Does Using Counterfactual Help LLMs Explain Textual Importance in Classification?

Anonymous ACL submission

Abstract

Large language models (LLMs) are becoming useful in many domains due to their impressive abilities that arise from large training datasets and large model sizes. More recently, they have been shown to be very effective in textual classification tasks, motivating the need to explain the LLMs' decisions. Motivated by practical constrains where LLMs are black-boxed and LLM calls are expensive, we study how incorporating counterfactuals into LLM reasoning can affect the LLMs ability to identify the top words that have contributed to its classification decision. To this end, we introduce a framework called the *decision changing rate* that helps us quantify the importance of the top words in classification. Our experimental results show that using counterfactuals can be helpful.

1 Introduction

Large language models (LLMs) are becoming useful in many domains (Li et al., 2023a; Sakai and Lam, 2025; Tan et al., 2025) due to their impressive abilities that arise from large training datasets (Li et al., 2023b) and large model sizes (Naveed et al., 2023). More recently, they have been shown to be very effective in textual classification tasks (Zhao et al., 2024b; Sakai and Lam, 2025), motivating the need to explain the LLMs' decisions. In this paper, we investigate how well LLMs can explain their decisions when making textual classifications – specifically, we study the LLM's ability to select the top k key words that led to its classification decision. Some of the potential business use cases of being able to identify the keys words that lead to a certain LLM classification are as follows: (i) Can aid the prompt designer to trim unneccesary words or shorten phrases. This reduces the number of tokens processed which helps to reduce cost. (ii) Helps to provide auditable records of how text inputs drive LLM classifications.

Conterfactuals. In this paper, we focus on a specific type of self-generated explanations called self-generated counterfactual explanations (Wachter et al., 2017; Dehghanighobadi et al., 2025) – which we call simply as counterfactuals: Given an input $\boldsymbol{x}$ and the LLM output $\hat{y}$, the counterfactual $\boldsymbol{x}'$ is a minimally modified version (i.e., the edit/Levenshtein distance between $\boldsymbol{x}$ and $\boldsymbol{x}'$ are minimized) of the input such that when $\boldsymbol{x}'$ is feed into the LLM, its output $\hat{y} \neq \hat{y}'$. For example, $\boldsymbol{x}$ is *'This item is neither affordable, nor of good quality.'* and $\boldsymbol{x}'$ is *'This item is both affordable, and of good quality.'*

Note that we do not solve any optimization problem to obtain $\boldsymbol{x}'$, but simply ask for the LLM for $\boldsymbol{x}'$. This is motivated by recent work (Dehghanighobadi et al., 2025) that showed that LLMs are generally able to produce counterfactuals with a high success rate for classification problems with a few classes. We build on their work to study how these counterfactuals affect key-word identificiation.

Problem setup. Given an input $\boldsymbol{x}$ and completely black-box LLM (i.e., no accesss to weights, gradients, etc.), we want to produce the top-k words that explain the classification decision of the LLM, where k is up to the user to choose.

Motivation. Our problem setup is motivated by businesses that use black-box LLMs (e.g., GPT-4o) but have no access to the inner workings of the LLMs. At the same time, LLM calls are expensive so we want to minimize it for cost savings. This leads to incoporating counterfactuals into LLM reasoning as a viable approach for LLM explainability. Our main research questions are:

1. Do counterfactuals help LLM with explaining the importance of words in textual classification?

2. How do we score the LLM's ability to select

the top-k words that influenced its classification decision?

2 Related Work

Known black-box explainability techniques for textual classification and for LLMs are as follows:

- SHAP (Lundberg and Lee, 2017) and LIME (Ribeiro et al., 2016) are common input-feature based techniques that works reasonably well for smaller models, but are they too expensive for LLMs, requiring 100-1000 LLM calls per classification output which is not practical due to the high cost. Furthermore, prior work (Ichikawa and Steup, 2014) argues that counterfactuals constitute a better test of knowledge over LIME and SHAP.

- Prompting the LLM directly can take as low as 1 prompt, but this totally ignores the internal mechanism of the LLM and it has been shown that LLMs do not always say what they think (Chen et al., 2025). Nevertheless, it has been shown to align with human intuition (Agarwal et al., 2024). There has also been other work that has argued that the contrastive nature of counterfactuals are better aligned with human reasoning (Miller, 2019) – making it interesting to study whether counterfactuals can improve on direct prompting.

Black-box approaches aside, there are many other LLM explainability approaches – we refer the interested reader to the following surveys (Zhao et al., 2024a; Luo and Specia, 2024) for a comprehensive review.

Main contributions. With respect to the main research questions, are contributions are as follows:

1. We investigate the usefuless of counterfactuals in helping LLM explain the importance of input words in textual classification. Our experimental results show that using counterfactuals can help improve the identification of the top-k words.

2. To achieve the above goal, we introduce a metric known as the *decision-changing* rate, which captures how important the top-k words to the LLM's classification decision. To the best of our knowledge, this is a new metric that could be useful for the field.

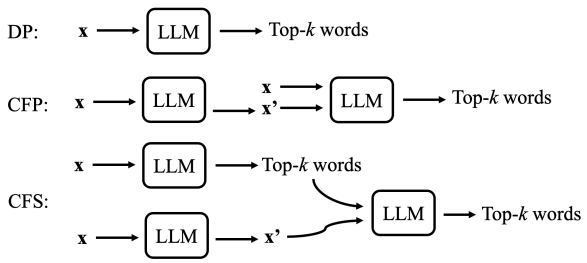

Figure 1: Visualization of the different approaches

3 Methodology

We introduce 3 approaches to obtain the top-k words that led to an LLM's classification decision: *direct prompting*, *counterfactual-parallel*, and *counterfactual-sequential*. The prompts for all 3 approaches are given in Appendices A, B, and C, and we give the high-level ideas below.

Direct prompting (DP). Directly prompt the LLM for the classification and ask it to list the top-k most important words in the input that led to its classification decision. This is shown visually in Figure 1.

Counterfactual-parallel (CFP). The key steps are shown visually in Figure 1 and are stated as follows: (1.) Ask the LLM to produce the counterfactual $\boldsymbol{x}'$. (2.) Check if the LLM outputs satisfy $\hat{y}' \neq \hat{y}$; if this is not true then use DP to produce the top-k most important words. Otherwise, proceed with the next step. (3. Parallel step) Feed $\boldsymbol{x}$ and $\boldsymbol{x}'$ into the LLM and ask it to produce the top-k most important words that led to its previous classification decision $\hat{y}$.

Counterfactual-sequential (CFS). The key steps are shown visually in Figure 1 and are stated as follows: (1.) Use DP to get the top-k most important words that led to its classification decision $\hat{y}$. (2.) Ask the LLM to produce the counterfactual $\boldsymbol{x}'$. (3.) Check if the LLM outputs satisfy $\hat{y}' \neq \hat{y}$; if this is not true then use DP to produce the top-k most important words. Otherwise, proceed with the next step. (4. Refinement step) Feed $\boldsymbol{x}$ and $\boldsymbol{x}'$ into the LLM and ask it to refine its top-k most important words that led to its classification decision. It is allowed to keep the initial top k words as the final choice of words if it is confident.

Extensions to CFP and CFS. We can take advantage of multiple LLM prompts (i.e., sampling) to produce a weights vector corresponding to the words in the input text, where each word in the

input text corresponds to a weight and the weight indicates how important the word is in the classification decision. The key steps are as follows: (i) Set the temperature of the LLM to be greater than 0 (e.g., 1), and decide on the number of important words k. (ii) Have a vector of weights $\boldsymbol{w}$ (initialized to all zeros) with each entry corresponding to a word in $\boldsymbol{x}$. (iii) Run either CFP or CFS $n > 1$ times. For each run, if a word is included in the top-k most important words, then increment its corresponding weight in $\boldsymbol{w}$ by 1. (iv) Divide all entries in $\boldsymbol{w}$ by n and produce $\boldsymbol{w}$ as the final output. This vector $\boldsymbol{w}$ can be used to produce a heatmap of the words.

4 Experiments

4.1 Dataset and Evaluation Metric

We look at the following textual binary classification datasets of different lengths:

- **Amazon (Kotzias et al., 2015) (short-length).** Dataset consists of 500 positive and 500 negative sentences from customer reviews were extracted from the amazon.com.
- **SST2 (Pang and Lee, 2005) (medium-length).** Dataset consists of 11,855 single sentences extracted from movie reviews.
- **IMDB (Maas et al., 2011) (long-length).** Dataset consists of 50,000 reviews from the Internet Movie Database (IMDb) labeled as positive or negative.

We use the following LLMs with temperature set to 0 for consistency of results: LLaMA3-70B (weaker model) and GPT-4o (stronger model). For each dataset, we randomly sample 100 reviews for our experiments – for our IMDB dataset, whenever the review exceeds the context length of LLaMA3-70B, we manually modify it to fit into the context length (we also run GPT-4o on this modified dataset). Details of our sampled dataset are as follows:

- **Amazon.** Average of 10.86 words per review; our aim would be to seek the top-k most important words, where $k = 1, 2, 3$ (9.21%, 18.42%, 27.62% correspondingly).
- **SST2.** Average of 17.76 words per review; our aim would be to seek the top-k most important words, where $k = 1, 2, 3$ (5.63%, 11.26%, 16.89% correspondingly).
- **IMDB.** Average of 213.28 words per review; our aim would be to seek the top-k most important words, where $k = 3, 5$ (1.41%, 2.34% correspondingly).

We run the following approaches: CFP, CFS, and DP. For each approach, the same LLM is used throughout the entire approach. The prompts for the experiments are given in Appendices A (Amazon dataset), B (SST2 dataset), and C (IMDB dataset).

Decision changing rate. We introduce a metric to measure explainability: *decision-changing* rate (DCR), which is the average of the decision-changing scores defined below:

- For a given list of important words $\mathcal{S}$ and input text $\boldsymbol{x}$, for each word $w \in \mathcal{S}$, replace it in $\boldsymbol{x}$ with '{MASK}'.
- Ask the LLM (that initially gave the classification), to produce a modified review $\boldsymbol{x}''$ by replacing each sub-string '{MASK}' with a word or short phrase such that the overall sentiment of the movie review gets flipped. The LLM is also asked to make sure that no other replacements are made, and that the review flows coherently.
- Feed $\boldsymbol{x}''$ back into the LLM and ask for its classification. If the classification is flipped, then we set the decision-changing score to be 1. Otherwise, we set it to be 0.

4.2 Main Results

Our experimental results are displayed in Table 1 and Figure 2. As expected, the accuracy generally remains unchanged for a given dataset and LLM, and increasing k increases the DCR since there are more opportunities for the LLM to change the sentiment of the review. Furthermore, the DCR for LLaMA3-70B (weaker model) is generally higher than that of GPT-4o (stronger model) because a weaker model's classification decision is more easily swayed due to its weaker reasoning abilities.

We also observe that the smaller the proportion of k (over the word length of the input string), the lower the DCR – we can observe this by looking at how DCR_3 decreases across Amazon (short), SST2 (medium), and IMDB (long). This is because when proportion of k is bigger (like in Amazon), there are few words/phrases outside of the top-k words that can counter the effect of the replaced words.

Approach	Amazon (short)				SST2 (medium)				IMDB (long)		
	Acc.	DCR_1	DCR_2	DCR_3	Acc.	DCR_1	DCR_2	DCR_3	Acc.	DCR_3	DCR_5
CFP-L3	98%	0.82	0.92	0.96	94%	0.71	0.90	0.91	99%	0.70	0.75
CFS-L3	98%	0.74	0.88	0.93	96%	0.71	0.84	0.92	99%	0.71	0.80
DP-L3	98%	0.82	0.90	0.96	96%	0.70	0.82	0.91	99%	0.65	0.72
CFP-G4	98%	0.69	0.88	0.93	99%	0.69	0.82	0.83	98%	0.46	0.70
CFS-G4	98%	0.68	0.87	0.93	99%	0.65	0.84	0.85	96%	0.46	0.69
DP-G4	98%	0.69	0.87	0.93	99%	0.62	0.77	0.81	96%	0.44	0.68

Table 1: L3 refers to LLaMA3-70B and G4 refers to GPT-4o. The best DCR for each LLM and dataset is highlighted in blue. The subcript of DCR denotes k. Acc. refers to the average accuracy over the k's.

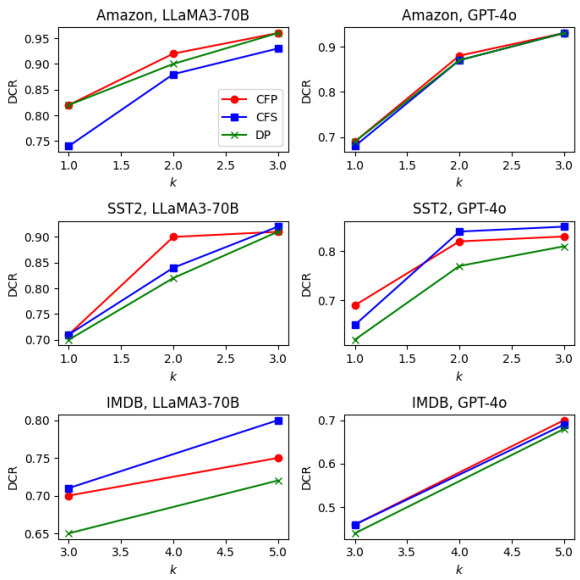

Figure 2: Visualization of the different approaches

However, when the proportion of k is smaller (like in IMDB), there are many words/phrases outside of the top-k words that can counter the effect of the replaced words. For example, consider a movie review with $k = 3$ and a negative sentiment, the review *'This movie is {MASK}, {MASK}, and {MASK}.'* is much easier to negate (to become positive) compared to the longer review *'This movie is {MASK}, {MASK}, and {MASK} – I am definitely not recommending it to anyone!'*

Finally, we have the following observations from Table 1 and Figure 2 for each dataset:

- For short-length inputs (Amazon), we observe that CFP performs the best, outperforming or matching DP's performances, whereas CFS performs equal to or worse than DP.
- For medium-length inputs (SST2), both CFP and CFS outperform DP for all cases, with CFP being the best performer 3 times and CFS being the best performer 4 times.
- For long-length inputs (IMDB), both CFP and CFS outperform DP for all cases, with CFP being the best performer 2 times and CFS being the best performer 3 times.

Overall, CFP is able to consistently outperform or match DP's performance. While CFS is also able to outperform and match DP's performance in most cases, there were a few cases where it underperformed, making it slightly inconsistent. Therefore, we would recommended the use of CFP over CFS or DP.

5 Conclusion

Our study shows that incorporating counterfactuals into LLM key-words identification can be useful under the framework that we have introduced. We can further extend our approach to multi-class classification problems by reducing them into two classes – e.g., (positive, neutral, negative) can be reduced to (positive, not positive). Specifically, if the predicted sendiment is positive, then we run CFP on the classes (positive, not positive) and the output would be the top-k words that led to a positive classification. We leave the exploration of this idea as a future work.

6 Limitations

While we believe that our choice of LLM should be sufficient, we acknowledge that we did not exhaustively evaluate a large selections of LLMs. Regarding the decision-changing score, we notice that despite our best efforts to be explicit to the LLM (and also setting the temperature to 0) when asking it to *only* replace the masked words, the LLM sometimes (but rarely) replace some other non-mask words. This injects some minor noise to the DCR calculation.

References

Chirag Agarwal, Sree Harsha Tanneru, and Himabindu Lakkaraju. 2024. Faithfulness vs. plausibility: On the (un) reliability of explanations from large language models. *arXiv preprint arXiv:2402.04614*.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, et al. 2025. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*.

Zahra Dehghanighobadi, Asja Fischer, and Muhammad Bilal Zafar. 2025. Can LLMs explain themselves counterfactually? *arXiv preprint arXiv:2502.18156*.

Jonathan Ichikawa and Matthias Steup. 2014. The analysis of knowledge. *Stanford Encylopedia of Philosophy*.

Dimitrios Kotzias, Misha Denil, Nando De Freitas, and Padhraic Smyth. 2015. From group to individual labels using deep features. *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 597–606.

Yinheng Li, Shaofei Wang, Han Ding, and Hang Chen. 2023a. Large language models in finance: A survey. *Proceedings of the fourth ACM international conference on AI in finance*, pages 374–382.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023b. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Haoyan Luo and Lucia Specia. 2024. From understanding to utilization: A survey on explainability for large language models. *arXiv preprint arXiv:2401.12874*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 142–150.

Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 115–124.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?" explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Hajar Sakai and Sarah S Lam. 2025. Large language models for healthcare text classification: A systematic review. *arXiv preprint arXiv:2503.01159*.

Nelvin Tan, Zian Seng, Liang Zhang, Yu-Ching Shih, Dong Yang, and Amol Salunkhe. 2025. Improved LLM agents for financial document question answering. *arXiv preprint arXiv:2506.08726*.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31:841.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024a. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15:1–38.

Hang Zhao, Qile P Chen, Yijing Barry Zhang, and Gang Yang. 2024b. Advancing single-and multi-task text classification through large language model fine-tuning. *arXiv preprint arXiv:2412.08587*.

A Prompt for Amazon Dataset

We provide the prompts used for the Amazon dataset. The user's prompts are colored in red, and the LLM's outputs are colored in black. We present the case of $k = 3$.

Prompts for DP:

```
Given the customer review '{review}', your task is to classify it either
↪ as 'positive' or 'negative', and state the top 3 most important
↪ words that led to you decision. Please follow the following output
↪ format and produce nothing else: <class,word1,word2,word3>
```

```
<classification1,DP word1,DP word2,DP word3>
```

Prompts for CFP:

```
Given the customer review '{review}', your task is to classify it
↪ either as 'positive' or 'negative' (and nothing else). Please only
↪ produce the classification and enclose it within <new> and </new>
↪ tags (i.e., <new>positive</new> or <new>negative</new>).
```

```
<new>{classification1}</new>
```

```
Customer reviews are classified as either positive or negative. The
↪ customer review '{review}' is classified as 'negative'. Your task
↪ is to modify the review with minimal edits to flip the sentiment
↪ prediction. Please only produce the modified review and enclose it
↪ within <new> and </new> tags. Avoid making any unnecessary changes,
↪ and don't produce anything else.
```

```
<new>{counterfactual}</new>
```

```
Given the customer review '{counterfactual}', your task is to classify
↪ it either as 'positive' or 'negative' (and nothing else). Please
↪ only produce the classification and enclose it within <new> and
↪ </new> tags.
```

```
<new>{classification2}</new>
```

```
A counterfactual of a customer review x1, is another customer review x2
↪ that is modified from x1 with a minimal number of edits such that
↪ the classification of x2 is different from x1. Given a customer
↪ review x1 that states <{review}> and its counterfactual x2 that
↪ states <{counterfactual}>, your task is to study x1 and x2
↪ carefully, and state the top 3 most important words (that are not
↪ stop words) from x1 that led to your previous decision of
↪ '{classification1}'. Please follow the following output format and
↪ produce nothing else: [word1,word2,word3]
```

```
{top-3 words}
```

Prompts for CFS:

```
Given the customer review '{review}', your task is to classify it either
↪ as 'positive' or 'negative', and state the top 3 most important
↪ words that led to you decision. Please follow the following output
↪ format and produce nothing else: <class,word1,word2,word3>
```

```
<classification1,DP word1,DP word2,DP word3>
```

```
Customer reviews are classified as either positive or negative. The
↪ customer review '{review}' is classified as '{classification1}'.
↪ Your task is to modify the review with minimal edits to flip the
↪ sentiment prediction. Please only produce the modified review and
↪ enclose it within <new> and </new> tags. Avoid making any
↪ unnecessary changes, and don't produce anything else.
```

```
<new>{counterfactual}</new>
```

```
Given the customer review '{counterfactual}', your task is to classify
↪ it either as 'positive' or 'negative' (and nothing else). Please
↪ only produce the classification and enclose it within <new> and
↪ </new> tags.
```

```
<new>{classification2}</new>
```

```
A counterfactual of a customer review x1, is another customer review x2
↪ that is modified from x1 with minimal number of edits such that the
↪ classification of x2 is different from x1. Review your previous
↪ selection of the 3 most important words in x1 below with relation
↪ to its classification (also stated below), using the initial
↪ customer review x1 and the counterfactual customer review x2. If
↪ you are confident about your answer, maintain your answer.
↪ Otherwise, update your answer. Note that the words must only come
↪ from x1, and never from x2. Please follow the following output
↪ format and produce nothing else: [word1,word2,word3]
### initial customer review x1 with rating '{classification1}'
{review}
### counterfactual customer review x2 with rating '{classification2}'
{counterfactual}
### 3 most important words from x1
[DP word1,DP word2,DP word3]
```

```
[CF word1,CF word2,CF word3]
```

Prompts for obtaining decision-changing score:

```
Given the review '{masked review}', for each sub-string '{MASK}' in the
↪ review, replace it with a word or short phrase such that the
↪ overall sentiment of the review becomes 'positive'. Also make sure
↪ that the sentence flows coherently. Please only produce the
↪ modified review and enclose it within <new> and </new> tags. Avoid
↪ making any unnecessary changes, never change any part of the review
↪ that is not '{MASK}', and don't produce anything else.
```

```
<new>{new review}</new>
```

```
Given the customer review '{new review}', your task is to classify it
↪ either as 'positive' or 'negative' (and nothing else). Please only
↪ produce the classification and enclose it within <new> and </new>
↪ tags (i.e., <new>positive</new> or <new>negative</new>).
```

```
<new>{new classification}</new>
```

B Prompt for SST2 Dataset

We provide the prompts used for the SST2 dataset. The user's prompts are colored in red, and the LLM's outputs are colored in black. We present the case of $k = 3$.

Prompts for DP:

```
Given the movie review '{review}', your task is to classify it either
↪ as 'positive' or 'negative', and state the top 3 most important
↪ words that led to you decision. Please follow the following output
↪ format and produce nothing else: <class,word1,word2,word3>
```

```
<classification1,DP word1,DP word2,DP word3>
```

Prompts for CFP:

```
Given the movie review '{review}', your task is to classify it either
↪  as 'positive' or 'negative' (and nothing else). Please only produce
↪  the classification and enclose it within <new> and </new> tags
↪  (i.e., <new>positive</new> or <new>negative</new>).
```

```
<new>{classification1}</new>
```

```
Movie reviews are classified as either positive or negative. The movie
↪  review '{review}' is classified as 'negative'. Your task is to
↪  modify the review with minimal edits to flip the sentiment
↪  prediction. Please only produce the modified review and enclose it
↪  within <new> and </new> tags. Avoid making any unnecessary changes,
↪  and don't produce anything else.
```

```
<new>{counterfactual}</new>
```

```
Given the movie review '{counterfactual}', your task is to classify it
↪  either as 'positive' or 'negative' (and nothing else). Please only
↪  produce the classification and enclose it within <new> and </new>
↪  tags.
```

```
<new>{classification2}</new>
```

```
A counterfactual of a movie review x1, is another movie review x2 that
↪  is modified from x1 with a minimal number of edits such that the
↪  classification of x2 is different from x1. Given a movie review x1
↪  that states <{review}> and its counterfactual x2 that states
↪  <{counterfactual}>, your task is to study x1 and x2 carefully, and
↪  state the top 3 most important words (that are not stop words) from
↪  x1 that led to your previous decision of '{classification1}'.
↪  Please follow the following output format and produce nothing else:
↪  [word1,word2,word3]
```

```
{top-3 words}
```

Prompts for CFS:

```
Given the movie review '{review}', your task is to classify it either
↪  as 'positive' or 'negative', and state the top 3 most important
↪  words that led to you decision. Please follow the following output
↪  format and produce nothing else: <class,word1,word2,word3>
```

```
<classification1,DP word1,DP word2,DP word3>
```

```
Movie reviews are classified as either positive or negative. The movie
↪  review '{review}' is classified as '{classification1}'. Your task
↪  is to modify the review with minimal edits to flip the sentiment
↪  prediction. Please only produce the modified review and enclose it
↪  within <new> and </new> tags. Avoid making any unnecessary changes,
↪  and don't produce anything else.
```

```
<new>{counterfactual}</new>
```

```
Given the movie review '{counterfactual}', your task is to classify it
↪  either as 'positive' or 'negative' (and nothing else). Please only
↪  produce the classification and enclose it within <new> and </new>
↪  tags.
```

```
<new>{classification2}</new>
```

```
A counterfactual of a movie review x1, is another movie review x2 that
↪  is modified from x1 with minimal number of edits such that the
↪  classification of x2 is different from x1. Review your previous
↪  selection of the 3 most important words in x1 below with relation
↪  to its classification (also stated below), using the initial movie
↪  review x1 and the counterfactual movie review x2. If you are
↪  confident about your answer, maintain your answer. Otherwise,
↪  update your answer. Note that the words must only come from x1, and
↪  never from x2. Please follow the following output format and
↪  produce nothing else: [word1,word2,word3]
### initial movie review x1 with rating '{classification1}'
{review}
### counterfactual movie review x2 with rating '{classification2}'
{counterfactual}
### 3 most important words from x1
[DP word1,DP word2,DP word3]
```

```
[CF word1,CF word2,CF word3]
```

Prompts for obtaining decision-changing score:

```
Given the movie review '{masked review}', for each sub-string '{MASK}'
↪  in the movie review, replace it with a word or short phrase such
↪  that the overall sentiment of the movie review becomes
↪  '{classification2}'. Also make sure that the sentence flows
↪  coherently. Please only produce the modified review and enclose it
↪  within <new> and </new> tags. Avoid making any unnecessary changes,
↪  never change any part of the review that is not '{MASK}', and don't
↪  produce anything else.
```

```
<new>{new review}</new>
```

```
Given the movie review '{new review}', your task is to classify it
↪  either as 'positive' or 'negative' (and nothing else). Please only
↪  produce the classification and enclose it within <new> and </new>
↪  tags (i.e., <new>positive</new> or <new>negative</new>).
```

```
<new>{new classification}</new>
```

C Prompt for IMDB Dataset

We provide the prompts used for the IMDB dataset. The user's prompts are colored in red, and the LLM's outputs are colored in black. We present the case of $k = 3$.

Prompts for DP:

```
Given the movie review below, your task is to classify it either as
↪  'positive' or 'negative', and state the top 3 most important words
↪  that led to you decision. Please follow the following output format
↪  and produce nothing else:
↪  <new>classification,word1,word2,word3</new>
### movie review
{review}
```

```
<new>classification1,DP word1,DP word2,DP word3,DP word4,DP word5</new>
```

Prompts for CFP:

```
Given the movie review below, your task is to classify it either as
↪  'positive' or 'negative' (and nothing else). Please only produce
↪  the classification and enclose it within <new> and </new> tags
↪  (i.e., <new>positive</new> or <new>negative</new>)).
### movie review
{review}
```

```
<new>{classification1}</new>
```

```
Movie reviews are classified as either positive or negative. The movie
↪  review below is classified as 'negative'. Your task is to modify
↪  the review with minimal edits to flip the sentiment prediction.
↪  Please only produce the modified movie review, and don't produce
↪  anything else.
### movie review
{review}
```

```
{counterfactual}
```

```
Given the movie review below, your task is to classify it either as
↪  'positive' or 'negative' (and nothing else). Please only produce
↪  the classification and enclose it within <new> and </new> tags.
### movie review
{counterfactual}
```

```
<new>{classification2}</new>
```

```
A counterfactual of a movie review x1, is another movie review x2 that
↪  is modified from x1 with a minimal number of edits such that the
↪  classification of x2 is different from x1. Given a strings x1 and
↪  x2 below, your task is to study x1 and x2 carefully, and state the
↪  top 3 most important words (that are not stop words) from x1 that
↪  led to your previous decision of 'negative'. Please follow the
↪  following output format and produce nothing else:
↪  [word1,word2,word3]
### x1:
{review}
### x2:
{counterfactual}
```

```
[word1,word2,word3]
```

Prompts for CFS:

```
Given the movie review below, your task is to classify it either as
↪  'positive' or 'negative', and state the top 3 most important words
↪  that led to you decision. Please follow the following output format
↪  and produce nothing else:
↪  <new>classification,word1,word2,word3</new>
### movie review
{review}
```

```
<new>classification1,DP word1,DP word2,DP word3</new>
```

```
Movie reviews are classified as either positive or negative. The movie
↪  review below is classified as '{classification1}'. Your task is to
↪  modify the review with minimal edits to flip the sentiment
↪  prediction. Please only produce the modified movie review, and
↪  don't produce anything else.
### movie review
{review}
```

```
{counterfactual}
```

```
Given the movie review below, your task is to classify it either as
↪  'positive' or 'negative' (and nothing else). Please only produce
↪  the classification and enclose it within <new> and </new> tags.
### movie review
{counterfactual}
```

```
<new>{classification2}</new>
```

```
A counterfactual of a movie review x1, is another movie review x2 that
↪  is modified from x1 with minimal number of edits such that the
↪  classification of x2 is different from x1. Review your previous
↪  selection of the 3 most important words in x1 below with relation
↪  to its classification (also stated below), using the initial movie
↪  review x1 and the counterfactual movie review x2. If you are
↪  confident about your answer, maintain your answer. Otherwise,
↪  update your answer. Note that the words must only come from x1, and
↪  never from x2. Please follow the following output format and
↪  produce nothing else: [word1,word2,word3]
### initial movie review x1 with rating '{classification1}'
{review}
### counterfactual movie review x2 with rating '{classification2}'
{counterfactual}
### 3 most important words from x1
[DP word1,DP word2,DP word3]
```

```
[CF word1,CF word2,CF word3]
```

Prompts for obtaining decision-changing score:

```
Given the movie review below, for each sub-string '{MASK}' in the movie
↪  review, replace it with a word or short phrase such that the
↪  overall sentiment of the movie review becomes '{classification2}'.
↪  Ensure that your replacement makes the sentence flow coherently,
↪  but do not change anything else in the movie review (except for the
↪  '{MASK}' sub-strings). Please only produce the modified movie
↪  review, and don't produce anything else.
### movie review
{masked review}
```

```
{new review}
```

```
Given the movie review below, your task is to classify it either as
↪  'positive' or 'negative' (and nothing else). Please only produce
↪  the classification and enclose it within <new> and </new> tags
↪  (i.e., <new>positive</new> or <new>negative</new>).
### movie review
{new review}
```

```
<new>{new classification}</new>
```