

A PRELIMINARIES

A.1 MAXIMUM FLOW

Definition A.1 (Network Flow). Given a network $G = (V, E, s, t, \mathbf{u})$, where s and t are the source and target nodes respectively and u_{ij} is the capacity for the edge $(i, j) \in E$, a flow is characterized as a function $f : E \rightarrow \mathbb{R}^{\geq 0}$ s.t.

$$\begin{aligned} f_{ij} &\leq u_{ij} \quad \forall (i, j) \in E \quad (\text{capacity constraints}) \\ \sum_{j:(i,j) \in E} f_{ij} - \sum_{j:(j,i) \in E} f_{ji} &= 0, \quad \forall i \in V, i \neq s, t \quad (\text{flow conservation constraints}) \end{aligned} \quad (17)$$

We define $|f_{\text{out}}(i)|$ to be the total outflow value of a node i and $|f_{\text{in}}(i)|$ to be the total inflow value of a node i . For a given set $K \subseteq V$ of nodes, we define $|f(K)| = \sum_{i \in K} |f_{\text{out}}(i)|$ for every flow f . The value of a flow in a given network $G = (V, E, s, t, \mathbf{u})$ is denoted as $|f| = \sum_{v:(s,v)} f_{sv} - \sum_{v:(v,s)} f_{vs} = |f_{\text{out}}(s)| - |f_{\text{in}}(s)|$, and a maximum flow is identified as a feasible flow with the highest attainable value.

A.2 MULTI-COMMODITY MAXIMUM FLOW

The multi-commodity flow problem is an important variant of the maximum flow problem. This problem involves multiple source-sink pairs, unlike the standard maximum flow problem, which only considers one source and one sink. The goal is to find multiple optimal flows, denoted by $f^1(\cdot, \cdot), \dots, f^r(\cdot, \cdot)$, where each $f^k(\cdot, \cdot)$ represents a feasible flow from the source s_k to the sink t_k . The objective is to ensure that all capacity constraints are satisfied, which are represented by the equation:

$$\sum_{k=1}^r f^k(i, j) \leq u(i, j) \quad \forall (i, j) \in E \quad (18)$$

Such a flow is known as a "multi-commodity" flow. A multi-commodity maximum flow problem is to maximize the function $\sum_{k=1}^r \sum_{v:(v,s_k)} f^k(s_k, v)$.

To solve the problem of multi-commodity maximum flow, we can simplify it by transforming it into a standard maximum flow problem. This can be achieved by introducing two new nodes, a "super-source" node ss and a "super-target" node st . The "super-source" node ss should be connected to all the original sources s_i through edges of finite capacities, while the "super-target" node st should be connected to all the original sinks t_i with edges of finite capacities:

- Each outgoing edge from the "super-source" node ss to each source node s_i gets assigned a capacity that is equal to the total capacity of the outgoing edges from the source node s_i .
- Each incoming edge from an original "super-target" node st to each sink node t_i gets assigned a capacity that is equal to the total capacity of the incoming edges to the sink node t_i .

It is easy to demonstrate that the maximum flow from ss to st is equivalent to the maximum sum of flows in a feasible multi-commodity flow in the original network.

A.3 SHAPLEY VALUES

The Shapley value, introduced by [Shapley \(1952\)](#), concerns the cooperative game in the coalitional form (N, ϑ) , where N is a set of n players and $\vartheta : 2^N \rightarrow \mathbb{R}$ with $\vartheta(\emptyset) = 0$ is the characteristic (payoff) function. In the game, the marginal contribution of the player i to any coalition S with $i \notin S$ is considered as

$\vartheta(S \cup i) - \vartheta(S)$. These Shapley values are the only constructs that jointly satisfy the efficiency, symmetry, nullity, and additivity axioms (Shapley, 1952; Young, 1985):

Efficiency: The Shapley values must add up to the total value of the game, which means $\sum_{i \in N} \phi_i(\vartheta) = \vartheta(N)$.

Symmetry: If two players are equal in their contributions to any coalition, they should receive the same Shapley value. Mathematically, if $\vartheta(S \cup \{i\}) = \vartheta(S \cup \{j\})$ for all $S \subseteq N \setminus \{i, j\}$, then $\phi_i(\vartheta) = \phi_j(\vartheta)$.

Nullity (Dummy): If a player has no impact on any coalition, their Shapley value should be zero. Mathematically, if $\vartheta(S \cup \{i\}) = \vartheta(S)$ for all $S \subseteq N \setminus \{i\}$, then $\phi_i(\vartheta) = 0$.

Linearity: If the game $\vartheta(\cdot)$ is a linear combination of two games $\vartheta_1(\cdot), \vartheta_2(\cdot)$ for all $S \subseteq N$, i.e. $\vartheta(S) = \vartheta_1(S) + \vartheta_2(S)$ and $(c \cdot \vartheta)(S) = c \cdot \vartheta(S), \forall c \in \mathbb{R}$, then the Shapley value in the game ϑ is also a linear combination of that in the games ϑ_1 and ϑ_2 , i.e. $\forall i \in N, \phi_i(\vartheta) = \phi_i(\vartheta_1) + \phi_i(\vartheta_2)$ and $\phi_i(c \cdot \vartheta) = c \cdot \phi_i(\vartheta)$.

Considering these axioms, the attribution of a player j is uniquely given by (Shapley, 1952; Young, 1985):

$$\phi_j(\vartheta) = \sum_{S \subseteq N \setminus \{j\}} \frac{|S|!(n - |S| - 1)!}{n!} (\vartheta(S \cup \{j\}) - \vartheta(S)) \quad (19)$$

where the difference $\vartheta(S \cup \{j\}) - \vartheta(S)$ represents the j -th feature's contribution to the subset S , and the summation represents a weighted average across all subsets that do not include j .

Initially, a payoff function based on model accuracy was suggested (Lundberg & Lee, 2017). Since then, various alternative coalition functions have been proposed (Jethani et al., 2022; Sundararajan & Najmi, 2020), each resulting in a different feature importance score. Many of these alternative approaches are widely used and have been shown to outperform the basic SHAP method (Lundberg & Lee, 2017) in empirical studies.

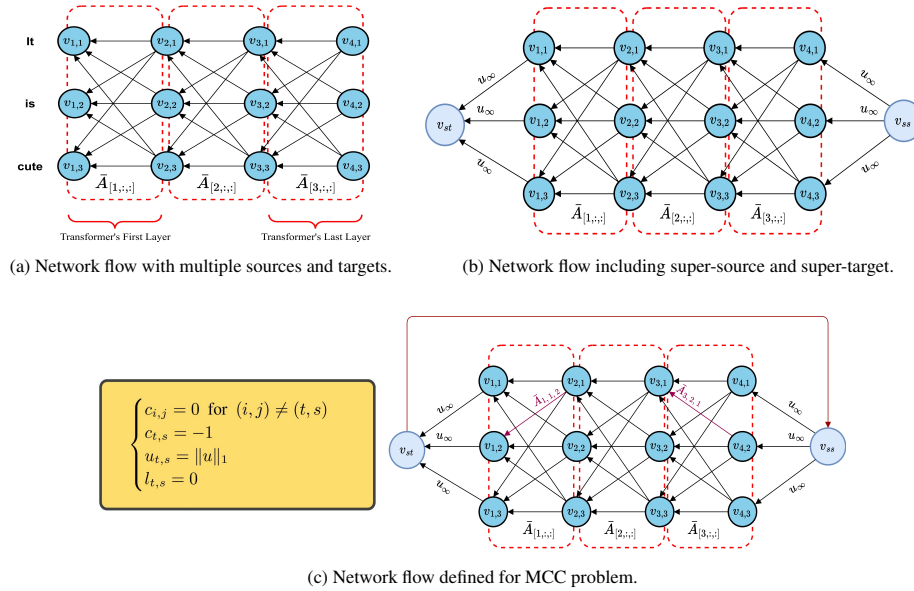


Figure 2: Initial network flow to be used in our proposed method, the multi-commodity flow with multiple sources and targets, and the network flow for MCC problems.

B NETWORK FLOW GENERATION

Fig. 2 will detail the process of defining a graph network and its parameters using Algorithm 1 for use in our proposed method. It is worth noting that the same method can be used for Algorithm 2. To solve the maximum flow or MCC problem within this graph network, we must compute network flow with multiple sources and targets, assigning all nodes in the first and last layers of transformers as sources and targets, respectively (Fig. 2a).

To solve this problem, we leverage the concept of multi-commodity flow (multiple-sources multiple-targets maximum flow) by introducing a super-source node ss and a super-target node st (Fig. 2b). To define the upper-bound and lower-bound capacities of this new graph network, we utilize the procedure defined in Sec. 3.2. In the last step, we add a new edge from the super-target node st to the super-source node ss and define the cost vector, upper-bound capacities, and lower-band capacities according to Fig. 2c. Subsequently, we can input all derived parameters into eq. 12, solve the optimization problem, and evaluate feature attributions.

C NON-UNIQUENESS OF MAXIMUM FLOW

Fig. 3 visually describes our proposed approach for computing feature attributions. Using maximum flow to derive these attributions produces a convex set containing all optimal flows, which makes it unsuitable as a feature attribution technique. In contrast, our proposed approach, which utilizes the log barrier method, generate a unique optimal flow and provides an interpretable set of feature attributions.

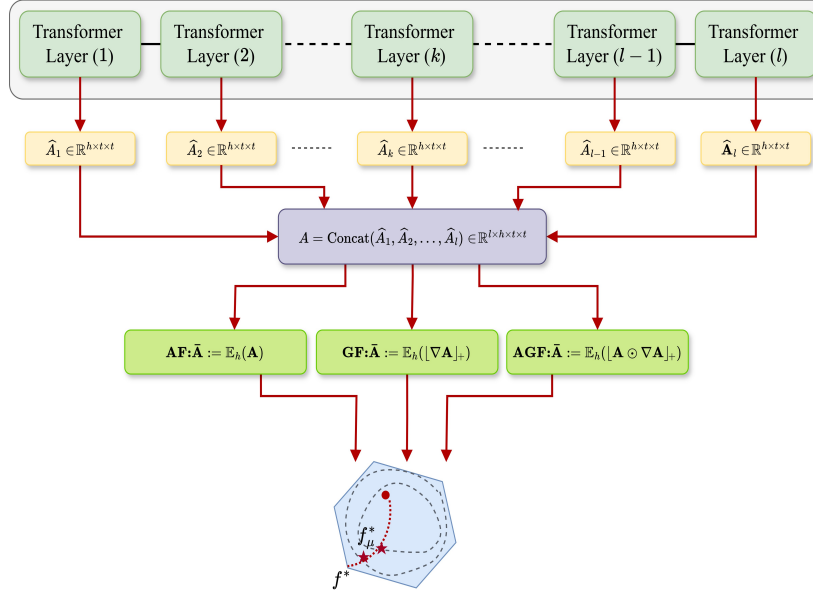


Figure 3: Overview of how the proposed method evaluates the unique optimal flow computed using the log barrier method, attention weights, and their gradients in Transformers.

Fig. 4 shows the capacities and optimal flows obtained by solving maximum flows on the network, constructed with the synthetic information tensor $\tilde{A} \in \mathbb{R}^{4 \times 3 \times 3}$ as input, using Algorithm 1 and Algorithm 2. While the maximum flows are the same for both algorithms, their optimal flows differ. Notably, significant differences in

flows between node pairs $\{v_4, v_8\}$ and $\{v_3, v_5\}$ are visible in Fig. 4c and Fig. 4d. Additionally, we evaluated the maximum flow and its optimal flow generated by both algorithms across various combinations of token numbers t and Transformer layers l . Our findings indicate that the optimal flows from the two algorithms do not coincide in any scenario.

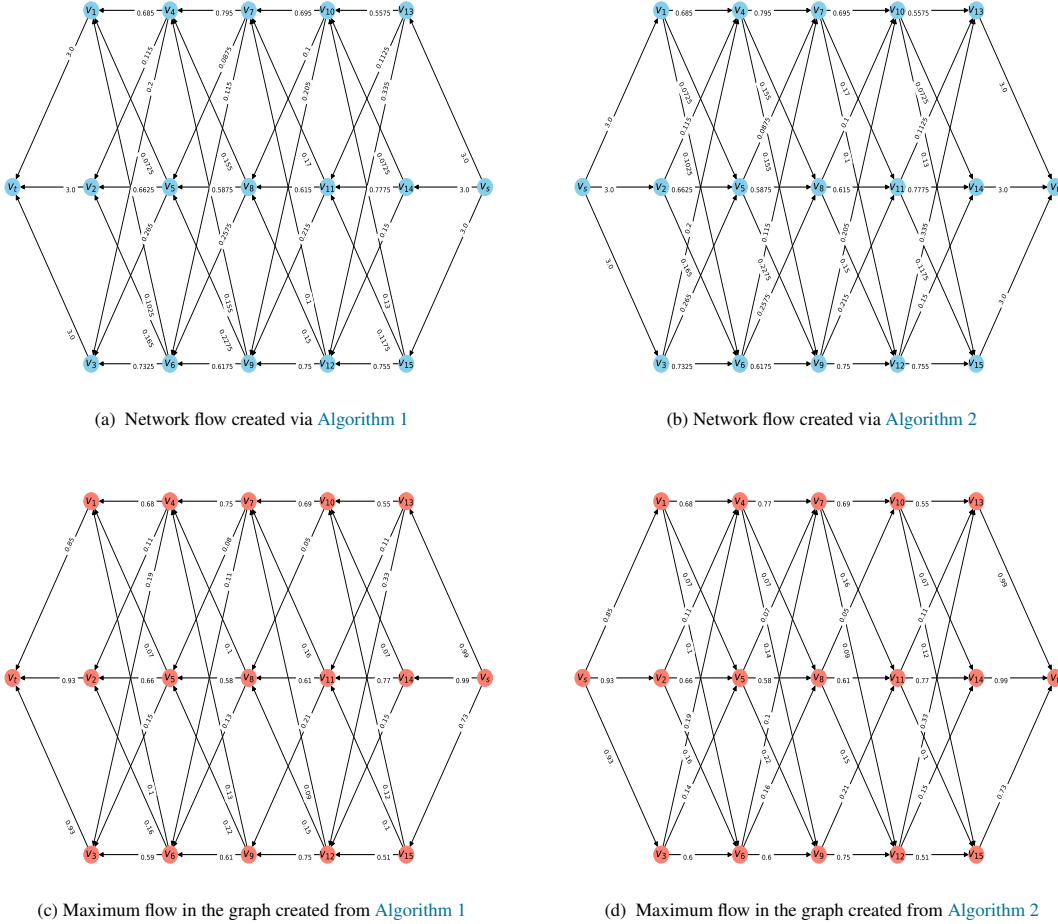
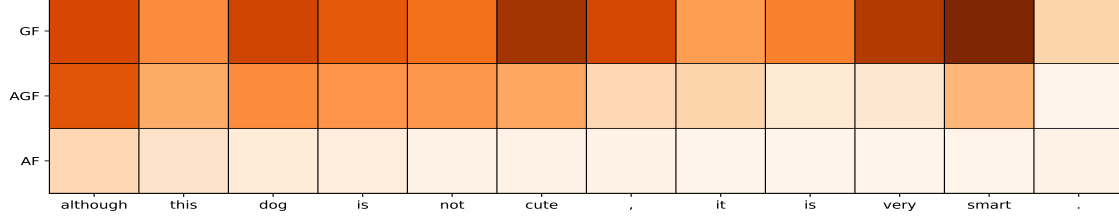


Figure 4: Network flow, maximum flow, and residual flow created by Algorithm 1 and Algorithm 2. The optimal flows and residual flows evaluated using Algorithm 1 and Algorithm 2 are different.

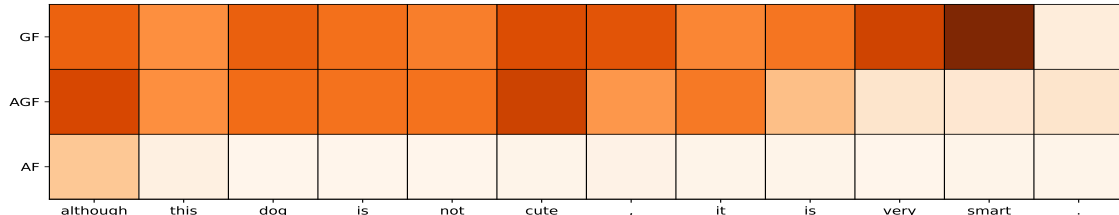
Fig. 5a and Fig. 5b display the normalized feature attributions evaluated across three information tensors introduced in Sec. 3.1 for sentiment analysis of the sentence "although this dog is not cute, it is very smart." employing both Algorithm 1 and Algorithm 2. Across each of the three information tensors, the resulting optimal flows and their corresponding normalized attributions differ depending on whether Algorithm 1 or Algorithm 2 is used.

The layer-wise normalized feature attributions, obtained through the same process, are displayed in Fig. 6. For each information tensor type and layer, the resulting optimal flows and their normalized attributions differ based on whether Algorithm 1 or Algorithm 2 is utilized. We also computed the optimal flow and its feature

attributions for various input sentences using both algorithms for each of the information tensors AF, GF, and AGF. Our findings reveal that the optimal flows and corresponding feature attributions generated by Algorithm 1 and Algorithm 2 differ for all sentences.

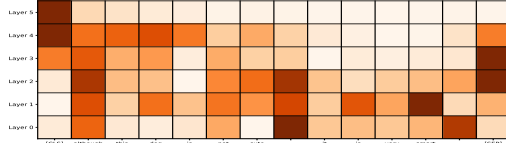


(a) Normalized feature attributions for Transformer's input layer evaluated by Algorithm 1 for different information tensors.

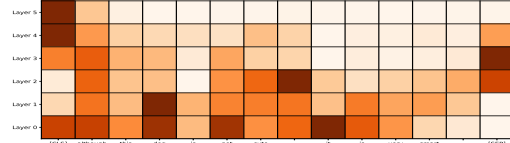


(b) Normalized feature attributions for Transformer's input layer evaluated by Algorithm 2 for different information tensors.

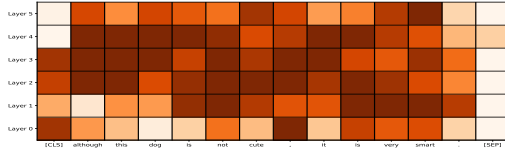
Figure 5: Normalized feature attributions for Transformer's input layer and different information tensors.



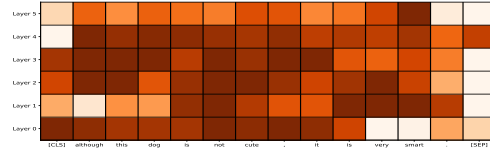
(a) AF method: Algorithm 1



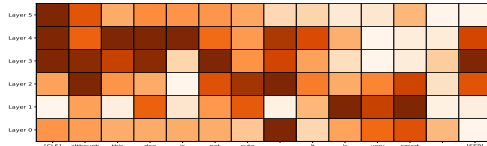
(b) AF method: Algorithm 2.



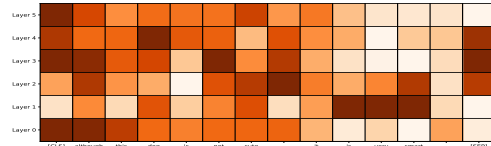
(c) GF method: Algorithm 1.



(d) GF method: Algorithm 2.



(e) AGF method: Algorithm 1.



(f) AGF method: Algorithm 2.

Figure 6: Normalized feature attributions for all Transformer layers evaluated by Algorithm 1 and Algorithm 2.

D RESULTS

D.1 QUALITATIVE VISUALIZATIONS

This section visually examines the feature attributions derived from our proposed methods, applied to information tensors as defined in Sec. 3.1. Fig. 7 illustrates feature attributions obtained from our proposed methods applied to two graphs generated by either Algorithm 1 or Algorithm 2. Remarkably, our approaches consistently yield identical results for both graphs. The outcomes vividly demonstrate the superiority of AGF over AF and GF, offering more insightful and reasonable feature attributions. Specifically, both AGF and GF effectively highlight the importance of tokens like 'smart' and 'cute', while assigning lower values to less significant tokens such as 'this', 'it', and 'and'. In contrast, AF fails to capture the expected feature attribution of 'smart' and tends to produce an almost uniform distribution for feature attributions.

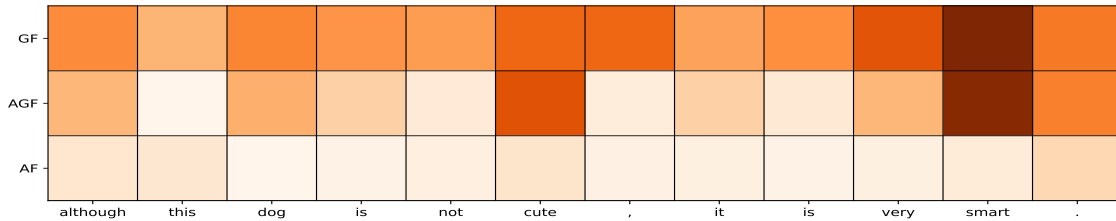


Figure 7: Visualizations of the feature attributions generated by running our proposed method on the three introduced information tensors on the showcase example.

D.2 ADDITIONAL RESULTS

Fig. 8 presents a detailed comparison of the performance dynamics of various feature attribution methods under different corruption rates across three distinct datasets: IMDB, Amazon, and Yelp. The efficacy of these methods is evaluated using two key metrics, AOPC and LOdds, which measure how well each method identifies important tokens that influence model predictions. Notably, our proposed AGF method consistently outperforms the other techniques, maintaining the highest average AOPC and LOdds scores across a range of corruption levels, particularly for both the IMDB and Amazon datasets. This consistent superiority highlights the robustness of AGF in pinpointing the most important tokens, which significantly affect the model's decision-making process.

We also evaluated various explanation methods by analyzing their performance on classification metrics, with results summarized in Tab. 3, which details the average Accuracy, F1, Precision, and Recall scores across multiple k values. On the SST2 dataset, our proposed methods AGF and GD, in conjunction with KernelShap, achieved the highest overall performance. For the IMDB dataset, AGF and GF, along with Integrated Gradients (IG), showed significant improvement over other methods. In the Amazon dataset, AGF and GF, when combined with TransAtt, outperformed all competitors. For the AG News dataset, AGF and AF, paired with AttGrads, demonstrated superior performance. The consistently strong performance of AGF across diverse datasets and evaluation metrics highlight its versatility and effectiveness in accurately identifying important tokens, reinforcing its reliability as a feature attribution method in NLP models.

However, the Yelp dataset poses a distinct challenge where our proposed methods, including AGF, do not consistently achieve optimal results across all evaluation metrics. This performance discrepancy is likely due to the unique characteristics inherent to the Yelp dataset, which frequently contains a higher concentration of informal language, colloquialisms, and typographical errors. The prevalence of such linguistic noise in Yelp reviews is notably higher compared to other datasets like IMDB or Amazon. These textual irregularities

introduce complexity that AGF, with its current configuration, may be less equipped to handle effectively. Consequently, AGF’s performance suffers, as it appears to have a lower tolerance for handling noisy or non-standard text inputs, which compromise its ability to accurately attribute features and identify the most influential tokens within these reviews.

Fig. 9 compare the feature attribution methods evaluated on the aforementioned sentence using a model trained on SST2. In all instances, the feature attribution methods predict positive sentiment for the showcased example. Our methods, AGF and GF, effectively capture the most important tokens, such as ‘cute’ and ‘smart’ (indicated in dark orange shading), which significantly contribute to the positive sentiment prediction. Some other methods, including Grads, LIME, RawAtt, and PartialLRP, also exhibit some capability in identifying important tokens. However, certain methods like AF, AttCAT, CAT, Rollout, KernelShap, and IG struggle to correctly identify important tokens.

Table 3: The average of F1, Accuracy, Precision, and Recall scores of all methods in explaining the Transformer-based model on each dataset when we mask **top** $k\%$ tokens. Lower scores are desirable for all metrics (indicated by \downarrow), indicating a strong ability to mark important tokens.

Methods	SST2				IMDB				Yelp				Amazon				AG News			
	F1 \downarrow	Acc \downarrow	Prec \downarrow	Rec \downarrow	F1 \downarrow	Acc \downarrow	Prec \downarrow	Rec \downarrow	F1 \downarrow	Acc \downarrow	Prec \downarrow	Rec \downarrow	F1 \downarrow	Acc \downarrow	Prec \downarrow	Rec \downarrow	F1 \downarrow	Acc \downarrow	Prec \downarrow	Rec \downarrow
RawAtt	0.75	0.75	0.72	0.79	0.69	0.67	0.70	0.69	0.68	0.72	0.74	0.63	0.67	0.68	0.67	0.66	0.65	0.68	0.68	0.68
Rollout	0.81	0.82	0.80	0.82	0.74	0.67	0.65	0.84	0.80	0.83	0.88	0.74	0.71	0.73	0.71	0.72	0.61	0.63	0.64	0.63
Grads	0.78	0.75	0.72	0.79	0.69	0.67	0.70	0.69	0.68	0.72	0.74	0.63	0.67	0.68	0.67	0.66	0.65	0.68	0.68	0.68
AttGrads	0.78	0.78	0.75	0.82	0.76	0.75	0.78	0.76	0.91	0.91	0.89	0.93	0.79	0.82	0.80	0.78	0.58	0.61	0.60	0.60
CAT	0.68	0.65	0.61	0.76	0.56	0.49	0.51	0.65	0.70	0.70	0.68	0.73	0.68	0.64	0.67	0.70	0.63	0.64	0.64	0.64
AttCAT	0.68	0.65	0.62	0.76	0.57	0.48	0.50	0.64	0.67	0.66	0.65	0.70	0.67	0.62	0.66	0.68	0.64	0.64	0.65	0.64
PartialLRP	0.75	0.75	0.71	0.78	0.66	0.65	0.67	0.67	0.65	0.70	0.71	0.61	0.65	0.66	0.65	0.64	0.65	0.68	0.68	0.68
TransAtt	0.73	0.72	0.69	0.76	0.61	0.58	0.60	0.62	0.62	0.66	0.66	0.60	0.62	0.63	0.63	0.61	0.63	0.66	0.66	0.66
LIME	0.61	0.63	0.62	0.63	0.55	0.55	0.55	0.55	0.72	0.73	0.72	0.73	0.72	0.73	0.72	0.73	0.67	0.67	0.68	0.67
KernelShap	0.53	0.52	0.53	0.53	0.67	0.69	0.68	0.69	0.77	0.78	0.77	0.78	0.67	0.68	0.67	0.68	0.74	0.74	0.75	0.74
IG	0.56	0.57	0.56	0.57	0.48	0.50	0.48	0.50	0.72	0.72	0.72	0.72	0.73	0.74	0.73	0.74	0.67	0.68	0.67	0.67
AF	0.72	0.72	0.71	0.71	0.65	0.68	0.70	0.68	0.71	0.71	0.71	0.70	0.69	0.71	0.71	0.71	0.64	0.62	0.65	0.64
GF	0.56	0.57	0.56	0.56	0.45	0.48	0.45	0.48	0.70	0.71	0.70	0.71	0.65	0.67	0.66	0.67	0.67	0.68	0.67	0.67
AGF	0.54	0.52	0.54	0.54	0.46	0.47	0.47	0.47	0.70	0.72	0.70	0.70	0.67	0.67	0.67	0.67	0.64	0.63	0.65	0.64

D.3 STATISTICAL SIGNIFICANCE TEST

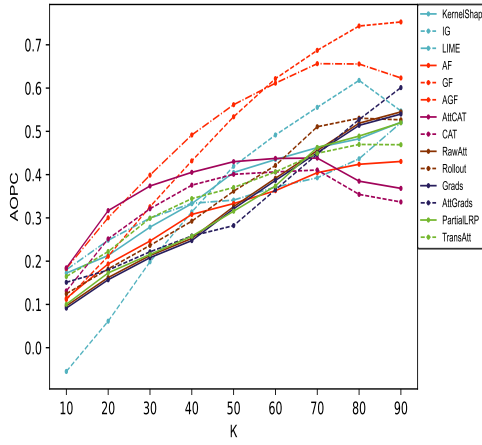
To implement a statistical significance test, we employ the ASO (Almost Stochastic Order) method (Ulmer et al., 2022; Dror et al., 2019; del Barrio et al., 2017), which compares the cumulative distribution functions (CDFs) of two score distributions to determine stochastic dominance. Notably, ASO imposes no assumptions about score distributions, making it applicable to any metric where higher scores indicate better performance.

When comparing model A with model B using the ASO method, we obtain the value ϵ_{\min} , which is an upper bound on the violation of stochastic order. If $\epsilon_{\min} \leq \tau$ (with $\tau \leq 0.5$), model A is considered stochastically dominant over model B , implying superiority. This value can also be interpreted as a confidence score; a lower ϵ_{\min} suggests greater confidence in model A ’s superiority. The null hypothesis for ASO is defined as:

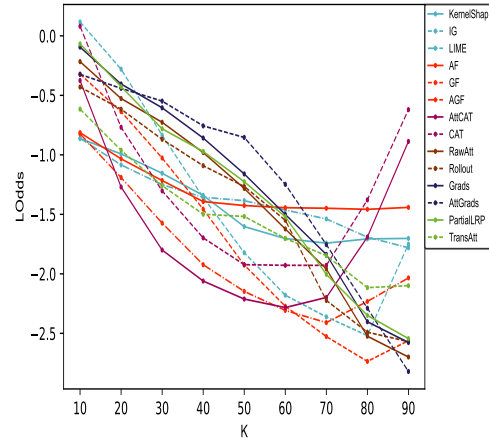
$$H_0 : \epsilon_{\min} \geq \tau \quad (20)$$

where the significance level α is an input parameter that influences ϵ_{\min} .

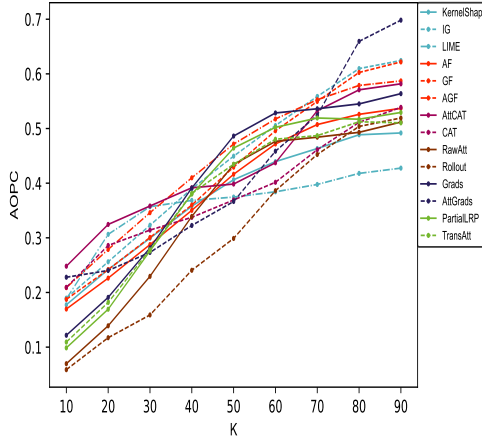
In this study, we conduct 500 independent runs per method to perform comprehensive statistical tests, comparing the AOPC and LOdds metrics between our best-performing proposed method, AGF, and the top benchmark methods outlined in Tab. 1 and Tab. 2, using $\tau = 0.5$. As illustrated in Tab. 4 and Tab. 5, the AGF method stochastically dominates the performance of these benchmark methods across all datasets, with the exception of the Yelp dataset.



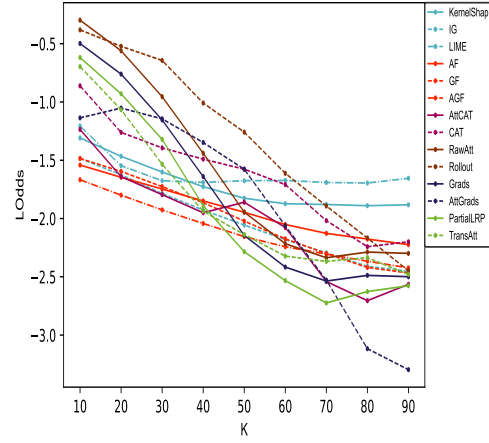
(a) AOPC score for IMDB dataset



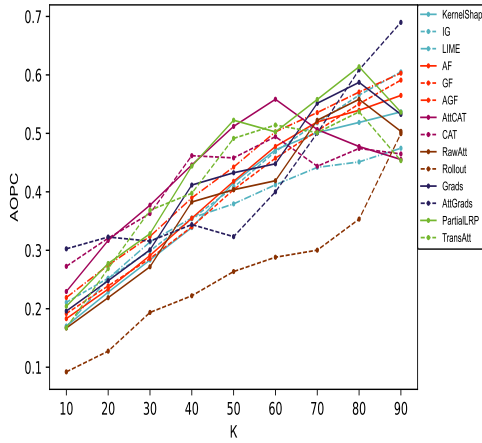
(b) LOdds score for IMDB dataset



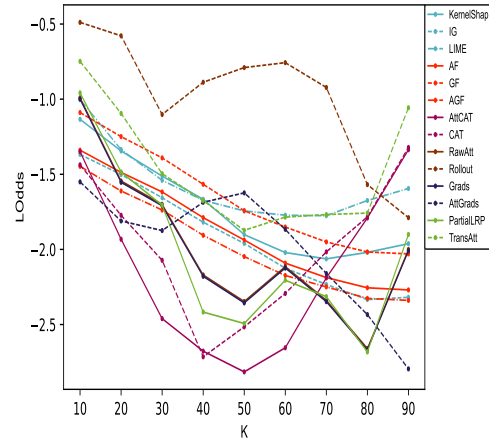
(c) AOPC score for Amazon dataset



(d) LOdds score for Amazon dataset



(e) AOPC score for Yelp dataset



(f) LOdds score for Yelp dataset

Figure 8: AOPC and LOdds scores of different methods in explaining BERT across the varying corruption rates k on IMDB, Amazon, and Yelp datasets. The x-axis illustrates masking the $k\%$ of the tokens in an order of decreasing saliency.

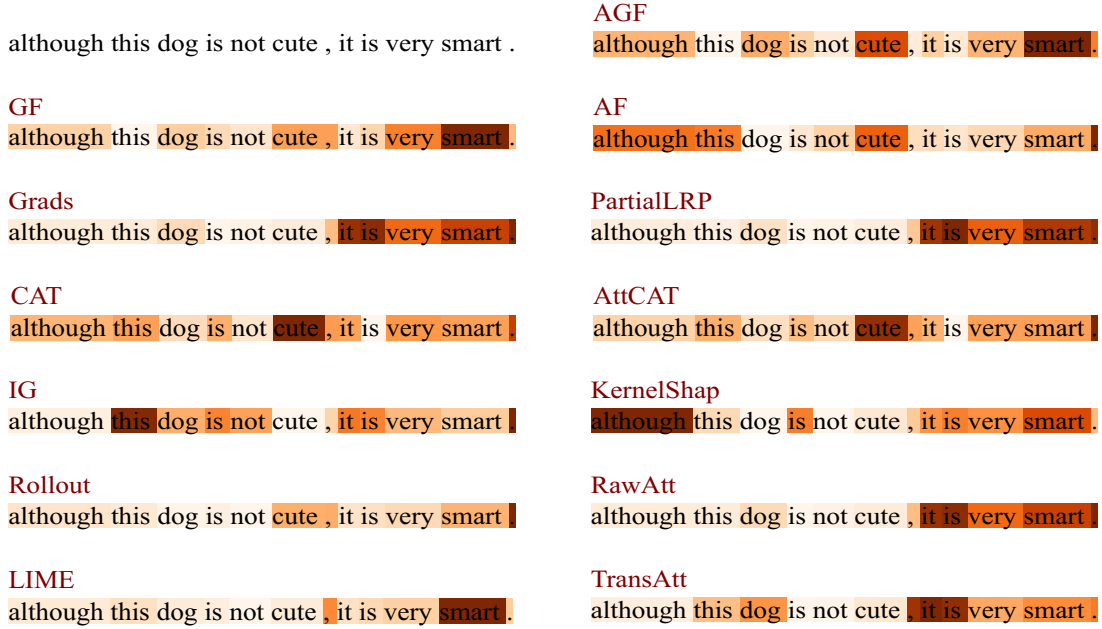


Figure 9: Visualizations of the normalized feature attributions generated by the methods on the binary classification task.

Table 4: ASO test to compare AGF method with the best benchmark method when we mask **top** $k\%$ tokens.

Dataset	SST2		IMDB		Yelp		Amazon		AG News	
	AOPC	LOdds	AOPC	LOdds	AOPC	LOdds	AOPC	LOdds	AOPC	LOdds
ϵ_{\min}	0.054	0.039	0.078	0.061	0.813	0.924	0.053	0.043	0.091	0.076

Table 5: ASO test to compare AGF method with the best benchmark method when we mask **bottom** $k\%$ tokens.

Dataset	SST2		IMDB		Yelp		Amazon		AG News	
	AOPC	LOdds	AOPC	LOdds	AOPC	LOdds	AOPC	LOdds	AOPC	LOdds
ϵ_{\min}	0.049	0.037	0.113	0.085	0.913	0.824	0.062	0.053	0.091	0.067

E PROOFS

Corollary 3.1. While Shapley values are inherently unique, our findings in [Sec. 3.3](#) and [App. C](#) expose a critical inconsistency. We demonstrate that the optimal solution of the maximum flow problems defined in [eq. 8](#) is not necessarily unique, thereby disproving the claim that the feature attributions proposed by [Ethayarajh & Jurafsky \(2021\)](#) are Shapley values.

The non-uniqueness of these attributions, as evidenced by our proof, fundamentally conflicts with the defining properties of Shapley values. If these attributions defined by [Ethayarajh & Jurafsky \(2021\)](#) were indeed Shapley values, they would necessarily be unique. However, our observations demonstrate that since the optimal solution of the maximum flow problem is not necessarily unique, for each optimal solution of the maximum flow problem, we can derive corresponding feature attributions that differ from one another.

□

Theorem 3.1. Since the optimal flow f^* is computed once for the entire graph and not for each potential subgraph, and the players (tokens) are all disjoint without any connections in S , blocking the flow through one player does not impact the outflow of any other players. Therefore, for every $S \subseteq N$ where $i \notin S$, we have $|f_{\text{out}}(i)| = v(S \cup \{i\}) - v(S)$. Utilizing the definition of Shapley values in eq. 14, we obtain:

$$\begin{aligned} \phi_i(\vartheta) &= \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (\vartheta(S \cup \{i\}) - \vartheta(S)) \\ &= \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (|f_{\text{out}}(i)|) \\ &= |f_{\text{out}}(i)| \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} \\ &= |f_{\text{out}}(i)| \end{aligned}$$

It is also evident that the defined function meets all four fairness-based axioms of efficiency, symmetry, linearity, and additivity. □

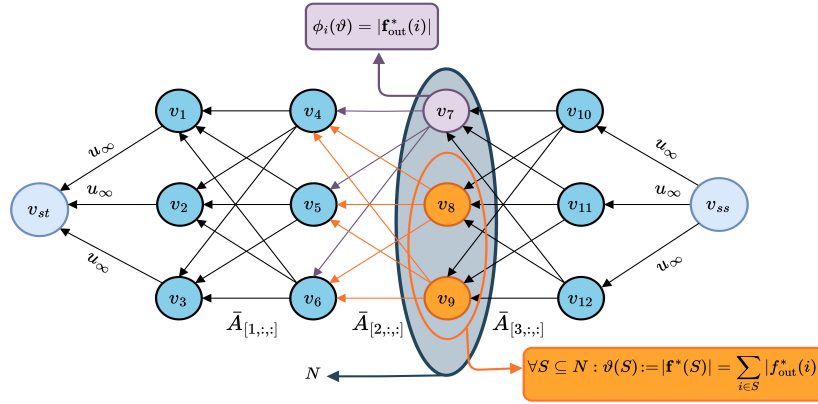


Figure 10: Visualizations of the procedure to define the cooperative game (N, ϑ) using the solution of barrier-regularized maximum flow or its corresponding MCC problem.

F IMPLEMENTATION DETAILS

F.1 DATASETS

Tab. 6 illustrates comprehensive statistics of the datasets utilized for the classification task. We randomly extracted 5,000 sentences from each test section of the datasets, except for those with a test size less than 5,000, where we retained all samples. Furthermore, we prioritized diversity in our sampling process by incorporating sentences of varying lengths, with an equal distribution between those shorter and longer than the mode size of the test dataset.

Table 6: Statistical information and the pre-trained models employed for each dataset.

Datasets	# Test Samples	# Classes	ℓ_{mode}	ℓ_{min}	ℓ_{max}	ℓ_{avg}	Pre-trained Model
SST2	1,821	2	108	5	256	103.3	textattack/bert-base-uncased-SST-2
Amazon	5,000	2	127	15	1009	404.9	fabriceyhc/bert-base-uncased-amazon_polarity
IMDB	5,000	2	670	32	12988	1293.8	fabriceyhc/bert-base-uncased-imdb
Yelp	5,000	2	313	4	5107	723.8	fabriceyhc/bert-base-uncased-yelp_polarity
AG News	5,000	4	238	100	892	235.3	fabriceyhc/bert-base-uncased-ag_news

F.2 TIME COMPLEXITY OF PROPOSED METHODS

In the minimum-cost circulation problem, we are given a directed graph $G = (V, E)$ with $|V| = n$ vertices and $|E| = m$ edges, upper and lower edge capacities $\mathbf{u}, \mathbf{l} \in \mathbb{R}^m$, and edge costs $\mathbf{c} \in \mathbb{R}^m$. Our objective is to find a circulation $\mathbf{f} \in \mathbb{R}^m$ satisfying:

$$\begin{aligned} \arg \min \quad & \mathbf{c}^\top \mathbf{f} \\ \text{subject to} \quad & \mathbf{B}^\top \mathbf{f} = \mathbf{0} \\ & \mathbf{l} \leq \mathbf{f} \leq \mathbf{u} \end{aligned}$$

where $\mathbf{B} \in \mathbb{R}^{m \times n}$ is the edge-vertex incidence matrix.

To compare running times, we assume that $\tilde{\mathbf{l}}, \tilde{\mathbf{u}}, \tilde{\mathbf{c}}$ represent the integral versions of $\mathbf{l}, \mathbf{u}, \mathbf{c}$ obtained from either Algorithm 1 or Algorithm 2, and define $U = \|\tilde{\mathbf{u}}\|_\infty$ and $C = \|\tilde{\mathbf{c}}\|_\infty$. Tab. 7 compares the latest iterative algorithms for solving the maximum flow and minimum-cost circulation problems. While the most efficient algorithm achieves nearly linear running time relative to the number of edges m , the runtime can still be significant with long input sequences. To solve the minimum-cost circulation problem, we implemented the algorithm by (Lee & Sidford, 2014) using CVXPY (Agrawal et al., 2018; Diamond & Boyd, 2016).

Table 7: Overview of recent iterative algorithms for maximum flow and minimum-cost circulation problems.

Year	MCC Bound	Max-Flow Bound	Author
2014	$O(m\sqrt{n} \text{polylog}(n) \log^2(U))$	$O(m\sqrt{n} \text{polylog}(n) \log^2(U))$	Lee & Sidford (2014)
2022	$O\left(m^{\frac{3}{2} - \frac{1}{762}} \text{polylog}(n) \log(U + C)\right)$	$O\left(m^{\frac{10}{7}} \text{polylog}(n) U^{\frac{1}{7}}\right)$	Axiotis et al. (2022)
2023	$O\left(m^{\frac{3}{2} - \frac{1}{58}} \text{polylog}(n) \log^2(U)\right)$	$O\left(m^{\frac{3}{2} - \frac{1}{58}} \text{polylog}(n) \log^2(U)\right)$	van den Brand et al. (2023)
2023	$O\left(m^{1+o(1)} \log(U) \log(C)\right)$	$O\left(m^{1+o(1)} \log(U) \log(C)\right)$	Chen et al. (2023a)

Experiments were conducted on a computing device running Ubuntu 20.04.4 LTS, equipped with an Intel(R) Xeon(R) Platinum 8368 CPU at 2.40GHz, featuring 12 cores and 24 threads for parallel processing. Graphics processing was handled by an NVIDIA RTX 3090 Ti with 40GB of dedicated memory. The device also had 230GB of system memory, ensuring ample computational resources for efficient and effective experimentation.

Tab. 8 compares the runtime of benchmark methods for analyzing the sentence "although this dog is not cute, it is very smart." Methods like RawAtt and Rollout, which rely on raw attention weights, have the shortest runtime. In contrast, methods requiring complex post-processing to evaluate feature attributions have longer runtime. Our proposed methods' runtime is comparable to others in this latter group.

Table 8: Runtime of all methods for the showcase example.

Methods	Runtime (seconds)
RawAtt	0.123
Rollout	0.154
Grads	1.554
AttGrads	1.571
CAT	1.684
AttCAT	1.660
PartialLRP	1.571
TransAtt	1.620
LIME	1.462
KernelShap	2.342
IG	2.701
AF	2.301
GF	2.305
AGF	2.306