Evaluating Input Feature Explanations through a Unified Diagnostic Evaluation Framework

Anonymous ACL submission

Abstract

Explaining the decision-making process of machine learning models is crucial for ensuring their reliability and transparency for end users. One popular explanation form highlights key input features, such as i) tokens (e.g., Shapley Values and Integrated Gradients), ii) interactions between tokens (e.g., Bivariate Shapley and Attention-based methods), or iii) interactions between spans of the input (e.g., Louvain Span Interactions). However, these explanation types have only been studied in isolation, making it difficult to judge their respective applicability. To bridge this gap, we develop a unified framework that facilitates an automated and direct comparison between highlight and interactive explanations comprised of four diagnostic properties[1]. We conduct an extensive analysis across these three types of input feature explanations–each utilizing three different explanation techniques–across two datasets and two models, and reveal that each explanation has distinct strengths across the different diagnostic properties. Nevertheless, interactive span explanations outperform other types of input feature explanations across most diagnostic properties. Despite being relatively understudied, our analysis underscores the need for further research to improve methods generating these explanation types. Additionally, integrating them with other explanation types that perform better in certain characteristics could further enhance their overall effectiveness.

1 Introduction

Input feature explanations reveal how a model makes decisions based on a specific input. Among these, the most widely used explanation type is *Token Explanations* (`TokenEx`), which for Natural language Understanding tasks provide importance scores for the tokens of the input, using methods such as Shapley Values (Lundberg and Lee, 2017) and Integrated Gradients (Sundararajan et al., 2017). However, for complex reasoning tasks that require reasoning across multiple pieces of text, e.g., Fact Checking is performed given a claim and evidence, Natural Language Inference is performed given a premise and a hypothesis, `TokenEx` can be insufficient to capture the relations employed between the different parts of the input. To this end, *Token Interactive Explanations* (`TokenIntEx`) are proposed as another explanation type that provides importance scores for interactions between two tokens of the input with methods such as Bivariate Shapley (Masoomi et al., 2022) and Layer-wise Attention Attribution (Ye et al., 2021). Further, to enhance the semantic coherence of these interactive explanations, *Span Interactive Explanations* (`SpanIntEx`) is an explanation type that provides importance scores for interactions across tuples of spans, e.g., generated by Louvain community detection (Ray Choudhury et al., 2023). Figure 1 showcases these three types of input feature explanations.

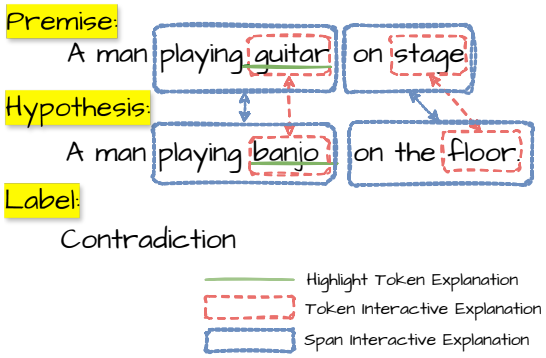


Figure 1: An example of the three types of Input Feature Explanations on an instance from the SNLI dataset, with their two most important pieces of explanation (token for `TokenEx`, token tuple for `TokenIntEx`, span tuple for `SpanIntEx`, correspondingly).

It is crucial to develop automated, rigorous and

[1]We will make the code available after acceptance.

comprehensive evaluation frameworks to ensure the principled selection of the most suitable explanation in a practical application (Yu et al., 2024) and systematic progress in the development of different types of explanations (Atanasova et al., 2022; Jolly et al., 2022). However, these three types of input feature explanations have typically been studied in isolation, where explanation methods of the same type are compared (Atanasova et al., 2020a; DeYoung et al., 2020; Janizek et al., 2021; Ray Choudhury et al., 2023). Moreover, evaluations of interactive explanations have been restricted to one property. To address these gaps, we develop *a unified framework that facilitates an automated and direct comparison between highlight and interactive input feature explanations on a suite of four diagnostic properties*. The framework allows for a rapid, scalable comparison across explainability methods, essential for the growing number of new techniques. Using the framework, we then perform an *extensive empirical analysis of the properties of existing explanation methods across all three explanation types*.

Unified Evaluation Framework. Our unified evaluation framework consists of four essential diagnostic properties – Faithfulness, Agreement with Human Annotation, Simulatability, and Complexity. They are the most widely used for `TokenEx` (Nauta et al., 2023) and include the only property used for interactive explanations – Faithfulness. *Faithfulness* (Chen et al., 2020, 2021; Ray Choudhury et al., 2023) measures the extent to which explanations accurately reflect the reasons used by the model in its predictions. *Agreement with Human Annotation* (Atanasova et al., 2020a) evaluates whether explanations exhibit an inductive bias akin to human reasoning, potentially enhancing their plausibility to end users. *Simulatablity* (Pruthi et al., 2022) estimates whether the explanations are useful to an agent for replicating the model's decisions. Finally, *Complexity* (Bhatt et al., 2021) evaluates whether the explanations are easy to comprehend. In the unified evaluation framework, we further extend the four properties to facilitate their application and comparison across all three explanation types (§2).

Extensive Empirical Analysis of Input Feature Explanations. We conduct an extensive analysis across two different textual tasks, two language models, and three explanation techniques for each input feature explanation type. Our findings indicate that `TokenEx` exhibit greater Comprehensiveness, and `SpanIntEx` – Sufficiency. Additionally, `SpanIntEx` and `TokenIntEx` align more closely with human annotations at the token level than `TokenEx`. Moreover, `SpanIntEx` demonstrate the highest interaction overlap with human annotations. Further, `SpanIntEx` are found to be most beneficial for Simulatability. Finally, our results suggest that `SpanIntEx` and `TokenEx` are easier to comprehend.

Overall, our analysis highlights the strengths of each explanation type across various diagnostic properties, with `SpanIntEx` generally outperforming others on most measures. However, we observe a trade-off between Comprehensiveness and plausibility, particularly with `SpanIntEx`, underscoring the need for methods that enhance both. Future research could explore integrated approaches that combine explanation types to optimize performance across all diagnostic properties.

2 A Unified Automated Evaluation Framework for Highlight Explanations

According to the established taxonomy of evaluation approaches for explainability methods, evaluations can be functionally grounded, human-grounded, or application-grounded (Doshi-Velez and Kim, 2017). Our work focuses on automated functionally-grounded evaluation, which serves as a foundation for the other evaluation types as it enables the rapid, scalable comparison across explainability methods, essential for the growing number of new techniques. To this end, we present a unified framework comprised of four widely employed diagnostic properties: Faithfulness, Agreement with Human Annotations, Simulatability, and Complexity. This section formally introduces them and outlines the extensions that allow for their application across different input feature explanation types.

2.1 Preliminaries

We start with a dataset D, and a model M fine-tuned on D. An instance $x \in D$ comprises two parts, e.g., a claim and an evidence, the first consisting of m tokens, and the second – of n tokens. We apply an explanation attribution method A_E of type $E \in \{\texttt{TokenEx}, \texttt{TokenIntEx}, \texttt{SpanIntEx}\}$ to M, and each $x \in D$: $A_E(M, x) = \{e_k^x, a_k^x | k \in [0, s-1]\}$, where e_k^x is a token/pair of tokens/pair of token spans from the input and a_k^x denotes its importance score. k is the importance ranking

of the corresponding piece of explanation. s is an upper limit for the number of most important pieces of explanation for an instance, such that: $s \in [1, m+n]$ for `TokenEx`, $s \in [1, m \cdot n]$ for `TokenIntEx`, and for `SpanIntEx` s varies for each instance with $s \in [1, f], f < m! \cdot n!$. Depending on the explanation type E, e_k^x can consist of: one token x_i for `TokenEx`, $i \in [0, m+n-1]$, one token pair (x_p, x_q) for `TokenIntEx`, where $p \in [0, m-1]$ and $q \in [m, m+n-1]$, one span pair $((x_s, \ldots, x_{s+l_1}), (x_t, \ldots, x_{t+l_2}))$ for `SpanIntEx`, where $s, s+l_1 \in [0, m-1]$ and $t, t+l_2 \in [m, m+n-1]$.

Considering a particular threshold for the number of most important explanation pieces s, we compute the total set of input tokens involved in the presented explanation for x:

$$T_{A_E,M}(x, s) = \{x_i | x_i \in e_k^x, k \in [0, s-1]\} \quad (1)$$

However, as noted above, the upper bound for s can vary across input feature explanations. Additionally, the number of tokens included in the top-k important explanations can differ substantially among explanation types – in top-1 we can have: 1 token for `TokenIntEx`, 2 tokens for `TokenIntEx`, and more than 2 tokens for `SpanIntEx`. This variability makes it difficult to compare results across different explanation types. To this end, we propose extensions for each of the studied diagnostic properties that result in *unified diagnostic properties* applicable for a direct comparison of the different types of input feature explanations.

2.2 Faithfulness

Faithfulness (DeYoung et al., 2020) assesses whether explanations accurately reflect the model's decision-making process. It involves two aspects – Sufficiency and Comprehensiveness – measured as the number of the model's prediction changes after keeping (SP) or omitting (CP) k most important portions of the input (see omission details in §A):

$$CP(x, A_E, k) = \left\{ \begin{array}{ll} 0, & \text{if } f(x - T_{A_E,M}(x,k)) = f(x) \\ 1, & \text{otherwise} \end{array} \right\} \quad (2)$$

$$SP(X, A_E, k) = \left\{ \begin{array}{ll} 1, & \text{if } f(T_{A_E,M}(x,k)) = f(x) \\ 0, & \text{otherwise} \end{array} \right\} \quad (3)$$

To take different thresholds k, we average over $k \in [0, s = m+n-1]$. We then also average the results across instances within D to compute the final Comprehensiveness and Sufficiency scores:

$$Comp_{A_E} = \frac{\sum_{x \in D}^{|D|} \sum_{k=0}^{s} CP(x, A_E, k)}{s * |D|} \quad (4)$$

$$Suff_{A_E} = \frac{\sum_{x \in D}^{|D|} \sum_{k=0}^{s} SP(x, A_E, k)}{s * |D|} \quad (5)$$

Unified Faithfulness. To facilitate the comparison of faithfulness scores, we introduce a *dynamic threshold* $\theta_{x,k}$. It represents the number of unique tokens used for a perturbation on x, same across explanation methods of all explanation types. Since top-k explanations for A_E of type `SpanIntEx` typically contain more tokens than for `TokenEx` or `TokenIntEx`, we set $\theta_{x,k}$ across the explanation methods of all types to:

$$\theta_{x,k} = |T_{A_{\texttt{SpanIntEx}},M}(x, k)| \quad (6)$$

Thus, $\theta_{x,k}$ becomes a dynamic threshold that adapts based on each instance's top-k explanation tokens from $A_{\texttt{SpanIntEx}}$. We then adjust the number of top-k explanations selected from $A_{\texttt{TokenEx}}$ and $A_{\texttt{TokenIntEx}}$, $k_{A_{\texttt{TokenEx}}}$ and $k_{A_{\texttt{TokenIntEx}}}$, correspondingly, to result in the same number of perturbed tokens $\theta_{x,k}$:

$$|T_{A_{\texttt{TokenEx}},M}(x, k_{A_{\texttt{TokenEx}}}(x))| = \theta_{x,k} \quad (7)$$

$$|T_{A_{\texttt{TokenIntEx}},M}(x, k_{A_{\texttt{TokenIntEx}}}(x))| = \theta_{x,k} \quad (8)$$

Furthermore, a *Random baseline* is established, where the tokens for perturbation are selected randomly to match the average $\theta_{x,k}$ across D.

2.3 Agreement with Human Annotations

Agreement with Human Annotations has been used to assess the overlap between generated and human-annotated explanations, which can indicate the plausibility of the generated explanations to end users. For $E =$ `TokenEx`, the measure is computed by calculating a precision score for the top-k most important explanations compared to the gold human annotations (Atanasova et al., 2020a).

For $E =$ `TokenEx`, $a_i^x, i \in [1, m+n]$ is the attribution score of ith most important explanation for x. $s = m+n$ is the number of explanations extracted from x. Corresponding to each explanation's attribution score, s thresholds are set, forming the threshold list $\omega_{A_E}(x) = [a_0^x, ..., a_s^x]$. By selecting explanations with higher attribution scores than each threshold in $\omega_{A_E}(x)$, s targeted explanation sets are obtained, where $C_{A_E}(x, i)\{e_j^x : a_j^x <= a_i^x\}$ represents the set for the ith threshold, a_j^x is the attribution score of token e_j^x for $E =$ `TokenEx`. Comparing these sets with the golden explanation set e^G, s precision-recall pairs

$P_i/R_i(x, e^G, A_E)$ can be derived. Average Precision (AP) is then obtained by weighting P_i with the corresponding R_i increase:

$$P_i/R_i(x, e^G, A_E) = Pre/Rec(C_{A_E}(x, i), e^G) \quad (9)$$

$$AP_{A_E}(x, e^G) = \sum_{i=0}^{s}(R_i - R_{i-1}) * P_i \quad (10)$$

Mean AP (MAP) is calculated for all $x \in D$:

$$MAP_{A_E} = \frac{\sum_{x \in D}^{|D|} AP_{A_E}(x, e^G)}{|D|} \quad (11)$$

Unified Agreement with Human Annotation Measure. For a fair comparison between the different types of explanations, the thresholds $\omega_{A_E}(x)$ for including the same number of tokens across the explanation methods follows the procedure set for the Unified Faithfulness (§2.2). Furthermore, we measure Agreement with Human Annotations at the **interaction level** for the gold `SpanIntEx`/`TokenIntEx` explanations and at the **token level** for gold `TokenEx` explanations.

Interaction-level Agreement with Human Annotations. For a fair comparison between `TokenIntEx` and `SpanIntEx` methods, we adapt $MAP_{\texttt{TokenEx}}$ to the interaction level. Specifically, we compute the mean average precision (Eq. 11) w.r.t. the human-annotated `TokenIntEx`/`SpanIntEx` sets.

Token-level Agreement with Human Annotations. For a fair comparison between `TokenEx` and `TokenIntEx`/`SpanIntEx` methods, we extract tokens from `TokenIntEx`/`SpanIntEx` and compare them with tokens from golden `TokenIntEx`/`SpanIntEx` sets. To compute MAP at token level, we follow the similar procedure set for E = `TokenEx` (Eq. 11) with threshold lists $\omega_{A_{\texttt{TokenIntEx/SpanIntEx}}}(x)$, but the targeted sets $C_{A_{E_{token}}}(x, i)$ contain tokens extracted from `TokenIntEx`/`SpanIntEx` methods. The golden set $S_{A_{E_{token}}}(x)$ aggregates tokens from golden `TokenIntEx`/`SpanIntEx` sets.

We also set a *Random baseline*, where the number of randomly selected span pairs, token pairs, or tokens for each instance matches the average number of tokens per instance extracted with a `SpanIntEx` explanation method.

2.4 Simulatability

Simulatability was initially proposed to measure how accurately humans can predict a model's outputs based on its explanations (Chen et al., 2024; Hase et al., 2020). Previous studies demonstrated that Simulatability can be approximated using an automated agent model as a surrogate for human understanding (Pruthi et al., 2022). *Given the established positive correlation between Simulatability and human evaluation of explanation utility*, we integrate the Simulatability scores obtained from an agent model with different explanation types to approximate their utility for humans.

Following existing work (Hase et al., 2020), we train an agent model AM, sharing the same architecture as the original model M, to simulate M's predictions Y' using produced explanations. During AM's training phase, we extract the top-k explanations and incorporate them in the input. In comparison, another agent model, AM_O, is trained without explanation guidance as a baseline on the same training set. During the testing phase, the simulation accuracy of AM and AM_O over the shared dataset D is calculated.[2] The difference between the accuracies is interpreted as the explanation's effect in enhancing the simulatability of M:

$$Sim = ACC(AM(D), Y') - ACC(AM_O(D), Y') \quad (12)$$

Unified Simulatability. To compare the simulation utility of different explanation types, we train a separate agent model AM_E for each explanation method A_E and calculate the corresponding simulation performance on the common test set. For a fair comparison across the different explanation method types A_E, we first ensure top-k_E explanations are presented for assisting the agent's training for A_E, following Section 2.2. This ensures each model is exposed to the same quantity of unique tokens from different explanation types.

During the training of AM, we introduce the explanations from A_E into the learning of AM_{A_E}; we supplement x with top-k_{A_E} explanations instead so that the agent model is trained with the same mechanism whether the explanations are provided or not, and each training instance will contain the input sequence x_{A_E} and golden label Y' which is predicted by the original model M. Specifically, we examine two different ways of presenting the explanations as part of the original input sequence, I_{Symbol} and I_{Text} (see §B.); all aim to ensure the explanations of different types are inserted similarly.

At test time, the F1 scores of agent models AM_E

[2] While existing work (Hase et al., 2020) notes that incorporating natural language explanations in the testing phase could leak the predicted label, we use only input feature explanations that do not contain additional information.

and AM_O over D are calculated:

$$SF_E = F1(AM_E(x_E), Y'), x \in D \quad (13)$$

$$SF_O = F1(AM_O(x), Y'), x \in D \quad (14)$$

Note that the explanations from A_E are also provided to the input for the unseen instances for agent model AM_E. The final simulation metric then indicates how much this specific type of input feature explanation enhances the model's simulatability:

$$RSF_E = SF_E - SF_O \quad (15)$$

2.5 Complexity

Feature attribution explanations are designed to aid human understanding of a model's reasoning over specific instances. However, since humans have a limited capacity to process large amounts of information simultaneously, these explanations need to be easy to comprehend. Even if we select only the top-k attributions with the highest importance scores, they need to be distinctive as opposed to the attribution scores having a uniform distribution. Bhatt et al. (2021) propose to measure the complexity of a produced explanation with entropy (Rényi, 1961) over the attribution scores of all the produced explanations by $A_{\texttt{TokenEx}}$ method:

$$P(x,p) = |a_p^x| \,/\, \textstyle\sum_{q=1}^{m+n} |a_q^x| \quad (16)$$

$$CL(x) = -\textstyle\sum_{p=1}^{m+n} P(x,p) \ln(P(x,p)) \quad (17)$$

$m{+}n$ is the total number of generated $A_{\texttt{TokenEx}}$ explanations, and all explanations are considered for the complexity score computation. Higher entropy means different features have similar attribution scores, where the simplest explanation, with low entropy, would be concentrated on one feature.

Unified Complexity. To ensure a fair comparison across different types of explanation methods A_E, we maintain consistency in the size of the chosen explanation lists across all A_E for the same instance, denoted as k_x, as the number of generated $A_{\texttt{TokenEx}}$/$A_{\texttt{TokenIntEx}}$/$A_{\texttt{SpanIntEx}}$ explanations originally vary for the same x. The complexity score $CL_{A_E}(x, k_x)$ of the top-k_x explanation list under method A_E is calculated as:

$$P_{A_E}(x, k_x, i) = |a_i^x| \,/\, \textstyle\sum_{j=1}^{k_x} |a_j^x| \quad (18)$$

$$CL_{A_E}(x, k_x) = -\textstyle\sum_{i=1}^{k_x} P_{A_E}(x, k_x, i) \ln(P_{A_E}(x, k_x, i)) \quad (19)$$

where a_i^x/a_j^x represent the i/jth highest attribution score from the explanation set for x.

The final complexity score is an average of $CL_{A_E}(x, k_x)$ across all $x \in D$:

$$CL_E = \textstyle\sum_{x \in D}^{|D|} CL_E(x, k_x)/|D| \quad (20)$$

Notably, k_x is calculated from the number of explanations produced by method $A_{\texttt{SpanIntEx}}$ for x, which varies based on the span interaction extraction method and is known only after generation.

3 Experimental Setup

3.1 Datasets

We select the natural language inference dataset SNLI (Bowman et al., 2015), where instances consist of a premise, hypothesis, and a label $y \in \{entailment, neutral, contradiction\}$. Additionally, we select the fact checking dataset FEVER (Thorne et al., 2018; Atanasova et al., 2020b), where instances consist of a claim, evidence, and a label $y \in \{entailment, neutral, contradiction\}$.[3] We generate input feature explanations by sampling 4,000 instances from each train, dev, and test set, due to the high computational cost, particularly for Shapley-based explanations (Atanasova et al., 2020a). For Agreement with Human Annotations property, we use existing human explanation annotations for SNLI and FEVER (see §C).

3.2 Input Feature Explanation Methods

To generate importance scores, we first select three common `TokenEx` techniques – Shapley (Lundberg and Lee, 2017), Attention (DeYoung et al., 2020), and Integrated Gradients (IG, Sundararajan et al. (2017)). For `TokenIntEx`, we employ Bivariate Shapley (Masoomi et al., 2022), Attention (Clark et al., 2019), and Layer-wise Attention Attribution (Ye et al., 2021). Notably, the `TokenIntEx` techniques are the bivariate version of the techniques used for generating `TokenEx`; e.g. Layer-wise Attention Attribution uses IG.

Following Ray Choudhury et al. (2023), we apply the Louvain algorithm (Blondel et al., 2008) for each of the three selected `TokenIntEx` to generate importance scores for `SpanIntEx` methods, where the importance score of each span interaction is averaged over the importance scores of the token interactions within it. We will refer to Shapley, Attention, and IG as the explanation base types

[3] https://huggingface.co/datasets/copenlu/fever_gold_evidence

used for generating all types of input feature explanations for brevity. See §E for more details.

3.3 Models

We use the BERT-base-uncased model (Devlin et al., 2019) with 12 encoder layers, and the BART-base-uncased model (Lewis et al., 2020) with 6 encoder and 6 decoder layers, as common representatives of the encoder and the encoder-decoder Transformer architecture (Vaswani et al., 2017). Our choice is particularly influenced by the substantial computational requirements of the input feature explanations, especially pronounced for Shapley (Atanasova et al., 2020a). Additionally, our choice is guided by the need to directly access the models' internals for generating IG and Attention-based explanations. Furthermore, while our framework currently utilizes the said models, it is designed to be easily adaptable to other models or newly developed explainability techniques, provided that there are more robust computational resources available.

We fine-tune the base models on SNLI and FEVER and use them to generate explanations. Their performance on the test sets is shown in §D. For the Simulation property, we follow existing work (Fernandes et al., 2022; Pruthi et al., 2022) and train simulator agent models (§2.4) with the same architectures as the base ones. Following Fernandes et al. (2022), we split the test set into train/dev/test for the training of the agent model.

4 Results and Discussion

We now present the results of our unified evaluation framework (§2) illustrated in Fig. 2. They include explanation methods of types `SpanIntEx`, `TokenIntEx`, and `TokenEx` (§3.2), two models (§3.3), two datasets (§3.1), and three base explanation techniques per explanation type (§3.2). For Simulatability, we select the results of I_{sym}, as this form avoids repeating the input text and increasing the input size substantially. For Agreement with Human Annotations, we select the Token-level results as they are present for all explanation types. §F lists detailed results per property.

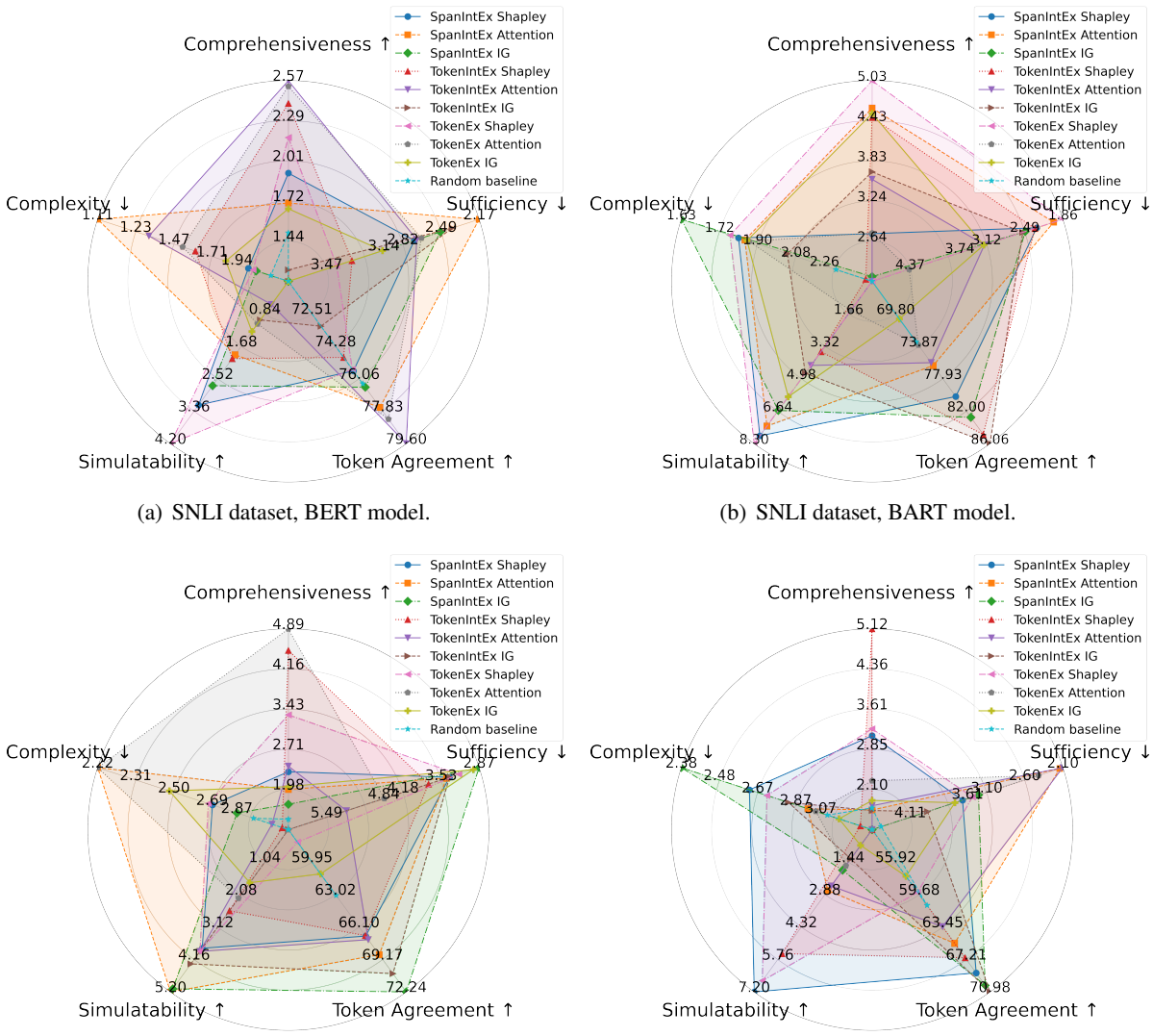
4.1 Faithfulness

Unified Comprehensiveness. Across both datasets and models, **`TokenEx` and `TokenIntEx` are identified as the most comprehensive explanation types**, achieving the highest scores in 7/12 and 5/12 cases, respectively. `SpanIntEx`, designed to enhance the semantic coherence of interactive explanations by including additional context, often incorporates tokens that do not directly contribute to the model's prediction, thus explaining its lower comprehensiveness scores. Compared to the random baseline, `TokenEx` and `SpanIntEx` always outperform it, while `TokenIntEx` mostly underperform it when based on IG. Across the base explanation techniques, `TokenEx` performs best when based on Attention for BERT and on Shapley for BART, indicating that **different base explanation techniques can perform better for different architectures**. Both `TokenIntEx` and `SpanIntEx` show optimal performance when based on Shapley and Attention. Overall, the results indicate a *stronger performance of Attention and Shapley over IG* across all explanation types.

Unified Sufficiency. *`SpanIntEx` ranks as the most sufficient explanation type in 7/12 cases*, surpassing `TokenEx`, which performs well in only 3/12 cases. While contrary to `SpanIntEx` Comprehensiveness performance, we attribute this to the semantic coherence of the extracted top spans, which provide more meaningful information. Note that while Sufficiency is highly desirable, Comprehensiveness is not required in all downstream applications as end-users prefer simpler, more general explanations with fewer causes (Thagard, 1989). Unlike `TokenEx`, which consistently outperforms the random baseline, `TokenIntEx` and `SpanIntEx` struggle to outperform it on FEVER, likely due to the longer input, posing challenges for the explanations to accurately unveil the model's internal processes. The results from different base explanation techniques show no clear trends, indicating a **significant variability stemming from the specific dataset and model architecture**.

4.2 Agreement with Human Annotation

`SpanIntEx` and `TokenIntEx` show higher agreement scores with human annotations than `TokenEx`. Similarly, `SpanIntEx` consistently achieve higher agreement with human interaction-level annotators, especially when based on Attention scores (see §F). This indicates that **`SpanIntEx` are more plausible to humans due to their enhanced semantic coherence**. In contrast, `TokenEx` often score lower than the random baseline. Moreover, considering `SpanIntEx`'s lower performance in Comprehensiveness, there emerges **a distinct trade-off between Compre-**



(a) SNLI dataset, BERT model.

(b) SNLI dataset, BART model.

(c) FEVER dataset, BERT model.

(d) FEVER dataset, BART model.

Figure 2: Unified evaluation framework (§2) results for all feature attribution methods (§3.2).

hensiveness and Agreement with Human Annotations. Across base explanation techniques, IG performs best for FEVER; IG and Attention – for SNLI. In addition, we find that for the interaction-level agreement, `TokenIntEx` and `SpanIntEx` perform worst when based on Shapley. The lower agreement results for Shapley compared to its better results on Comprehensiveness again indicate an existing trade-off between the two properties.

4.3 Simulatability

Our results show that `SpanIntEx` achieve the highest simulatability in 9/12 cases, helping the agent model accurately reproduce the original model's prediction. This again underscores the **critical role of contextual information and enhanced semantic coherence provided by `SpanIntEx`**. Notably, providing `SpanIntEx` to agents improves their ability to simulate the original model by up to 7.9 F1 points compared to without explanations. Among base explanation techniques, IG consistently performs best for `SpanIntEx`; other techniques do not exhibit a clear trend.

4.4 Complexity

`SpanIntEx` and `TokenEx` generally achieve similar complexity, which consistently remains lower than those of `TokenIntEx`. This suggests that **`TokenEx` and `SpanIntEx` generate more distinctive attribution scores, potentially making them easier for humans to understand**. Regarding the base explanation techniques, Attention consistently yields the best complexity scores for BERT across all explanation types. There is no

clear trend for BART. Additionally, TokenIntEx frequently underperform the random baseline, highlighting its complexity (see §F).

4.5 Overall

In summary, we find that while TokenEx and TokenIntEx generally provide more comprehensive insights, SpanIntEx performs better in Sufficiency due to its enhanced semantic coherence (§4.1). This calls for *better methods for generating SpanIntEx that are both comprehensive and sufficient*. Additionally, there is a trade-off between Comprehensiveness and Agreement with Human Annotations (§4.2), suggesting that the most faithful explanations might be less plausible to end users. This highlights the *need for advanced methods to boost both the Comprehensiveness and plausibility* of SpanIntEx possibly leveraging the advantage of TokenEx. Furthermore, SpanIntEx significantly improves simulatability by allowing agents to accurately replicate model decisions (§4.3), which is crucial in practice. Finally, the complexity analysis (§4.4) shows that SpanIntEx and TokenEx are potentially easier to comprehend than TokenIntEx when considering the importance score distribution.

Overall, our results highlight the differences between the different types of input feature explanations, with SpanIntEx outperforming others on most measures. As no one type performs best on all properties, we *call for the development of combined methods that can leverage the strength of the different explanation types and potentially lead to an overall improvement of the explanation utility*.

5 Related Work

Input Feature Explanations. Considerable research exists on extracting explanations for input data. Methods like perturbation-based attribution (e.g., Shapley (Lundberg and Lee, 2017)), attention-based methods (e.g., Attention (Jain and Wallace, 2019; Serrano and Smith, 2019)), and gradient-based methods (e.g., Integrated Gradients (Sundararajan et al., 2017; Serrano and Smith, 2019)) are prevalent for highlighting individual tokens (Atanasova et al., 2020a). As individual tokens might be insufficient to explain the model, many attribution methods have been extended to bivariate forms (Masoomi et al., 2022; Janizek et al., 2021; Sundararajan et al., 2017; Ye et al., 2021) to capture input token interactions. More recent work has explored how interactions between groups of tokens collectively contribute to model reasoning (Ray Choudhury et al., 2023; Chen et al., 2021). Unlike other work where token groups might consist of tokens from arbitrary positions, Ray Choudhury et al. (2023) explicitly capture span interactions, enhancing the comprehensiveness of explanations by containing the entire spans.

Automated Explanation Evaluation. For evaluating TokenEx, DeYoung et al. (2020); Atanasova et al. (2020a) propose metrics to measure how faithful explanations are to the model's inner reasoning. They also assess the plausibility of explanations to humans by measuring the agreement of TokenIntEx with human annotations. To assess the utility of explanations to humans, Pruthi et al. (2022) propose to use an agent model as a proxy for humans and evaluate whether explanations aid in model simulatability. Complexity Bhatt et al. (2021) measures the distribution of attribution scores of TokenEx and assesses whether the key tokens in token explanations are easily comprehensible to humans. To evaluate TokenIntEx most works adopt the faithfulness or axiomatic/theoretical path (Tsang et al., 2020; Sundararajan et al., 2020; Janizek et al., 2021). Current work on evaluating SpanIntEx has primarily focused on faithfulness (Ray Choudhury et al., 2023). However, since SpanIntEx, TokenIntEx, and TokenEx contain varying amounts of tokens, which, e.g., affects the faithfulness test, this makes direct comparisons between different explanation types using existing metrics challenging. To our knowledge, no prior paper has involved all types of input feature explanations within a unified evaluation framework.

6 Conclusion

We introduced a unified evaluation framework for input feature attribution analysis to guide the principled selection of the most suitable explainability technique in practical applications. Our analysis outlines the diverse strengths and trade-offs among TokenEx, TokenIntEx, and SpanIntEx. Our findings particularly underscore SpanIntEx's superior performance in Sufficiency, agreement with human inductive biases, its enhancement of Simulatability, and Complexity, compared to TokenEx and TokenIntEx. Future efforts should focus on developing combined methods that enhance all explanation properties.

Limitations

Our work introduces a unified framework to evaluate input feature explanations across four key properties. We generated three types of explanations using three attribution methods on two transformer models (BERT-base and BART-base) for two NLU tasks (NLI and fact-checking). This enabled us to assess and compare the properties of each explanation type. Due to computational resource limitations, we did not include larger decoder-only models in our evaluation. Future research could *explore other models* to provide additional insights.

We note that our work considered the FEVER and SNLI datasets as they are the only available datasets with annotations of human interactive explanations, required for the Agreement with Human Annotations property. In future work, given the availability of other datasets, examining the properties of different explanations in various *tasks beyond NLI and fact checking* would be valuable, especially for simpler tasks that consist of only one input part or more complex tasks that consist of more than two parts with possible relationships between them. Additionally, tasks with longer textual inputs, which are known to pose greater challenges for current explainability techniques (Atanasova et al., 2020a), could also be analyzed.

Furthermore, while we consider four widely used explanation properties in this automatic evaluation framework, future works should consider verifying, potentially with supplementary human studies, that the properties are well aligned with the downstream utility of the explanations in different application tasks (Miller, 2019). We note that manual evaluation, while valuable, is time-consuming and costly. Automated evaluation, with our proposed framework, allows for quicker insights, helping prioritize methods that may benefit from human-centered evaluation. Additionally, the properties we evaluate demonstrate why human evaluation is not necessarily required at this stage. Faithfulness measures whether explanations reflect the model's internal reasoning, a task humans cannot assess (see Faithfulness evaluation guidelines in (Jacovi and Goldberg, 2020)). Explanations that fail this test should not be considered for further human evaluation as they can be harmful, e.g. by hiding the model's flaws and biases. Agreement with Human Annotation already captures alignment with human reasoning, ensuring explanations are plausible. Automated Simulatability correlates strongly with human studies, providing a reliable proxy for replicating the model's behaviour without the need for expensive human experiments (Pruthi et al., 2022). Finally, automated methods ensure consistency and objectivity, while human annotations can introduce subjectivity and variability. Studies have even shown conflicting results from human evaluations (Poursabzi-Sangdeh et al., 2021; Ribeiro et al., 2016). Automated evaluation provides an objective, reproducible baseline, which can later be supplemented by human evaluations where needed.

We have also employed three base representative explainability methods for each of the three types of input feature explanations. However, *more existing and newly emerged base explainability methods* could be explored in future work. Apart from that, future work could adapt and extend our framework to other forms of explanations such as free-text explanations produced by self-rationalization models (Liu et al., 2022, 2023b,a, 2024b,a). All said potential future studies are well facilitated by the efficient automated evaluation proposed with our framework.

Our findings indicate that span interactive explanations (`SpanIntEx`) have a notable advantage over other types in Agreement with Human Annotation, Simulatability, and Complexity, suggesting they are easier for humans to understand. This insight could inspire future work to leverage `SpanIntEx` as the input feature explanation in HCI models. However, `SpanIntEx` shows low comprehensiveness in faithfulness evaluations. The Louvain algorithm, used for `SpanIntEx` generation, may limit its comprehensiveness despite using different attribution methods for `TokenIntEx`. Future work should explore better methods for capturing span interactions and possibly combine `SpanIntEx` and `TokenEx` for higher faithfulness, as `TokenEx` demonstrates a stable advantage in comprehensiveness.

Another core finding is that the attribution method significantly affects most diagnostic properties of all explanation types, such as sufficiency. No single attribution method consistently excels across all properties, highlighting the need for continuous evaluation and improvement in attribution methods, particularly for `SpanIntEx`.

To ensure a fair comparison, our unified evaluation framework currently considers only the token count differences among various input feature ex-

planations, with interactive explanations flattened. Future work could involve a human-in-the-loop approach to account for the effects of interactive explanations beyond just token count differences. For example, a display system could visually present highlighted tokens and interactions to gauge human preferences. Our work provides a starting point for comparing input feature explanations from an automated evaluation perspective, and future research could explore additional factors such as psychological elements and visual aspects from a human perspective.

References

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020a. A diagnostic study of explainability techniques for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3256–3274, Online. Association for Computational Linguistics.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2022. Diagnostics-Guided Explanation Generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10445–10453.

Pepa Atanasova, Dustin Wright, and Isabelle Augenstein. 2020b. Generating label cohesive and well-formed adversarial claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3168–3177, Online. Association for Computational Linguistics.

Umang Bhatt, Adrian Weller, and José M. F. Moura. 2021. Evaluating and Aggregating Feature-based Model Explanations. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. E-SNLI: Natural Language Inference with Natural Language Explanations. *Advances in Neural Information Processing Systems*, 31.

Hanjie Chen, Song Feng, Jatin Ganhotra, Hui Wan, Chulaka Gunasekara, Sachindra Joshi, and Yangfeng Ji. 2021. Explaining neural network predictions on sentence pairs via learning word-group masks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3917–3930, Online. Association for Computational Linguistics.

Hanjie Chen, Guangtao Zheng, and Yangfeng Ji. 2020. Generating hierarchical explanations on text classification via feature interaction detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5578–5593, Online. Association for Computational Linguistics.

Yanda Chen, Ruiqi Zhong, Narutatsu Ri, Chen Zhao, He He, Jacob Steinhardt, Zhou Yu, and Kathleen McKeown. 2024. Do models explain themselves? counterfactual simulatability of natural language explanations. In *Forty-first International Conference on Machine Learning*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Patrick Fernandes, Marcos Treviso, Danish Pruthi, André Martins, and Graham Neubig. 2022. Learning to Scaffold: Optimizing Model Explanations for Teaching. *Advances in Neural Information Processing Systems*, 35:36108–36122.

Peter Hase, Shiyue Zhang, Harry Xie, and Mohit Bansal. 2020. Leakage-adjusted simulatability: Can models generate non-trivial explanations of their behavior in natural language? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages

4351–4367, Online. Association for Computational Linguistics.

Alon Jacovi and Yoav Goldberg. 2020. Towards Faithfully Interpretable NLP Systems: How Should We Define and Evaluate Faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Joseph D Janizek, Pascal Sturmfels, and Su-In Lee. 2021. Explaining Explanations: Axiomatic Feature Interactions for Deep Networks. *Journal of Machine Learning Research*, 22(104):1–54.

Shailza Jolly, Pepa Atanasova, and Isabelle Augenstein. 2022. Generating Fluent Fact Checking Explanations with Unsupervised Post-editing. *Information*, 13(10):500.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Wei Liu, Haozhao Wang, Jun Wang, Zhiying Deng, Yuankai Zhang, Cheng Wang, and Ruixuan Li. 2024a. Enhancing the Rationale-Input Alignment for Self-explaining Rationalization. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 2218–2230. IEEE.

Wei Liu, Haozhao Wang, Jun Wang, Ruixuan Li, Xinyang Li, YuanKai Zhang, and Yang Qiu. 2023a. MGR: Multi-generator Based Rationalization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12771–12787.

Wei Liu, Haozhao Wang, Jun Wang, Ruixuan Li, Chao Yue, and YuanKai Zhang. 2022. FR: Folded Rationalization with a Unified Encoder. *Advances in Neural Information Processing Systems*, 35:6954–6966.

Wei Liu, Jun Wang, Haozhao Wang, Ruixuan Li, Zhiying Deng, Yuankai Zhang, and Yang Qiu. 2024b. D-Separation for Causal Self-Explanation. *Advances in Neural Information Processing Systems*, 36.

Wei Liu, Jun Wang, Haozhao Wang, Ruixuan Li, Yang Qiu, Yuankai Zhang, Jie Han, and Yixiong Zou. 2023b. Decoupled Rationalization with Asymmetric Learning Rates: A Flexible Lipschitz Restraint. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1535–1547.

Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. *Advances in neural information processing systems*, 30.

Aria Masoomi, Davin Hill, Zhonghui Xu, Craig P Hersh, Edwin K. Silverman, Peter J. Castaldi, Stratis Ioannidis, and Jennifer Dy. 2022. Explanations of black-box models based on directional feature interactions. In *International Conference on Learning Representations*.

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

Meike Nauta, Jan Trienes, Shreyasi Pathak, Elisa Nguyen, Michelle Peters, Yasmin Schmitt, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. 2023. From Anecdotal Evidence to Quantitative Evaluation Methods: A Systematic Review on Evaluating Explainable AI. *ACM Computing Surveys*, 55(13s):1–42.

Forough Poursabzi-Sangdeh, Daniel G Goldstein, Jake M Hofman, Jennifer Wortman Wortman Vaughan, and Hanna Wallach. 2021. Manipulating and Measuring Model Interpretability. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–52.

Danish Pruthi, Rachit Bansal, Bhuwan Dhingra, Livio Baldini Soares, Michael Collins, Zachary C. Lipton, Graham Neubig, and William W. Cohen. 2022. Evaluating explanations: How much do explanations from the teacher aid students? *Transactions of the Association for Computational Linguistics*, 10:359–375.

Sagnik Ray Choudhury, Pepa Atanasova, and Isabelle Augenstein. 2023. Explaining interactions between text spans. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12709–12730, Singapore. Association for Computational Linguistics.

Alfréd Rényi. 1961. On Measures of Entropy and Information. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability, volume 1: contributions to the theory of statistics*, volume 4, pages 547–562. University of California Press.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?" Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Mukund Sundararajan, Kedar Dhamdhere, and Ashish Agarwal. 2020. The Shapley Taylor Interaction Index. In *International conference on machine learning*, pages 9259–9268. PMLR.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Paul Thagard. 1989. Explanatory Coherence. *Behavioral and brain sciences*, 12(3):435–467.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Michael Tsang, Sirisha Rambhatla, and Yan Liu. 2020. How Does This Interaction Affect Me? Interpretable Attribution for Feature Interactions. *Advances in neural information processing systems*, 33:6147–6159.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xi Ye, Rohan Nair, and Greg Durrett. 2021. Connecting attributions and QA model behavior on realistic counterfactuals. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5496–5512, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haeun Yu, Pepa Atanasova, and Isabelle Augenstein. 2024. Revealing the parametric knowledge of language models: A unified framework for attribution methods. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8173–8186, Bangkok, Thailand. Association for Computational Linguistics.

A Unified Faithfulness Evaluation: Explanation Masking Details

As discussed in Section §2.2, we introduce the dynamic threshold $\theta_{x,k}$ to ensure an identical number of tokens from different types of top input feature explanations for the same instance in Unified Faithfulness evaluation.

For the Unified Comprehensiveness evaluation, we conduct a similar process for all three explanation types separately. Using `TokenEx` as an example, we first calculate the maximum number of top `TokenEx` for disturbing each instance x as $k_{\texttt{TokenEx}}(x)$ following Eq. 7. To omit the `TokenEx` tokens from the original input, we replace them with [MASK] tokens, while keeping the rest unchanged. The specific [MASK] token used depends on the model architecture. We then gradually increase the number of `TokenEx` tokens masked out until it reaches $k_{\texttt{TokenEx}}(x)$ and record the corresponding changes in model predictions. The average prediction change across the dynamic threshold and all instances gives the Unified Comprehensiveness score for `TokenEx`. For `TokenIntEx` and `SpanIntEx`, the only difference is that we mask out token pairs for `TokenIntEx` and span pairs for `SpanIntEx`, with the maximum explanations masked out as $k_{\texttt{TokenIntEx}}(x)$ and $k_{\texttt{SpanIntEx}}(x)$ for each instance x, calculated using Eq. 6 and Eq. 8 respectively.

For the Unified Sufficiency evaluation, we conduct experiments for the three explanation types separately. Unlike Unified Comprehensiveness, we retain only the tokens/token pairs/span pairs for the input while masking out all other tokens with [MASK] tokens for each instance, depending on the model architecture used. We first calculate the maximum number of top explanations involved in disturbance for each explanation type for instance x using Eq. 6, Eq. 7, and Eq. 8. Then, we keep the token/token pairs/span pairs in the model input by masking out all other tokens, starting with one explanation and adding one more explanation for each subsequent disturbance until the total number of explanations reaches $k_{\texttt{TokenEx}}(x)$, $k_{\texttt{TokenIntEx}}(x)$, or $k_{\texttt{SpanIntEx}}(x)$. Meanwhile, we record the model predictions for each disturbance. The Unified Sufficiency score for each explanation type is then calculated by averaging the prediction changes across the dynamic threshold for that explanation type, considering all instances.

B Detailed Explanation Insertion Method

To enable a fair comparison among different input feature explanations in terms of simulatability (§2.4), we applied consistent insertion formats to combine the explanations with the original input for training the agent models. This design aims to minimize noise from insertion format differences. We tested two ways, each applicable to all types of input feature explanations, to construct input sequences with inserted explanations of type E. These input sequences are denoted x_E in §2.4, omitting specific insertion format details for brevity.

For Symbol-Insertion I_{Symbol}, we preserve the original input sequence but insert special symbols $<$ and $>$ to quote the tokens (for `TokenEx` and `TokenIntEx`) or spans (for `SpanIntEx`) within the input. Additionally, for `TokenEx`, we append a ranking mark after each quoted token based on their attribution scores, ranked in descending order. For `TokenEx` and `SpanIntEx`, each quoted token/span is also assigned a ranking mark indicating the rank of their respective interactions by attribution score, ensuring tokens/spans from the same interaction share the same mark. This method allows us to generate input sequences combined with different input feature explanations in a consistent symbol insertion format.

For Text-Insertion I_{Text}, we append tokens, token tuples, or span tuples to the end of the original input sequence for each explanation type. They are added in the order ranked by descending attribution score. Specifically, for `TokenEx`, tokens from different `TokenEx`explanations are separated by semicolons. For `TokenIntEx` and `SpanIntEx`, tokens/spans within each interaction are connected by a comma, and different interactions are separated by semicolons. This approach constructs input sequences combined with each type of input feature explanation in a consistent text insertion format.

C Agreement Dataset Details

To assess how different types of input feature explanations overlap with human annotations, we collected golden explanations of various types from e-SNLI and SpanEx for instances within SNLI and FEVER, respectively. Detailed information about the annotated explanation types and the number of instances with labeled explanations for these datasets is shown in Table 1. For the SNLI dataset,

D	E	**Size**
SNLI	-	549367 Train 9842 Dev 9824 Test
e-SNLI	`TokenEx`	549367 Train 9842 Dev 9824 Test
SpanEx-SNLI	`SpanIntEx` `TokenIntEx`	3865 Test
FEVER	-	145449 Train 9999 Dev 9999 Test
SpanEx-FEVER	`SpanIntEx` `TokenIntEx`	3206 Test

Table 1: Overview of datasets SNLI (Bowman et al., 2015), FEVER (Thorne et al., 2018), *SpanEx* (Ray Choudhury et al., 2023) and e-SNLI (Camburu et al., 2018). *SpanEx* contains instances from SNLI and FEVER, annotated with `SpanIntEx` explanations including token-level explanations (`TokenIntEx` explanations). e-SNLI contains instances from SNLI dataset, annotated with `TokenEx` explanations.

e-SNLI provides `TokenEx` explanations, while *SpanEx-SNLI* includes `SpanIntEx` explanations and token-level interactions (`TokenIntEx` explanations). We selected 3,865 overlapping instances and evaluated the human agreement score for different types of input feature explanations. For FEVER, *SpanEx-FEVER* includes `SpanIntEx` and token-level interactions (`TokenIntEx` explanations). Since no `TokenEx` explanations are provided, we extracted tokens from the golden `TokenIntEx` explanations in *SpanEx-FEVER* as an approximation. These selected instances are also used when evaluating other properties of input feature explanations.

D Base Model Performance.

As shown in Table 2, we report the performance of fine-tuned BERT-base and BART-base models on SNLI and FEVER, respectively. These models, fine-tuned for their specific tasks, are used to generate various input feature explanations through different explainability techniques. Importantly, these are the original models that the agent models, as described in §2.4, learn to simulate.

E Explainability Techniques

In this section, we detail the explainability techniques employed to generate various types of input feature explanations. As outlined in §3.2, we categorize these techniques based on the method used

$Model$	**F1 score** Dev	Test
BERT-SNLI	87.21	88.43
BART-SNLI	86.81	85.40
BERT-FEVER	86.21	89.49
BART-FEVER	85.19	84.88

Table 2: The performance of our BERT-base and BART-base models fine-tuned on SNLI and FEVER, respectively, regarding F1 score(%).

for generating `TokenEx`, while `TokenIntEx` explanations stem from their bivariate variants, forming the basis for `SpanIntEx` explanations.

As denoted in Section §2.1, x_i represents the ith token with instance x. To better illustrate the explainability techniques below, we use F as the set of all tokens within this instance and S as the subset of F. All explanations are obtained using model M, which is omitted in the following notions for brevity. We use $A_{\texttt{TokenEx}}(x_i)$ to denote the attribution score generated by explainability technique A for the ith token x_i, $A_{\texttt{TokenIntEx}}(x_i, x_j)$ as the attribution score for token interaction (x_i, x_j), $A_{\texttt{TokenIntEx}}(x_i \mid x_j)$ as the importance score of token x_i conditioned on x_j is present when the directed importance between tokens within (x_i, x_j) is considered in some attribution techniques, $A_{\texttt{SpanIntEx}}(span_i^0, span_i^1)$ as the attribution score for corresponding span interaction, where $span_i^0 = (x_s, ..., x_{s+l_1})$ is a span from part1 and $span_i^1 = (x_t, ..., x_{t+l_2})$ is a span from part2 of the input. Note that in Section §2.1, we use a_k^x to denote the importance score of the kth most important explanation of instance x; here, we only focus on the attribution scores of explanations without ranking them.

Shapley. For `TokenEx`, we employ the SHAP method to assign importance scores to each token within the input by removing each token separately and computing its removal effect on the model prediction with different subsets of other tokens presented to the model, following Lundberg and Lee (2017).

$$Shap_{\texttt{TokenEx}}(x_i) = \sum_{S \subseteq F \setminus \{x_i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f(S \cup \{x_i\}) - f(S)] \quad (21)$$

$Shap_{\texttt{TokenEx}}(x_i)$ denotes the importance score of token x_i. As the calculation of $Shap_{\texttt{TokenEx}}(x_i)$ is computationally expensive, we utilize Kernel SHAP to approximate these Shapley values.

For TokenIntEx, we first apply Bivariate Shapley (Masoomi et al., 2022) to assess the mutual importance scores between two tokens, which are from different parts of the input, within a token interaction, and then average these two mutual importance scores as importance score of this token interaction. Specifically, to compute the importance score of a token x_i conditioned on the presence of token x_j, the sets of tokens S considered are limited to those containing token x_j, while the impact of other sets of tokens influences the importance of x_i is ignored in this case. Thus, $Shap_{\texttt{TokenIntEx}}(x_i \mid x_j)$ can be calculated by:

$$Shap_{\texttt{TokenIntEx}}(x_i \mid x_j) = \sum_{x_j \in S \subseteq F \setminus \{x_i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f(S \cup \{x_i\}) - f(S)] \tag{22}$$

The importance score $Shap_{\texttt{TokenIntEx}}(x_i, x_j)$ for token interaction (x_i, x_j) are calculated by averaging $Shap_{\texttt{TokenIntEx}}(x_i \mid x_j)$ and $Shap_{\texttt{TokenIntEx}}(x_j \mid x_i)$. We also use Kernel Shapley to approximate the calculation of Bivariate Shapley value.

For SpanIntEx, we first apply the Louvain Community Detection algorithm (Blondel et al., 2008) to extract span interactions and then average the importance scores of token interactions comprised in each span interaction as its importance score, following Ray Choudhury et al. (2023).

To extract span interactions, we first construct a directed bipartite graph for instance x, by taking each token x_i from the input as node i and the mutual importance scores between each two tokens from different parts obtained above as the weights of directed edges connecting them. Louvain Community Detection algorithm is then applied to search for communities of nodes with dense intra-cluster and sparse inter-cluster relationships. With each community of nodes(tokens) S_p obtained, we can get one span interaction $(span_p^0, span_p^1)$, where the two spans consist of neighboring tokens from the part1 subset of this community S_p^0, and the part2 subset of it S_p^1 respectively.

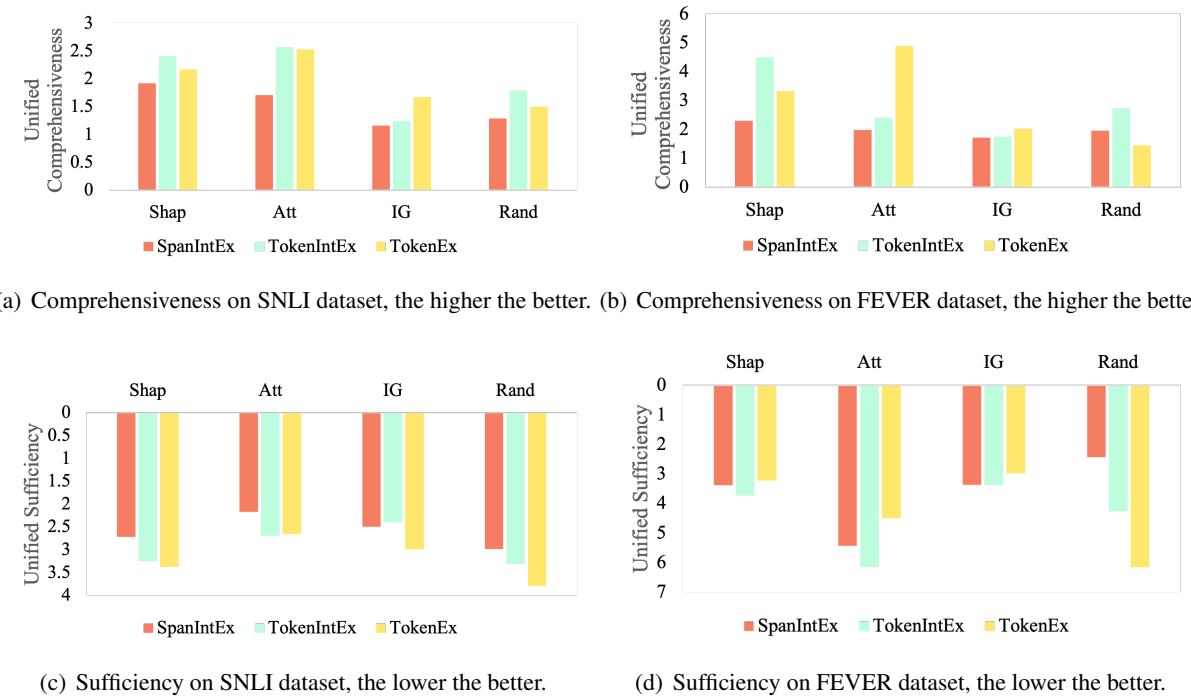
Then we calculate the importance score of this span interaction by averaging the importance scores of all token interactions it comprises.

$$Shap_{\texttt{SpanIntEx}}(span_p^0, span_p^1) = \frac{\sum_{\substack{x_i \in S_p^0 \\ x_j \in S_p^1 \\ i,j}} Shap_{\texttt{TokenIntEx}}(x_i, x_j)}{|S_p^0 \cup S_p^1|} \tag{23}$$

Note that in the following, no matter which explainability techniques to assign importance score to TokenIntEx, we apply the same method as stated above to extract span interactions and compute their importance scores, $A_{\texttt{SpanIntEx}}(span_p^0, span_p^1)$, based on corresponding token interaction importance score, $A_{\texttt{TokenIntEx}}(x_i, x_j)$.

Attention. For each token within the input sequence, we use the self-attention weights between this token and the first token as an indicator of its importance score (Jain and Wallace, 2019). We follow Ray Choudhury et al. (2023) to select the most important attention head in the last layer of the model to obtain these attention weights. For each possible token interaction, we use the method by Clark et al. (2019) to extract and average the attention weights between token pairs from different parts of the input to derive their importance scores, also from the most important head of the last layer. To obtain span interactions and assign them importance scores, we apply the same method to these token interaction scores as described above.

Integrated Gradients. To calculate the importance score for each token in the input sequence, we integrate the gradients of the model's output with respect to each token embedding, following Sundararajan et al. (2017). For generating the importance scores of token interactions, we use Layer-wise Attention Attribution (Ye et al., 2021), which attributes attention links between pairs of tokens within attention maps with a mechanism similar to Integrated Gradients. These attribution maps are created for each model layer and then aggregated across layers to form a final attribution map. The importance score for each token interaction is calculated as the average value from this final attribution map between the involved tokens. For span interactions, we generate and assign importance scores using the same approach based on the importance scores of the token interactions.



(a) Comprehensiveness on SNLI dataset, the higher the better. (b) Comprehensiveness on FEVER dataset, the higher the better.

(c) Sufficiency on SNLI dataset, the lower the better. (d) Sufficiency on FEVER dataset, the lower the better.

Figure 3: Unified Comprehensiveness and Sufficiency of three types of feature attribution explanations on SNLI and FEVER datasets using the BERT model. Subfigures (a) and (c) show Unified Comprehensiveness results, while (b) and (d) show Unified Sufficiency results. Explanations are generated by Shapley (Shap), Attention (Att), and Integrated Gradients (IG). Randomly selected span pairs, token pairs, and tokens are baselines corresponding to explanation type `SpanIntEx`, `TokenIntEx`, and `TokenEx` and form the group Random baseline (Rand). We set $k = 3$ for top span interactions and adjust token counts as per §2.2, also ensuring the random baseline matches the average token count of the top k span interactions.

F Detailed Experiment Results

F.1 Faithfulness

F.2 Agreement with Human Annotation

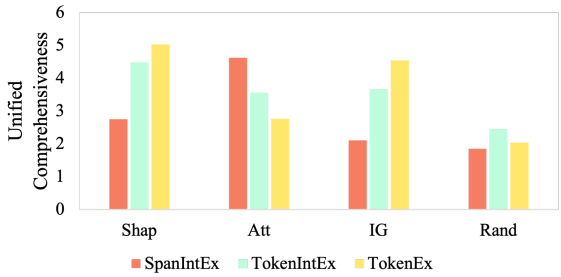
There is a notable gap between interaction-level and token-level agreement scores. For example, in Table 3, the highest interaction-level agreement score for `SpanIntEx` explanations is 57.40%, while the highest token-level agreement score for `SpanIntEx` is 78.26%. A similar pattern is observed for `TokenIntEx`. This suggests that although `SpanIntEx` and `TokenIntEx` explanations align more with human reasoning than `TokenEx` explanations, pairing important tokens or spans into interactions that are plausible to humans remains challenging.

E	Shap	Att	IG	Rand
	Interaction level agreement			
`SpanIntEx`	**30.18**	**57.40**	**39.40**	**33.82**
`TokenIntEx`	29.02	37.02	35.06	23.42
`TokenEx`	-	-	-	-
	Token level agreement			
`SpanIntEx`	**75.63**	78.26	**76.52**	**76.96**
`TokenIntEx`	74.89	**79.60**	73.19	74.96
`TokenEx`	75.54	77.62	70.74	76.33

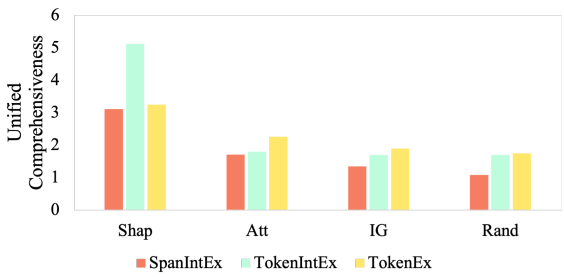
Table 3: Human Annotation Agreement Results (see §2.3) on SNLI dataset when explanations are generated based on BERT. Interaction-level and Token-level agreement scores, Average Precision(%), compared to human annotations for explanation types `SpanIntEx`, `TokenIntEx`, `TokenEx` generated by Shapley(**Shap**), Attention(**Att**), Integradiant Gradients(**IG**) respectively. Using the same attribution method, the highest alignment score for each category is highlighted in bold. **Rand** indicates the random baseline as described in §2.3.

F.3 Simulatability

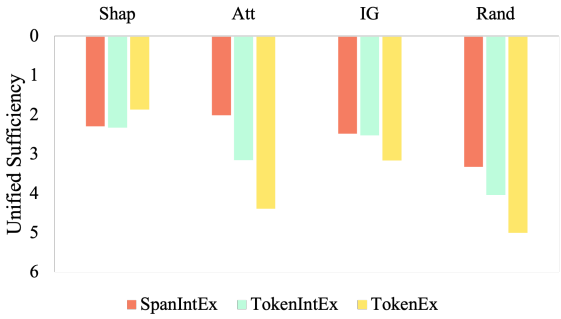
Regarding insertion formats, for BERT models, text insertion (I_{Text}), which adds explanation text to the end of the input sequence, consistently outperforms symbol insertion (I_{Sym}), where symbols are added to the original input sequence, as shown



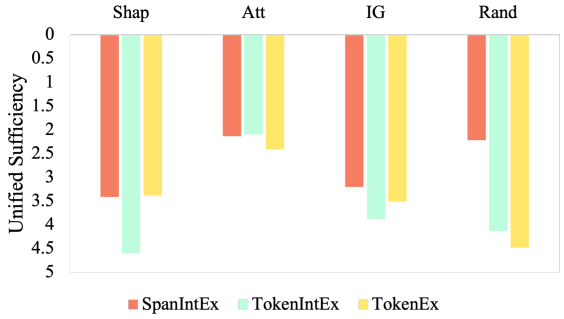
(a) Comprehensiveness on SNLI dataset, the higher the better.



(b) Comprehensiveness on FEVER dataset, the higher the better.



(c) Sufficiency on SNLI dataset, the lower the better.



(d) Sufficiency on FEVER dataset, the lower the better.

Figure 4: Unified Comprehensiveness and Sufficiency of three types of feature attribution explanations on SNLI and FEVER datasets using the BART model. Subfigures (a) and (c) show Unified Comprehensiveness results, while (b) and (d) show Unified Sufficiency results. Explanations are generated by Shapley (Shap), Attention (Att), and Integrated Gradients (IG). Randomly selected span pairs, token pairs, and tokens are baselines corresponding to explanation type `SpanIntEx` `TokenIntEx` and `TokenEx` and form the group Random baseline (Rand). We set $k = 3$ for top span interactions and adjust token counts as per section §2.2, ensuring the random baseline matches the average token count of the top k span interactions.

E	Shap	Att	IG	Rand
	Interaction level agreement			
`SpanIntEx`	**19.92**	**28.12**	**27.45**	**19.33**
`TokenIntEx`	3.96	10.27	21.30	10.23
`TokenEx`	-	-	-	-
	Token level agreement			
`SpanIntEx`	**66.95**	**68.71**	**72.24**	**67.5**
`TokenIntEx`	66.90	67.29	70.50	65.86
`TokenEx`	58.07	56.88	61.07	63.10

Table 4: Human Annotation Agreement Results (see §2.3) on the FEVER dataset when explanations are generated based on BERT. The rest of the settings are the same as Table 3.

E	Shap	Att	IG	Rand
	Interaction level agreement			
`SpanIntEx`	**37.36**	**47.33**	**34.18**	**28.25**
`TokenIntEx`	32.36	35.17	33.06	13.80
`TokenEx`	-	-	-	-
	Token level agreement			
`SpanIntEx`	80.16	**76.28**	82.76	70.04
`TokenIntEx`	**84.9**	75.92	**86.06**	**75.32**
`TokenEx`	65.74	73.71	70.44	73.34

Table 5: Human Annotation Agreement Results (see §2.3) on SNLI dataset when explanations are generated based on BART. The rest settings are the same as Table 3.

E	Shap	Att	IG	Rand
	Interaction level agreement			
SpanIntEx	**22.86**	**20.66**	**18.72**	**16.76**
TokenIntEx	4.71	2.64	10.54	8.64
TokenEx	-	-	-	-
	Token level agreement			
SpanIntEx	**68.77**	**65.33**	70.22	**69.51**
TokenIntEx	67.01	63.40	**70.98**	68.11
TokenEx	59.43	52.15	57.56	60.93

Table 6: Human Annotation Agreement Results (see §2.3) on FEVER dataset when explanations are generated based on BART. The rest settings are the same as Table 3.

D	E	Shap		Att		IG	
		SF	RSF	SF	RSF	SF	RSF
SNLI	SpanIntEx	**87.9**	**3.2**	**86.7**	**2.0**	**88.9**	**4.2**
	TokenIntEx	86.6	1.9	85.3	0.6	85.8	1.1
	TokenEx	87.4	2.7	85.7	1.0	86.0	1.3
FEVER	SpanIntEx	83.9	3.8	**85.3**	**5.2**	**85.2**	**5.1**
	TokenIntEx	82.7	2.6	84.0	**3.9**	84.4	4.3
	TokenEx	**84.0**	**3.9**	82.3	2.2	81.8	1.7

Table 7: Simulatability results on SNLI and FEVER with BERT as the model used for all explanations $E \in$ SpanIntEx, TokenIntEx, TokenEx generation with attribution method **Shapley**, **Attention**, and **Integrated Gradients** respectively. Note that insertion form I_{Sym} is adopted for combining the explanations and the original input sequence for agent model AM_E, as depicted in §2.4. The agent models used for baseline AM_O, trained without explanations, have simulation F1 scores, as denoted in §2.4, of 84.7% and 80.0% on test set shared with other agent models AM_E, as denoted in §2.4. We set $k = 1$ for top SpanIntEx and calculated the number of top TokenIntEx and TokenEx accordingly as stated in §2.2. The largest increases are highlighted in bold for each dataset with the identical attribution method.

in Tables 7 and 8. However, the opposite effect is observed for BART models, as shown in Tables 9 and 10. This indicates that simulatability results are sensitive to the explanation insertion form, highlighting the need for consistency in insertion form when comparing different explanation types.

F.4 Complexity

D	E	Shap		Att		IG	
		SF	RSF	SF	RSF	SF	RSF
SNLI	SpanIntEx	**87.8**	**3.1**	87.1	2.4	88.2	3.5
	TokenIntEx	86.5	1.8	**87.8**	**3.1**	86.4	1.7
	TokenEx	87.0	2.3	86.3	1.6	**88.4**	**3.7**
FEVER	SpanIntEx	85.7	5.6	85.1	5.0	**86.0**	**5.9**
	TokenIntEx	81.9	1.8	**85.6**	5.5	84.3	4.2
	TokenEx	**85.8**	**5.7**	84.5	4.4	82.0	1.9

Table 8: Simulatability results on SNLI and FEVER with BERT as the model used for all input feature explanation generation. Note that insertion form I_{Text} is adopted for combining the explanations and the original input sequence for agent model AM_E, as depicted in §2.4. The agent models used for baseline AM_O, which are trained without explanations, have simulation F1 scores of 84.7% and 80.0% on the test sets shared with agent model AM_E. The other setting is the same as Table 7

D	E	Shap		Att		IG	
		SF	RSF	SF	RSF	SF	RSF
SNLI	SpanIntEx	87.8	7.9	**87.3**	**7.4**	**86.5**	**6.6**
	TokenIntEx	83.5	3.6	84.2	4.3	84.6	4.7
	TokenEx	**88.2**	**8.3**	81.4	1.5	85.8	5.9
FEVER	SpanIntEx	**80.6**	7.2	**76.1**	**2.7**	**75.2**	**1.8**
	TokenIntEx	78.9	5.5	75.9	2.5	74.7	1.3
	TokenEx	80.1	**6.7**	75.0	1.6	74.1	0.7

Table 9: Simulatability results on SNLI and FEVER with BART as the model used for all input feature explanation generation. Note that insertion form I_{Sym} is adopted for combining the explanations and the original input sequence for agent model AM_E, as depicted in §2.4. The base agent models AM_O trained without explanations have the simulation f1 scores of 79.9% and 73.4%, respectively on the test sets sharing with other agent models AM_E. The other setting is the same as Table 7

D	E	Shap		Att		IG	
		SF	RSF	SF	RSF	SF	RSF
SNLI	SpanIntEx	**86.8**	**6.9**	**85.0**	**5.1**	84.3	**4.4**
	TokenIntEx	81.2	1.3	82.6	2.7	**81.6**	1.7
	TokenEx	83.3	3.4	83.8	3.9	82.2	2.3
FEVER	SpanIntEx	**78.2**	**4.8**	**75.3**	**1.9**	**74.6**	**1.2**
	TokenIntEx	74.8	1.4	74.1	0.7	73.9	0.5
	TokenEx	77.6	4.2	74.9	1.5	73.6	0.3

Table 10: Simulatability results on SNLI and FEVER with BART as the model used for all input feature explanation generation respectively. Note that insertion form I_{Text} is adopted for combining the explanations and the original input sequence for agent model AM_E, as depicted in §2.4. The base agent models AM_O trained without explanations have the simulation f1 scores of 79.9% and 73.4%, respectively. The other setting is the same as Table 7

Dataset	E	Shapley	Attention	IG	R	U
SNLI	`SpanIntEx`	2.05	**1.11**	2.10	2.19	2.62
	`TokenIntEx`	**1.72**	1.43	2.30	-	-
	`TokenEx`	2.08	1.64	**1.91**	-	-
FEVER	`SpanIntEx`	2.78	**2.22**	2.90	2.98	3.18
	`TokenIntEx`	3.12	3.07	3.15	-	-
	`TokenEx`	**2.76**	**2.22**	**2.57**	-	-

Table 11: Complexity results on SNLI and FEVER datasets for three types of explanations generated by different attribution methods based on BERT model. The **Random** baseline represents the complexity score obtained by randomly generated scores in the range [0,1], ensuring the same number of scores as the number of explanations used. The **Upperbound** is calculated by setting all the attribution scores to the same value while ensuring the same number of scores as the number of explanations used. The lowest complexity score for each specific explanation type compared is highlighted in bold when the explanations are generated by each attribution method.

Dataset	E	Shapley	Attention	IG	R	U
SNLI	`SpanIntEx`	1.90	**1.93**	**1.63**	2.36	2.76
	`TokenIntEx`	2.50	2.53	2.13	-	-
	`TokenEx`	**1.86**	1.95	1.94	-	-
FEVER	`SpanIntEx`	**2.73**	**3.03**	**2.38**	3.13	3.38
	`TokenIntEx`	3.30	3.36	2.93	-	-
	`TokenEx`	2.82	3.07	3.19	-	-

Table 12: Complexity results on SNLI and FEVER datasets for three types of explanations generated by different attribution methods based on BART model. The other settings are the same as Table 11.