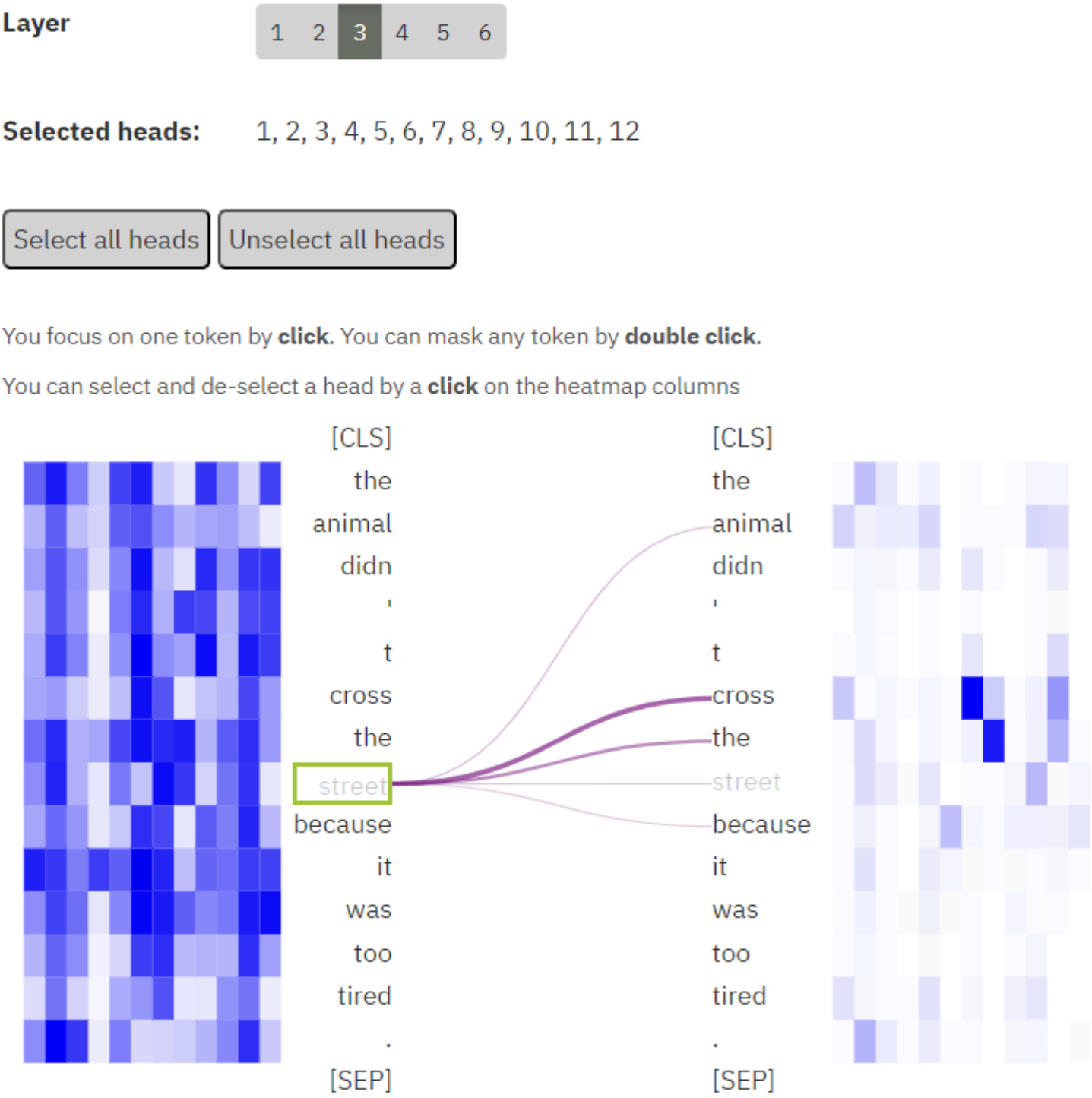


DistilBERT

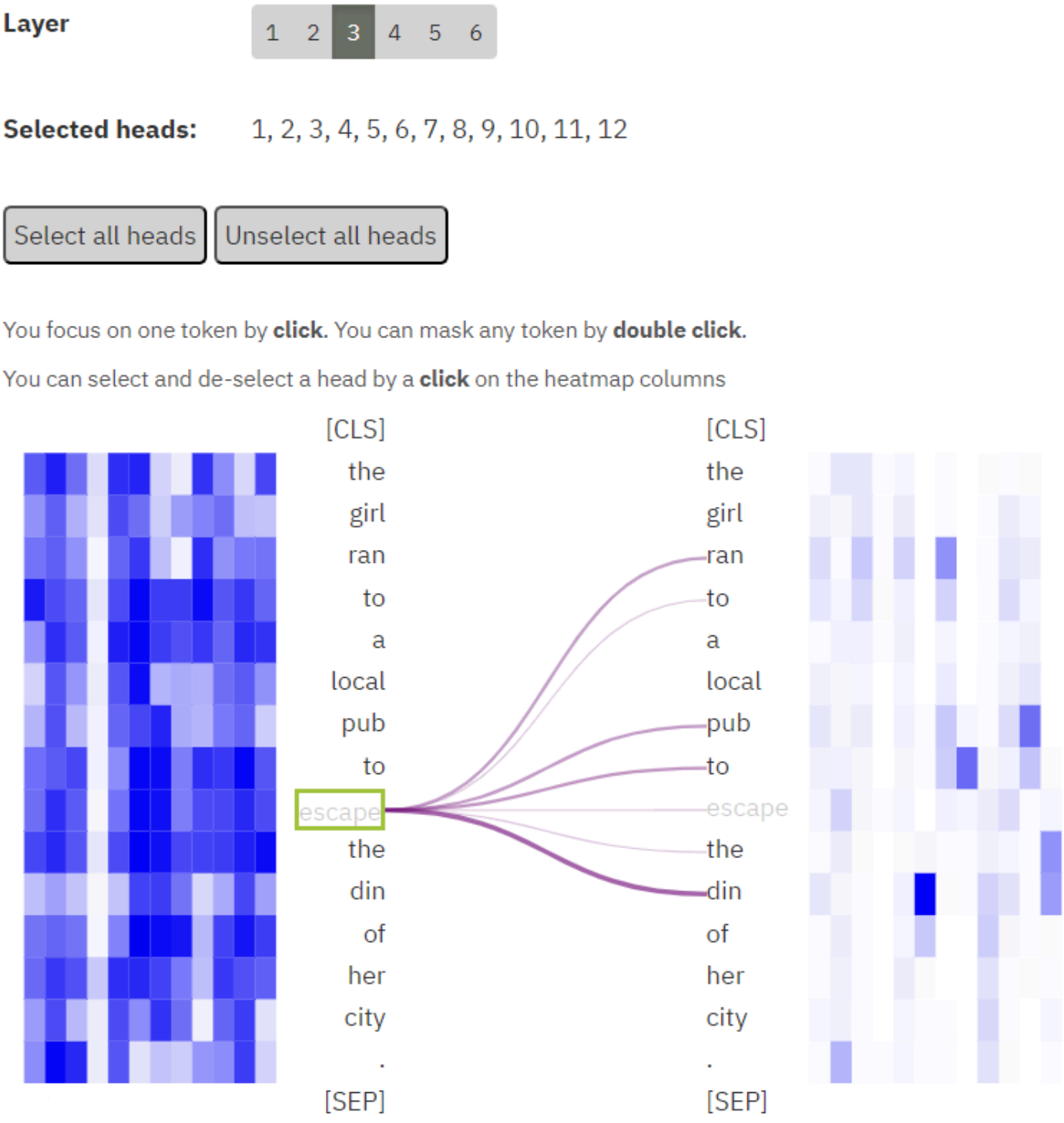
`DistilBERT` is a streamlined version of `BERT` designed to mimic the original model's capabilities. It employs a lighter architectural framework, reducing BERT's size by 40% while retaining 97% of its language comprehension abilities and operating 60% faster. This efficiency makes `DistilBERT` a resource-saving option without significant performance sacrifices.

Example of DistilBERT using exBERT

In this section, I utilize the distilbert-base-uncased model within exBERT to analyze DistilBERT's attention mechanisms. By examining the same sentence "The animal didn't cross the street because it was too tired," and specifically focusing on the masked word "street," I selected layer 3 and all heads for a detailed investigation. The result is similar to BERT. This demonstrates that DistilBERT has absorbed some knowledge from the original BERT model.

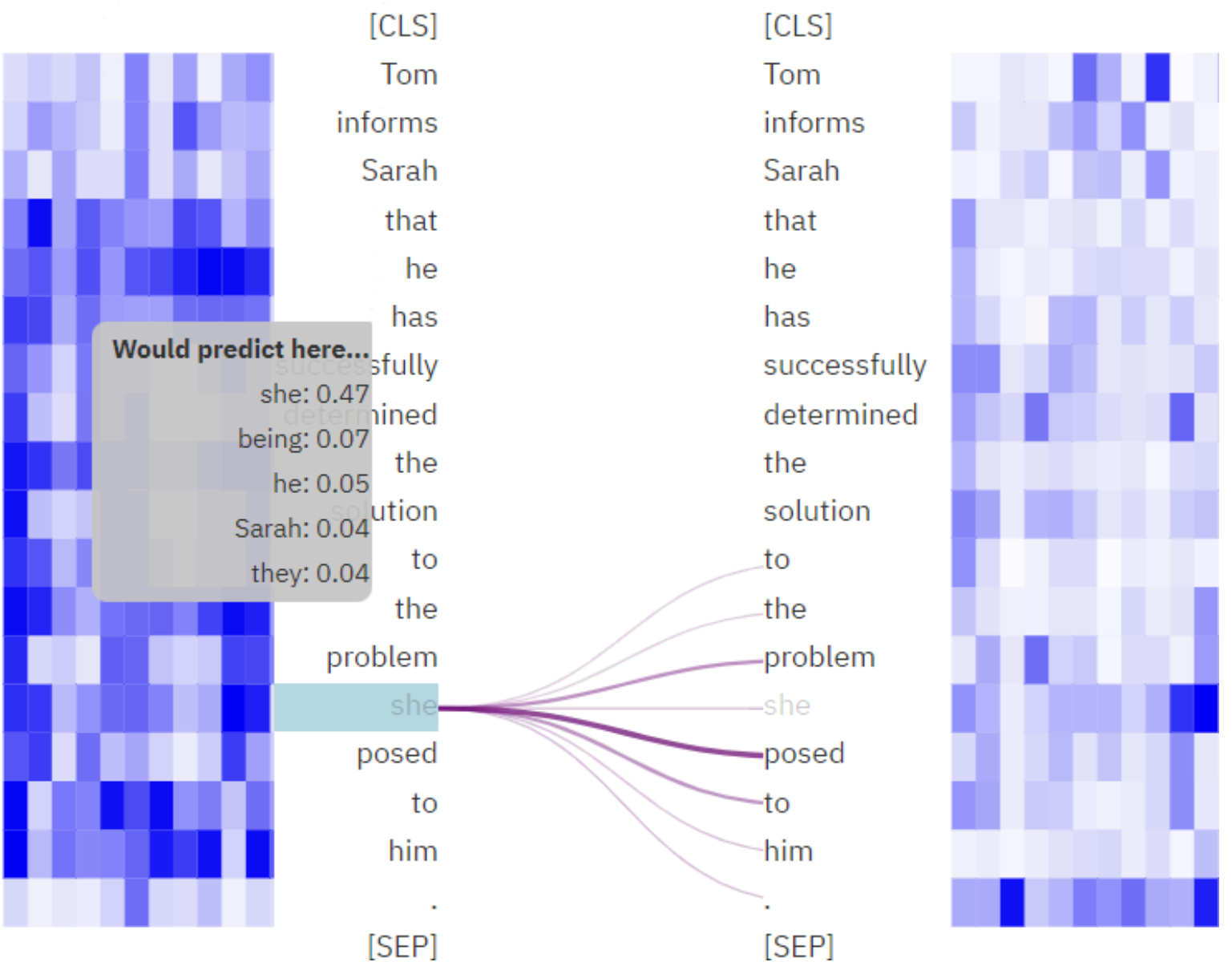


Another example, by examining the sentence "The girl ran to a local pub to escape the din of her city." and specifically focusing on the masked word "escape". I selected layer 3 and all heads for a detailed investigation. The analysis revealed that the words "ran", "pub", “din” are relevant to "escape" with "din" showing the highest relevance. This demonstrates how the attention mechanism can pinpoint connections between words in a sentence.

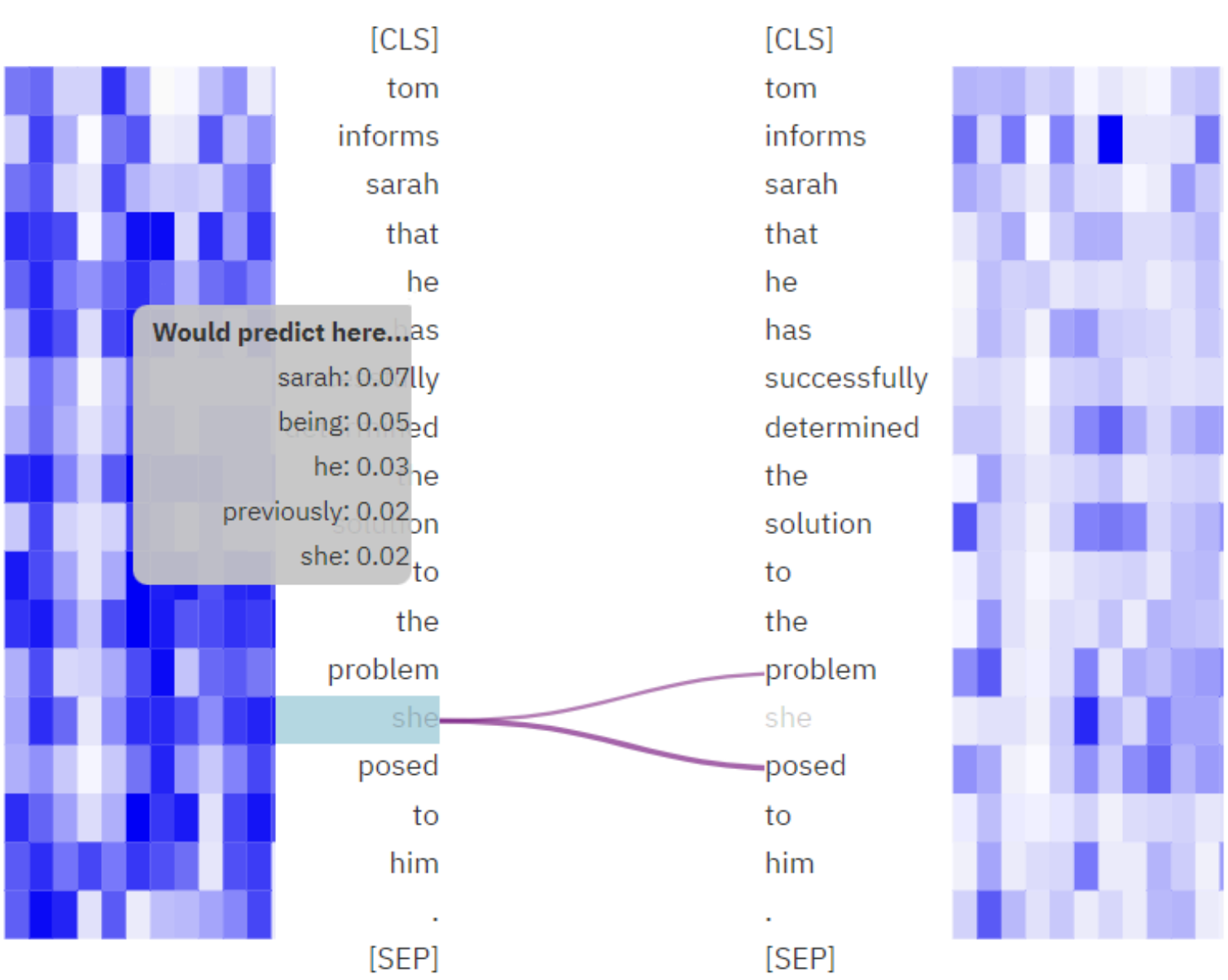


Compare with BERT and DistilBERT.

Using the sentence "Tom informs Sarah that he has successfully determined the solution to the problem she posed to him," I masked the word "she" and employed the bert-base-cased model with layer 9, covering all heads. The analysis accurately predicted "she" with the highest probability, showcasing the model's effective contextual understanding and predictive accuracy.



However, when I used the distilbert-base-uncased model on the same sentence with layer 3 and all heads, it failed to predict the correct answer.



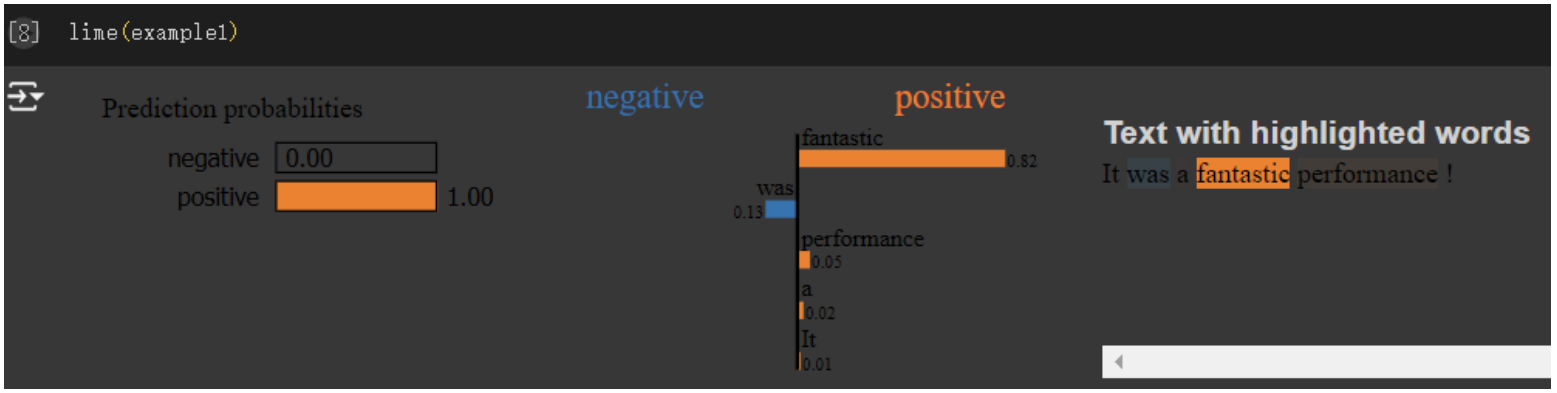
In conclusion, although DistilBERT can accurately predict words in simpler sentences, it may not perform as well when dealing with more complex sentences.

2. Compare at least 2 sentiment classification models (e.g., TA_model_1.pt, your model in HW2).

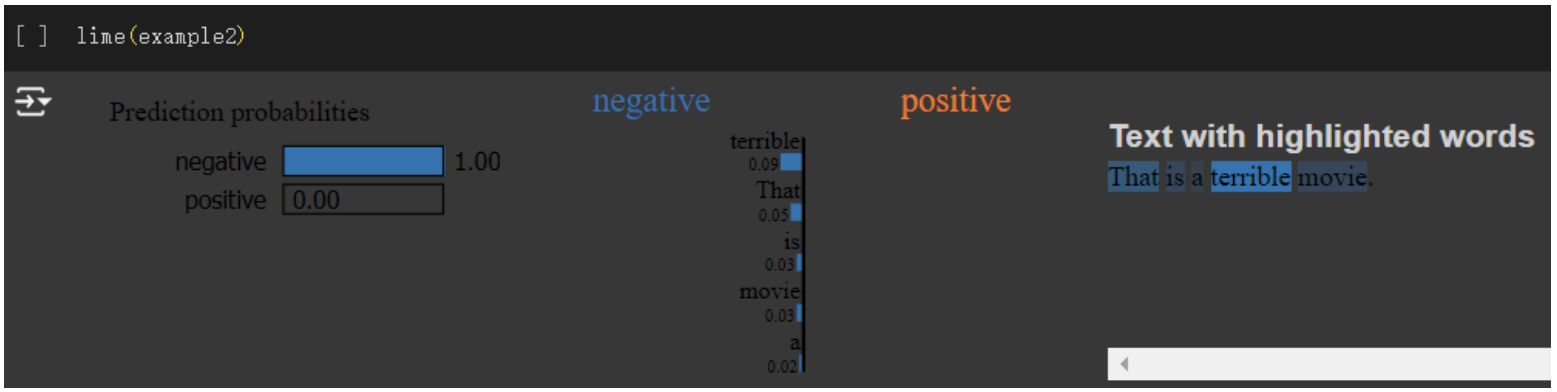
In this section, I analyze TA_model_1.pt and TA_model_2.pt using LIME and SHAP techniques across four examples.

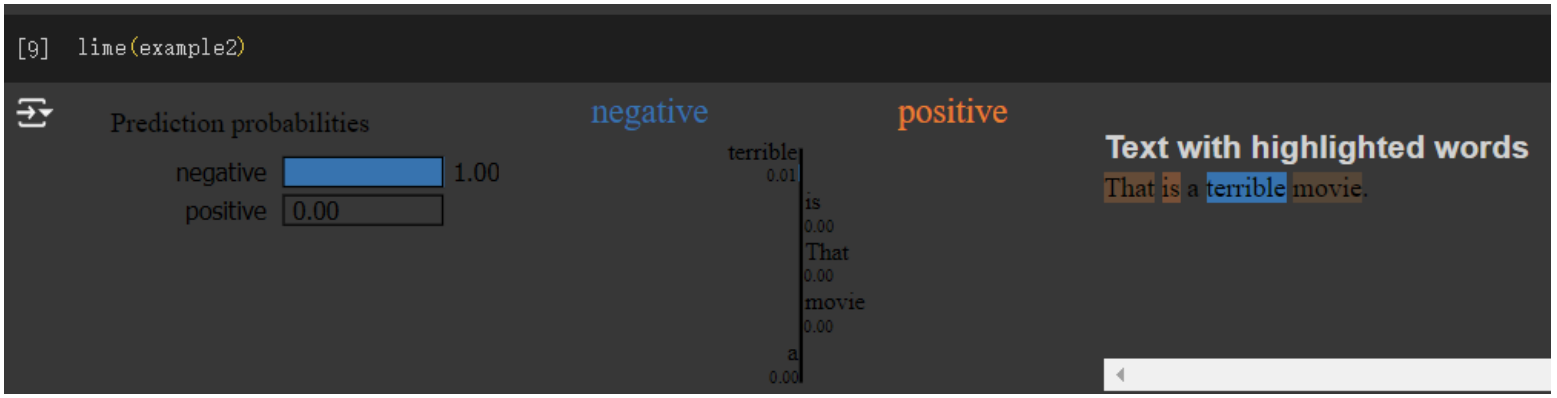
- example1 = 'It was a fantastic performance !'
- example2 = 'That is a terrible movie.'
- example3 = 'The movie dazzles with its inventive visuals and compelling storytelling. The performances are powerful, deeply enriching the emotional impact.'
- example4 = 'The film struggles with a convoluted plot and sluggish pacing. Despite a talented cast, the characters lack depth and fail to connect with the audience.'

1. LIME for TA_model_1.pt and TA_model_2.pt on example1

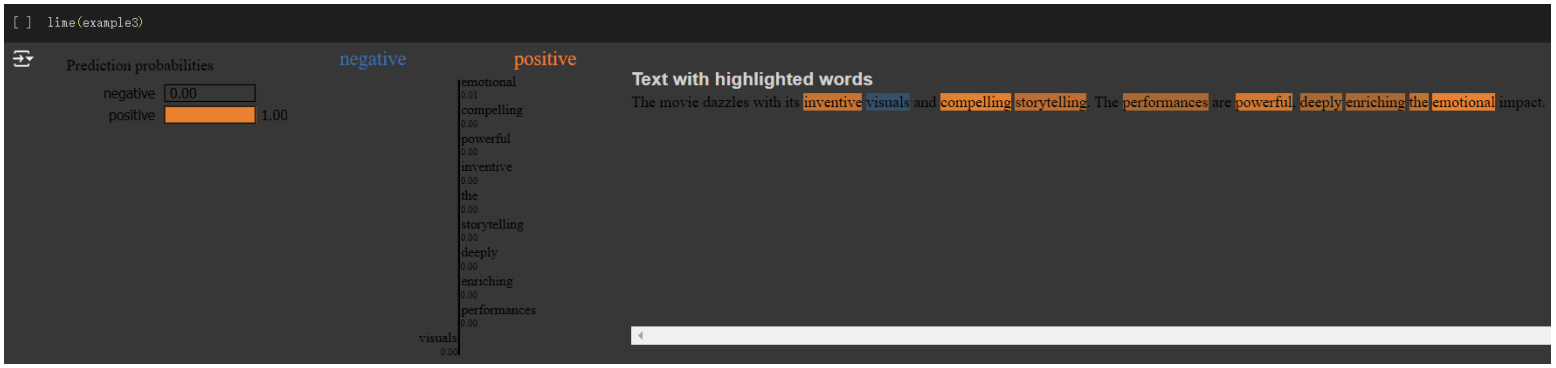


2. LIME for TA_model_1.pt and TA_model_2.pt on example2

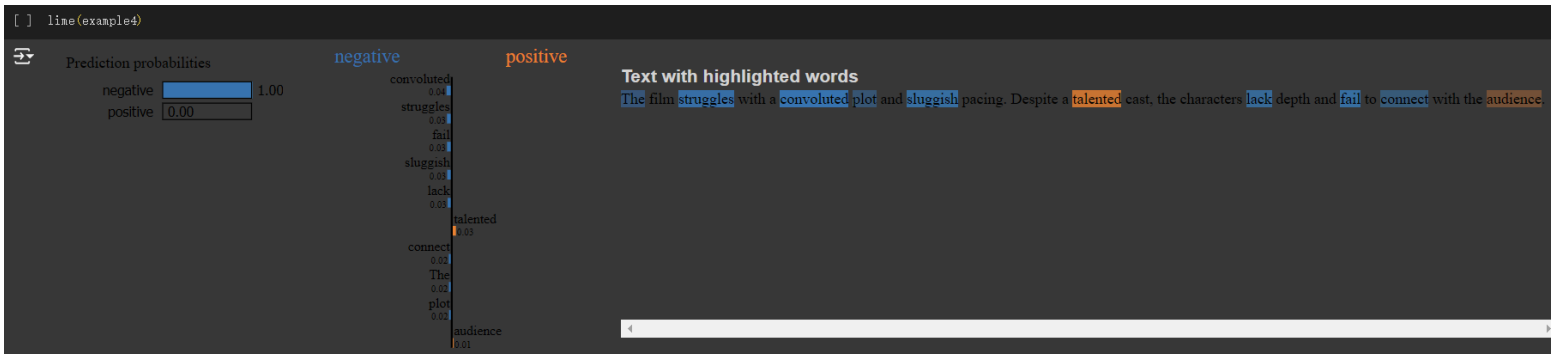


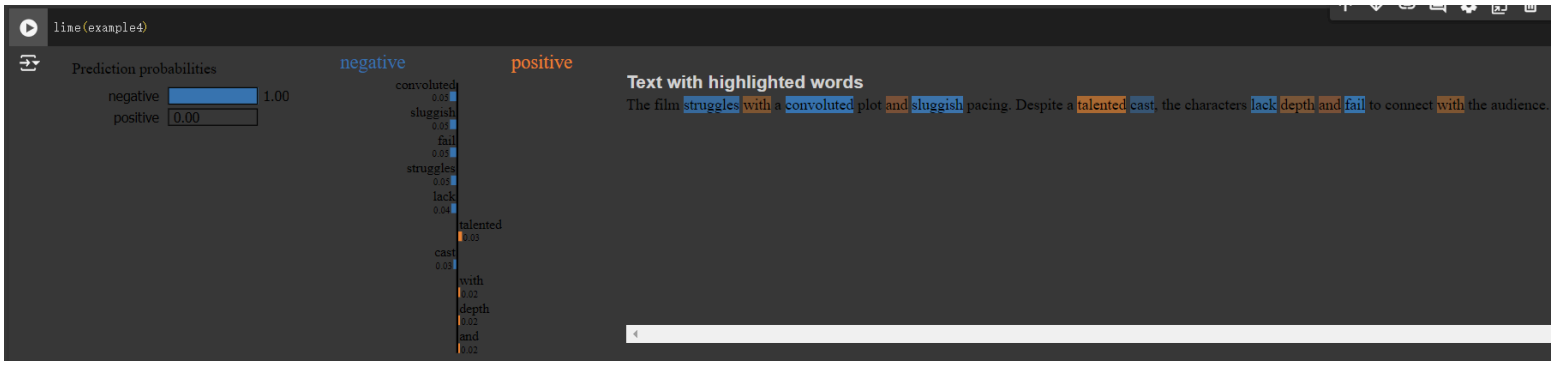


3. LIME for TA_model_1.pt and TA_model_2.pt on example3



4. LIME for TA_model_1.pt and TA_model_2.pt on example4





Based on LIME visualizations for two models (TA_model_1.pt and TA_model_2.pt) across four examples, here are some observations about the differences:

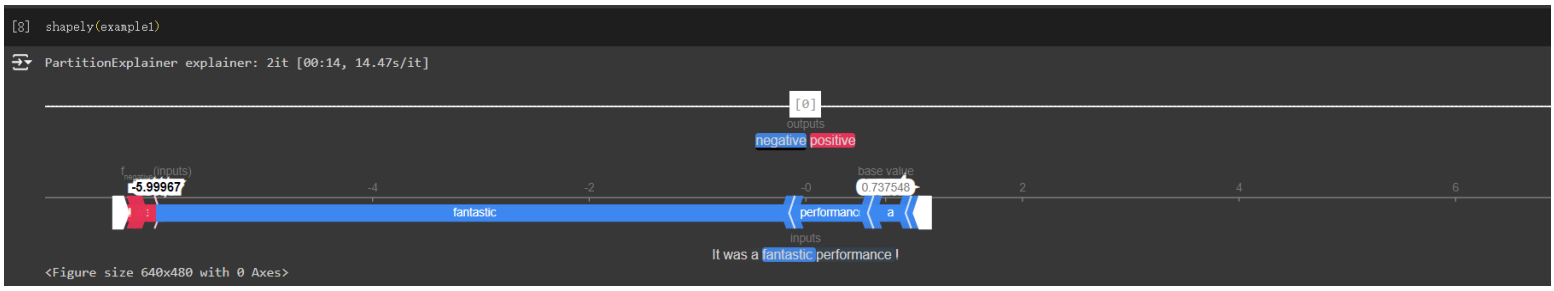
1. **Example 1 - Positive Sentiment**: Both models predict the sentiment as positive, but TA_model_2.pt assigns a significantly higher importance to the word "fantastic", suggesting it is more sensitive to positive keywords in its predictions.

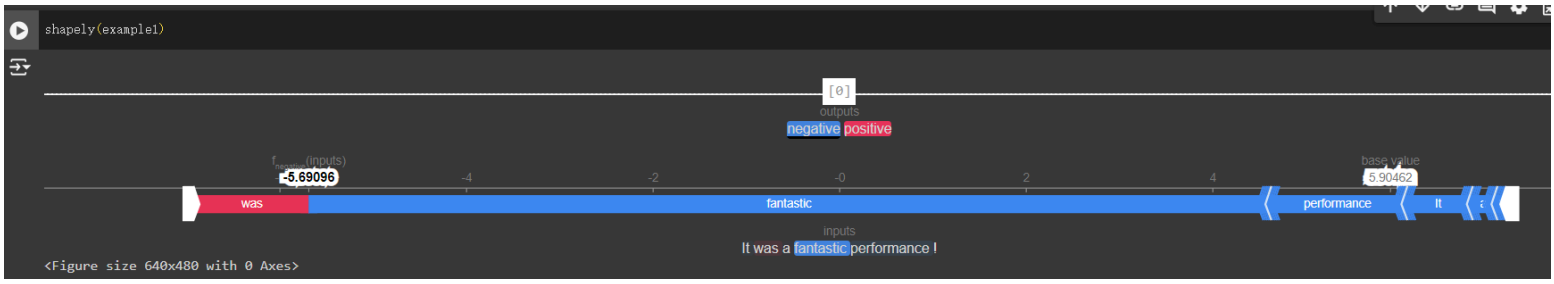
2. **Example 2 - Negative Sentiment**: Both models correctly predict the sentiment as negative for the sentence about the movie being terrible, with similar weight distributions across the key negative words.
3. **Example 3 - Positive Sentiment**: Again, both models predict positive sentiment. The weight distributions are quite similar, focusing on words like "emotional", "compelling", and "enriching", indicating consistency in recognizing positive sentiment descriptors.
4. **Example 4 - Negative Sentiment**: Both models agree on the negative sentiment of the film's description. They attribute similar importance to words like "struggles", "convoluted", and "sluggish", though TA_model_2.pt gives slightly higher weights to certain negative descriptors.

Overall, TA_model_2.pt appears to emphasize sentiment-indicative words more strongly than TA_model_1.pt, potentially making it more sensitive or biased towards certain expressions in the text. This could affect its reliability in nuanced contexts where the sentiment is less straightforward.

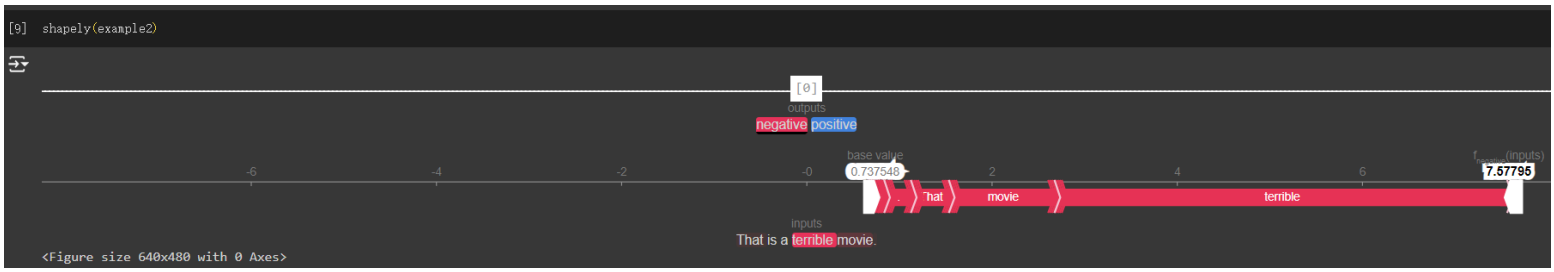
---

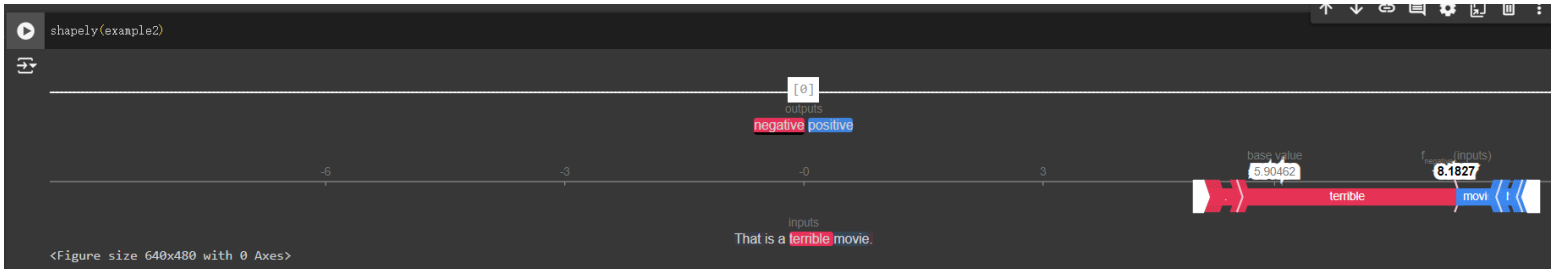
5. SHAP for TA_model_1.pt and TA_model_2.pt on example1



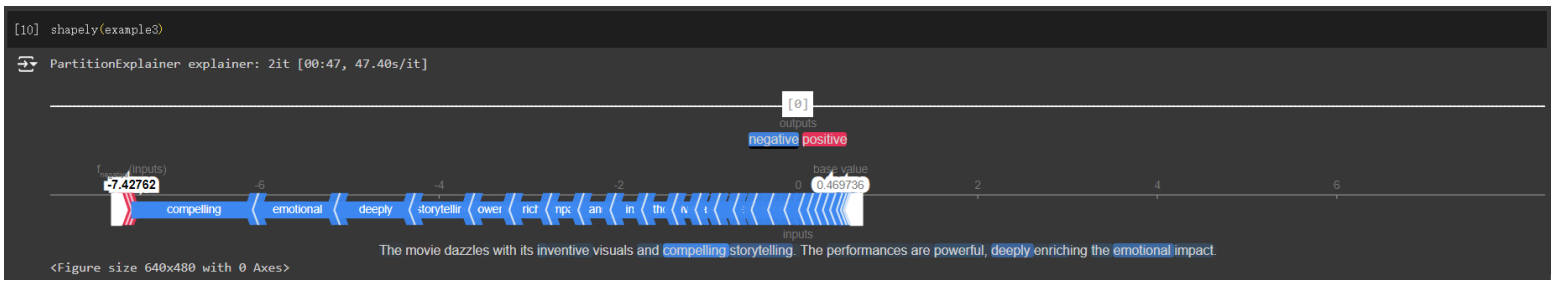


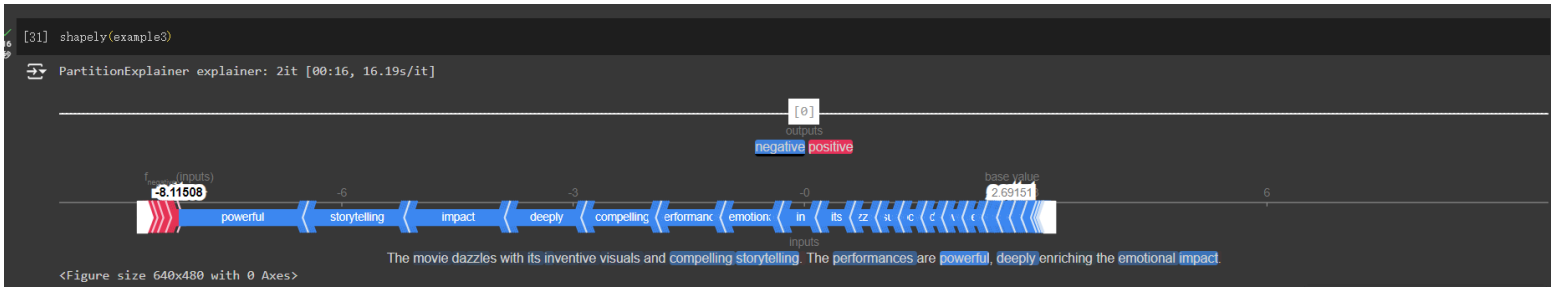
6. SHAP for TA_model_1.pt and TA_model_2.pt on example2



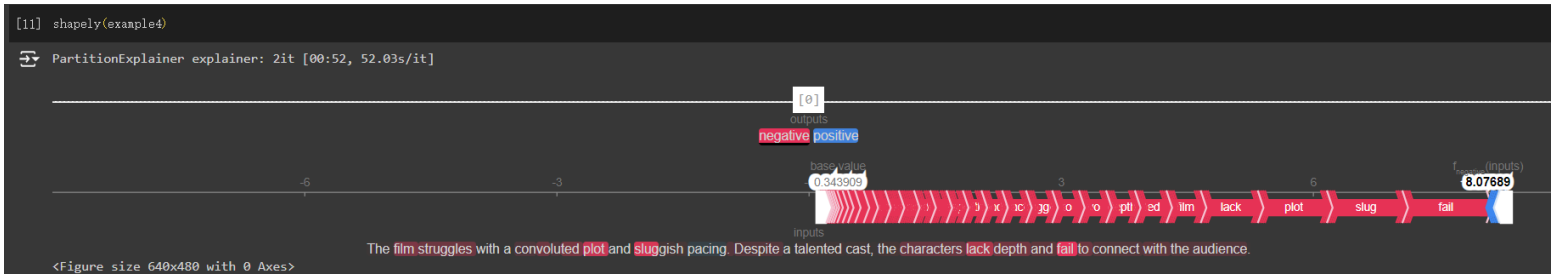


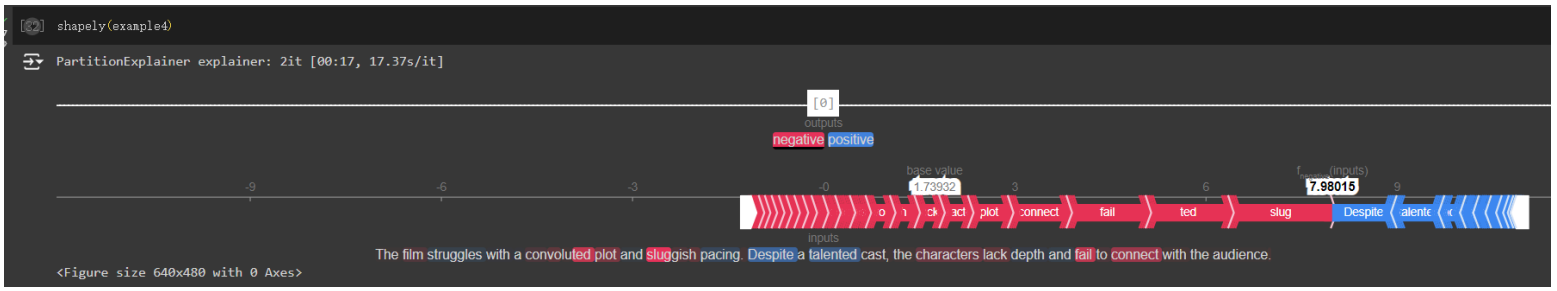
7. SHAP for TA_model_1.pt and TA_model_2.pt on example3





8. SHAP for TA_model_1.pt and TA_model_2.pt on example4





Based on the SHAP visualization outputs for both TA_model_1.pt and TA_model_2.pt across four examples:

1. **Example 1**: TA_model_1.pt assigns a negative SHAP value to the word "fantastic," indicating it is driving the prediction towards a negative outcome, which seems counterintuitive. In

contrast, TA_model_2.pt assigns a strong positive SHAP value to "fantastic," aligning more expectedly with its positive connotation.

2. **Example 2**: Both models identify "terrible" as the key driver towards a negative prediction, with TA_model_2.pt giving it a slightly higher negative SHAP value, indicating it weighs this negative sentiment more heavily.
3. **Example 3**: In this positive sentiment example, TA_model_2.pt assigns higher positive SHAP values to words like "powerful" and "storytelling" compared to TA_model_1.pt, suggesting that TA_model_2.pt may be more sensitive to specific positive descriptors.
4. **Example 4**: Both models emphasize "convoluted" and "sluggish" heavily in their negative predictions, but TA_model_2.pt shows a higher magnitude of negative SHAP values, indicating stronger attribution to these negative terms.

Overall, TA_model_2.pt generally attributes higher SHAP values to key descriptive words, whether positive or negative, suggesting a possibly more sensitive or definitive modeling approach compared to TA_model_1.pt.

3. Compare the explanation of LIME and SHAP.

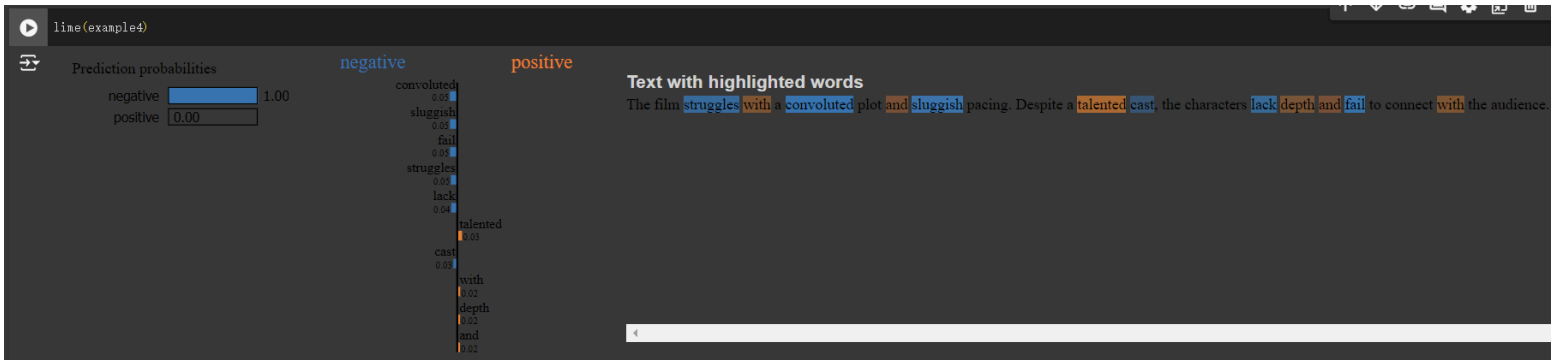
Explanation

`LIME` (Local Interpretable Model-agnostic Explanations) focuses on creating interpretable models around individual predictions, simplifying the model locally to explain how input features affect predictions at a specific point. It's particularly useful for understanding model behavior on a specific instance.

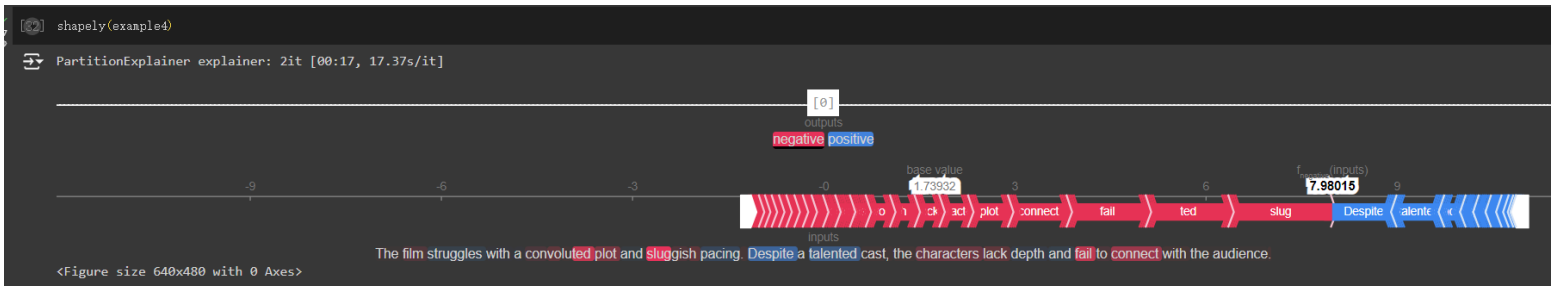
`SHAP` (SHapley Additive exPlanations), on the other hand, uses game theory to calculate the Shapley values of each feature, which quantitatively measure the impact of each feature on the prediction across the entire dataset, not just locally. This provides a more global perspective on feature importance, offering explanations that are consistent with the model across all data points.

compare the outputs of two distinct methods.

- LIME: Offers explanations by assigning weights to input variables, indicating their influence on specific predictions.



- SHAP: Assigns feature attributions that quantify each feature's contribution to the prediction outcome through Shapley values. These values not only highlight the importance of each feature but also establish a base value that ensures predictions aren't solely based on weighted contributions, providing a more thorough assessment.



This comprehensive approach is visible when we compare the differences with the results in problem 4, where SHAP not only reveals the significance of each word in a sentence but also balances positive and negative feedback for a complete analysis.

In summary, LIME generates local explanations for individual predictions using perturbation-based methods, whereas SHAP offers both local and global explanations through feature importance values derived from game theory concepts.

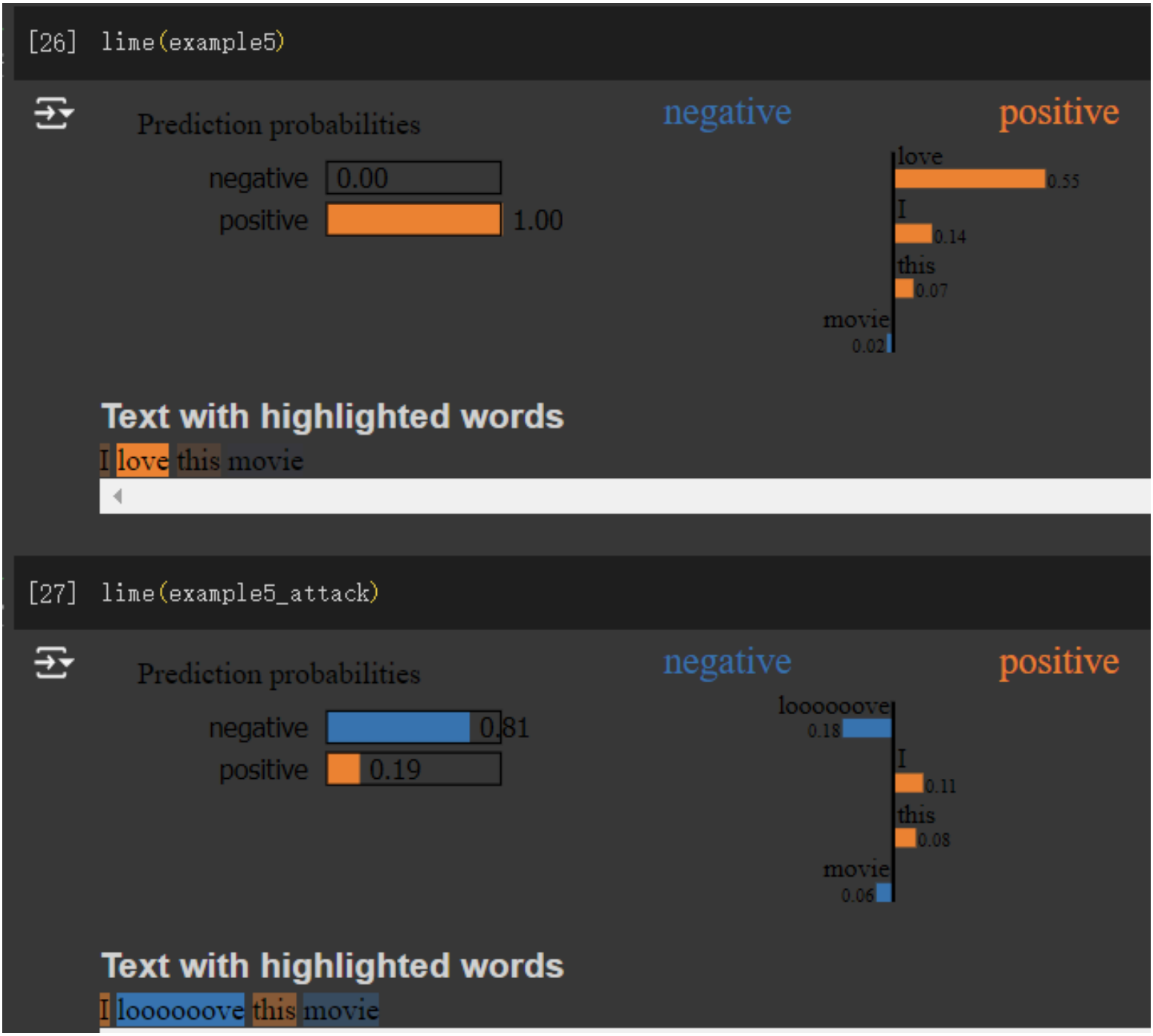
4. Try 3 different input sentences for attacks. Also, describe your findings and how to prevent the attack if you retrain the model in the future.

1. Misspelling

example5 = "I love this movie”

example5_attack = "I looooooove this movie”

misspelling “love” to “looooooove” and the result is different, which successfully attack the model.



how to prevent the attack

consider retraining your model with examples that include common misspellings and exaggerated expressions. Implementing spelling normalization or preprocessing steps that condense repeated characters can help the model understand and process these variations effectively.

2. complicate example

example7_attack = '''This is mainly about a notorious bad guy. Though the overall film is great, the lead actor's performance is quiet disappointing.'''

The review states that it is a great movie, yet it includes many negative words, misleading the model into misclassifying it.

how to prevent the attack

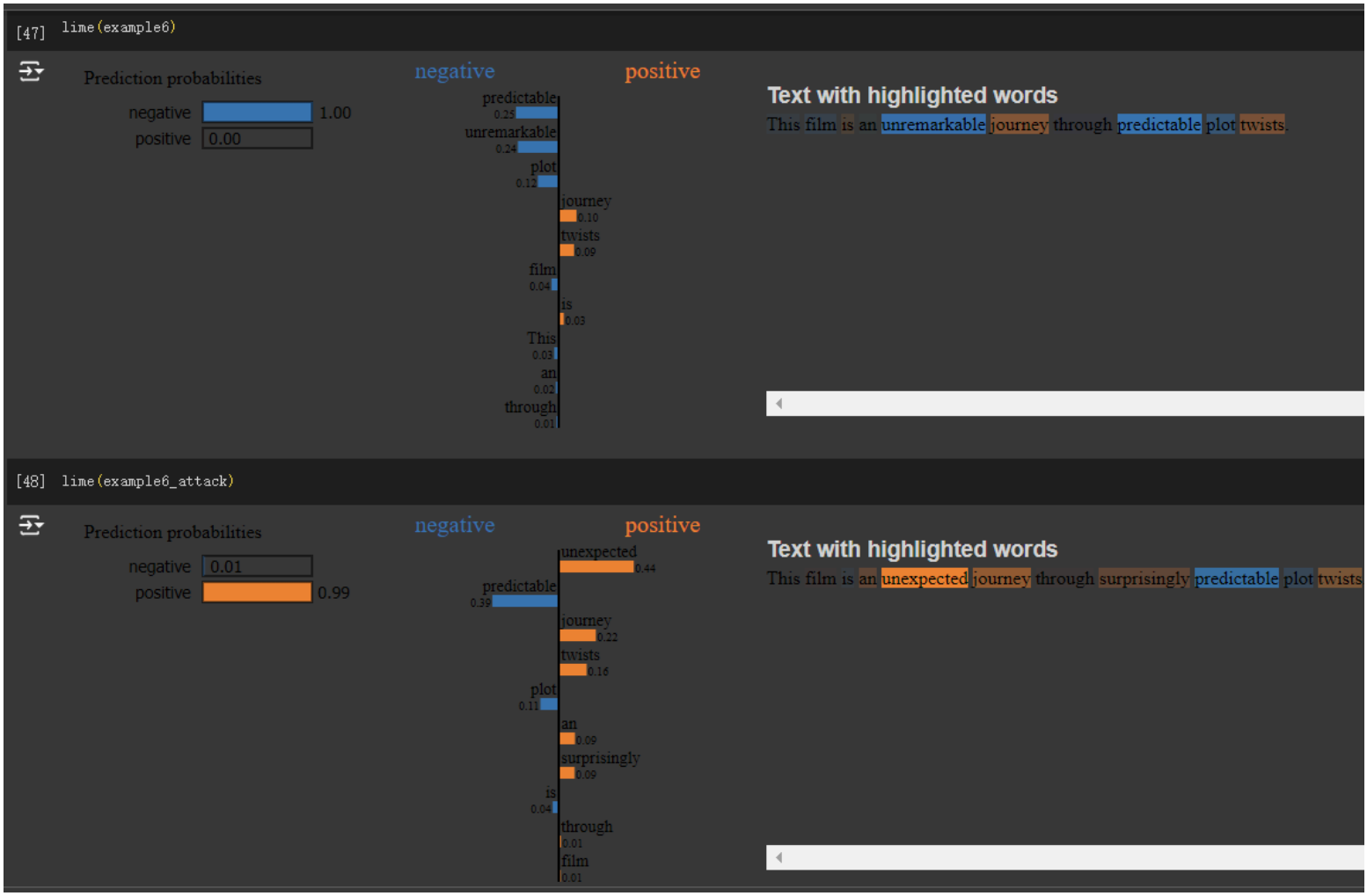
To mitigate attacks using misleading text, retrain the model on varied examples that blend positive and negative sentiments within single statements. Enhancing the model's ability to understand overall context and maintain sentiment consistency across the full sentence can improve its robustness.

3. Replace the words

example6 = "This film is an unremarkable journey through predictable plot twists."

example6_attack = "This film is an unexpected journey through surprisingly predictable plot twists."

"unremarkable" is replaced with "unexpected," which might still carry a neutral or slightly positive undertone, and "surprisingly" is added before "predictable," which might confuse the model by presenting a positive adverb with a typically negative adjective. This could lead the sentiment analysis model to misclassify the sentiment of the review.



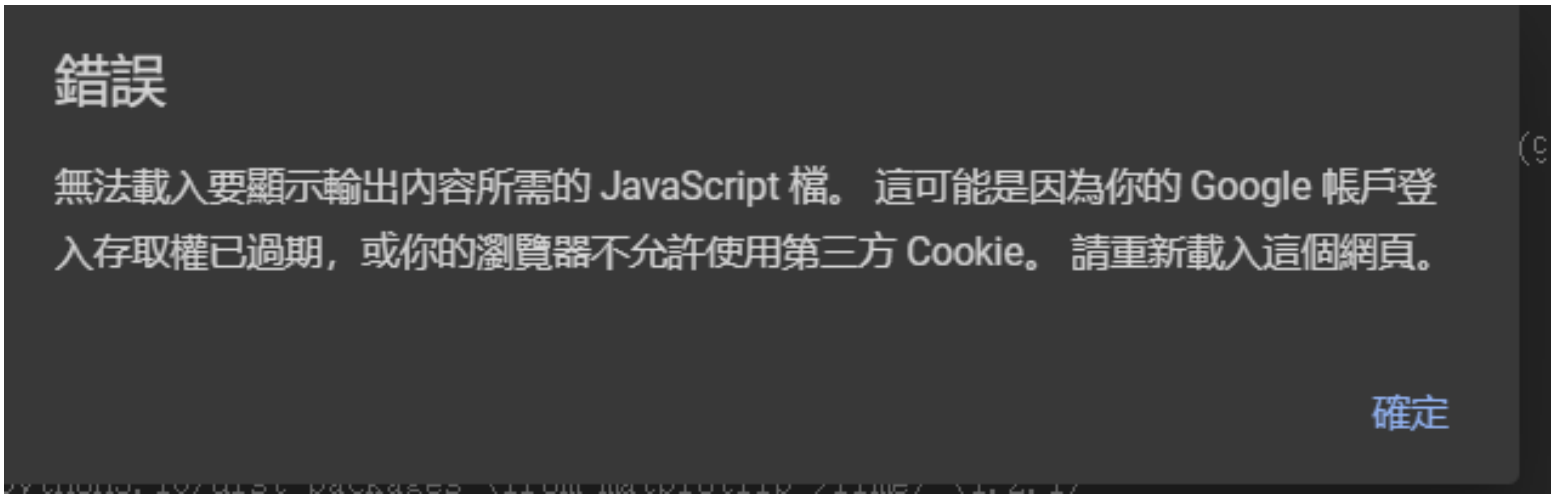
how to prevent the attack

To enhance resilience against word replacement attacks, retrain using data augmentation with synonyms and paraphrases, and include adversarial training with manipulated text examples. Utilizing contextual embeddings can also improve the model's ability to understand words within their broader textual context.

5. Describe problems you meet and how you solve them.

1. Fail to display the result of LIME.(many times)

solve: reload the page.



2. 'DistilBertModel' object has no attribute '_use_flash_attention_2'

```
lime(example1)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-9-bff984a1f786> in <cell line: 1>()
----> 1 lime(example1)

                                 10 frames
/usr/local/lib/python3.10/dist-packages/torch/nn/modules/module.py in __getattr__(self, name)
   1707             if name in modules:
   1708                 return modules[name]
-> 1709         raise AttributeError(f"'{type(self).__name__}' object has no attribute '{name}'")
   1710
   1711     def __setattr__(self, name: str, value: Union[Tensor, 'Module']) -> None:

AttributeError: 'DistilBertModel' object has no attribute '_use_flash_attention_2'
```

solve: install transformers with 4.30.0 version. (!pip install transformers==4.30.0)

3. Hard to attack the model

I spent a lot of time brainstorming ways to trick the model, trying numerous examples without success. Eventually, I turned to internet resources and consulted ChatGPT to craft a sentence capable of leading the model to an incorrect conclusion.