Sequential Integrated Gradients: a simple but effective method for explaining language models

Joseph Enguehard
Babylon Health
Skippr
joseph@skippr.com

Abstract

Several explanation methods such as Integrated Gradients (IG) can be characterised as path-based methods, as they rely on a straight line between the data and an uninformative baseline. However, when applied to language models, these methods produce a path for each word of a sentence simultaneously, which could lead to creating sentences from interpolated words either having no clear meaning, or having a significantly different meaning compared to the original sentence. In order to keep the meaning of these sentences as close as possible to the original one, we propose Sequential Integrated Gradients (SIG), which computes the importance of each word in a sentence by keeping fixed every other words, only creating interpolations between the baseline and the word of interest. Moreover, inspired by the training procedure of several language models, we also propose to replace the baseline token "pad" with the trained token "mask". While being a simple improvement over the original IG method, we show on various models and datasets that SIG proves to be a very effective method for explaining language models.[1]

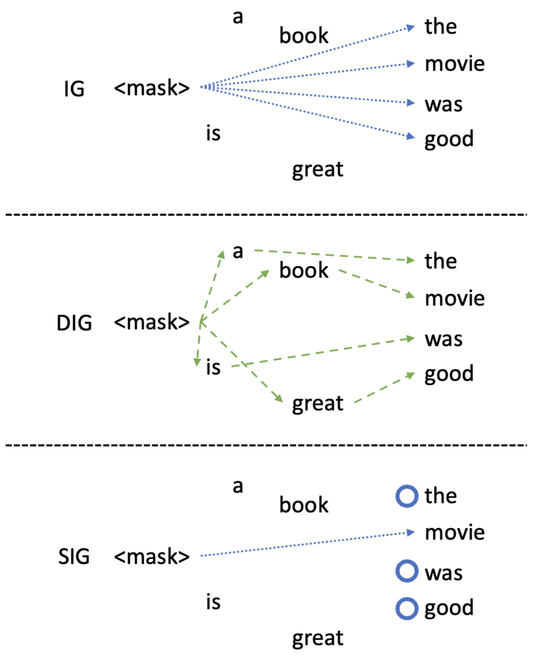

Figure 1: **Comparison between IG, DIG, and our method: SIG.** While DIG improves on IG by creating discretized paths between the data and the baseline, it can produce sentences with a different meaning compared to the original one. Our method tackles this issue by fixing every word to their true value except one, and moving the remaining word along a straight path (SIG)

1 Introduction

Language models such as BERT (Devlin et al., 2018) have demonstrated to be effective on various tasks, for instance on sentiment analysis (Hoang et al., 2019), machine translation (Zhu et al., 2020), text summarization (Liu, 2019) or intent classification (Chen et al., 2019). However, with the increased performance and usage of such models, there has been a parallel drive to develop methods to explain predictions made by these models. Indeed, BERT and its variations are complex models which do not allow a user to easily understand why a certain prediction has been produced. On the other hand, it is important to be able to explain a model's predictions, especially when this model is used to make high-stake decisions, or when there is a risk of a discriminating bias, for instance when detecting hate speech on social media (Sap et al., 2019).

As a result, developing effective methods to explain not only language models, but also machine learning models in general, has recently gained significant attention. Many different methods have therefore been proposed such as: LIME (Ribeiro et al., 2016), Grad*Inp (Shrikumar et al., 2016), Integrated Gradients (IG) (Sundararajan et al., 2017), DeepLift (Shrikumar et al., 2017) or GradientShap (Lundberg and Lee, 2017). Among these methods, some can be characterised as path-based, which

[1]An implementation of this work can be found at `https://github.com/josephenguehard/time_interpret`

means that they rely on a straight line between the data and an uninformative baseline. For instance, IG computes gradients on interpolated points along such a path, while DeepLift and GradientShap can be seen as approximations of IG (Ancona et al., 2017; Lundberg and Lee, 2017).

While these methods aim to be used on any type of models and data, some have been tailored to the specificity of language models. For instance, Sanyal and Ren (2021) challenge the use of continuous paths on a word embedding space which is inherently discrete. They propose as a result Discretized Integrated Gradient (DIG), which replaces the continuous straight path with a discretized one, where interpolated points are words.

In our work, we suggest another potential issue when applying path-based explanation methods on language models. These models are usually designed to be used on individual or multiple sentences, in order to perform for instance sentiment analysis or question answering. However, a path-based method applied on such models creates straight lines between each word and a baseline simultaneously. When interpolated points are grouped together to form a sentence, this sentence could have a very different meaning compared with the original one.

As a result, we propose a simple method to alleviate this potential issue: computing the importance of each word in a sentence or a text by keeping fixed every other word and only creating interpolations between the baseline and the word of interest. After computing the importance of each word in this way, we normalise these attributions across the sentence or text we aim to explain. We call this method Sequential Integrated Gradients (SIG), as, although we focus in this work on language models, such a method could be used on any sequential modelling. We also propose to use the token "mask" as a baseline, when possible, as its embedding has been trained to replace part of sentences when training language models. As a result, our method follows closely the training procedure of these models.

2 Method

SIG formulation Let's define a language model as a function $\text{F}(\mathbf{x}) : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$. The input $\mathbf{x}$ is here modelled as a sequence of m words, each having n features. These features are usually constructed by an embedding layer. We denote $\mathbf{x}_i$ the ith word of a sentence (or of a text, depending on the input of the model), and x_{ij} the jth feature of the ith word. The output of F is a value in $\mathbb{R}$, which is, in our experiments, a measure of the sentiment for a given sentence. We now define the baseline for each word $\mathbf{x}_i$ as $\overline{\mathbf{x}}^i = (\mathbf{x}_1, ..., \texttt{<mask>}, ..., \mathbf{x}_m)$. The baseline is therefore identical to $\mathbf{x}$ except at the ith position, where the word $\mathbf{x}_i$ is replaced by the embedding of the word "mask"[2], a token used in many language model to replace part of the sentence during training. Moreover, we use the notation $\overline{\mathbf{x}}^i$ instead of $\overline{\mathbf{x}}_i$ as $\overline{\mathbf{x}}^i$ corresponds to an entire sentence, not to be mistaken with a single word like $\mathbf{x}_i$.

In this setting, we keep the baseline as similar to the original sentence as possible, only changing the word of interest. This method of explaining a word is also kept similar to the way these language models are usually pre-trained, by randomly masking part of sentences.

Let's now define our Sequential Integrated Gradients (SIG) method. For a word $\mathbf{x}_i$ and a feature j, SIG is defined as:

$$\text{SIG}_{ij}(\mathbf{x}) := (x_{ij} - \overline{x}_{ij}) \times \int_0^1 \frac{\partial \text{F}(\overline{\mathbf{x}}^i + \alpha \times (\mathbf{x} - \overline{\mathbf{x}}^i))}{\partial x_{ij}} \, d\alpha$$

Similar to the original IG (Sundararajan et al., 2017), we compute the gradient of F along a straight line between $\overline{\mathbf{x}}^i$ and $\mathbf{x}$ for each word $\mathbf{x}_i$, the main difference being that the baseline differs for each word. Also similar to the original IG, we approximate in practice the integral with Riemann summation.

Finally, we compute the overall attribution of a word by computing the sum over the feature dimension j, and normalising the result:

$$\text{SIG}_i(\mathbf{x}) := \frac{\sum_j \text{SIG}_{ij}}{||\text{SIG}||}$$

Axioms satisfied by SIG The original Integrated Gradients method satisfies a few axioms that are considered desirable for any explanation methods to have. Among these axioms, SIG follows implementation invariance, which states that attributions should be identical if two models are functionally equivalent. Moreover, SIG follows completeness

[2]Certain language models, such as GPT-2 (Radford et al., 2019), do not have a "mask" token. A "pad" token should be therefore used for such models.

Method	**DistilBERT**			**RoBERTa**			**BERT**		
	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓
Grad*Inp	-0.412	0.112	0.375	-0.199	0.0760	0.426	-0.263	0.0923	0.439
DeepLift	-0.624	0.170	0.271	-0.261	0.0932	0.408	-0.244	0.0898	0.438
GradientShap	-1.32	0.303	0.258	-0.896	0.261	0.314	-0.622	0.219	0.388
IG	-1.96	0.445	0.151	-1.44	0.405	0.226	-0.981	0.345	0.352
DIG	-1.69	0.384	0.167	-0.824	0.263	0.278	-0.777	0.287	0.345
SIG	**-2.02**	**0.473**	**0.0992**	**-1.62**	**0.440**	**0.216**	**-1.19**	**0.392**	**0.312**

Table 1: Comparison of SIG with several feature attribution methods on three language models fine-tuned on the SST2 dataset. For ↑ metrics, the higher the better, while for ↓ ones, the lower the better.

Method	**DistilBERT**			**RoBERTa**			**BERT**		
	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓
Grad*Inp	-0.153	0.0766	0.209	-0.0892	0.0432	0.300	-0.291	0.0887	0.298
DeepLift	-0.269	0.117	0.159	-0.124	0.0557	0.269	-0.285	0.0701	0.366
GradientShap	-0.832	0.289	0.137	-0.606	0.204	0.144	-0.874	0.172	0.308
IG	-1.50	0.534	0.0428	-1.35	**0.441**	0.0327	-1.58	0.302	0.224
DIG	-0.779	0.304	0.133	-0.663	0.186	0.108	-1.06	0.207	0.232
SIG	**-1.95**	**0.564**	**0.00409**	**-1.37**	0.404	**-3.31E-05**	**-2.12**	**0.364**	**0.124**

Table 2: Comparison of SIG with several feature attribution methods on three language models fine-tuned on the IMDB dataset.

in a specific way: for each word $\mathbf{x}_i$, we have the following result:

$$\sum_j SIG_{ij}(\mathbf{x}) = \mathrm{F}(\mathbf{x}) - \mathrm{F}(\overline{\mathbf{x}}^i)$$

This means that for each word, the sum of its attribution across all features j is equal to the difference between the output of the model as $\mathbf{x}$ and at its corresponding baseline $\overline{\mathbf{x}}^i$. However, it does **not** entail that $\sum_{ij} SIG_{ij}(\mathbf{x}) = \mathrm{F}(\mathbf{x}) - \mathrm{F}(\overline{\mathbf{x}})$, where $\overline{\mathbf{x}}$ would be an overall baseline filled with `<mask>`.

Moreover, this last axiom entail another one called sensitivity, which here means that if, for a certain word, the input $\mathbf{x}$ has the same influence on the output of F as its corresponding baseline $\overline{\mathbf{x}}^i$, then $\sum_j SIG_{ij}(\mathbf{x}) = 0$.

Finally, we show in Appendix A that SIG preserves symmetry for each word on the embedding dimension, but that this axiom is not true in general.

Using mask instead of pad as a baseline We propose in this study to replace, as the baseline, the commonly used "pad" token with the "mask" token, on language models having such token. This seems to go against the intuition that the baseline should be uninformative, as "mask" is a trained token. To support the usage of "mask", we argue that, because `<PAD>` (denoting the embedding of "pad") is untrained, it could be arbitrarily close to some words, and far from others. Oh the other hand, `<MASK>` has been trained to replace random words, making it ideally as close to one word as to any other.

Another way to see it is to compare it with images. It is natural for images to choose the baseline as a black image, as this baseline has no information. However, there is no such guarantee in NLP. For instance, the embedding of "pad": `<0, 0, 0, . . . , 0>` could perfectly be very close to an embedding of a word with a specific meaning, which would harm the explanations. On the other hand, `<MASK>` has been trained to replace any word, and therefore seems more suited to be the baseline.

3 Experiments

3.1 Experiments design

We evaluate SIG against various explanation methods by closely following the experimental setup of Sanyal and Ren (2021). As such, we use the following language models: BERT (Devlin et al., 2018), DistilBERT (Sanh et al., 2019) and RoBERTa (Liu

Method	DistilBERT			RoBERTa			BERT		
	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓
Grad*Inp	-0.257	0.0681	0.315	-0.121	0.0617	0.363	-0.438	0.143	0.438
DeepLift	-0.332	0.101	0.260	-0.163	0.0804	0.348	-0.452	0.123	0.450
GradientShap	-0.452	0.237	0.212	-0.389	0.194	0.299	-0.715	0.204	0.438
IG	**-0.540**	**0.341**	0.163	-0.787	0.354	**0.242**	-1.19	0.307	0.410
DIG	-0.487	0.273	0.181	-0.426	0.223	0.286	-1.05	0.293	0.414
SIG	-0.533	0.331	**0.134**	**-0.869**	**0.361**	0.251	**-1.52**	**0.390**	**0.349**

Table 3: Comparison of SIG with several feature attribution methods on three language models fine-tuned on the Rotten Tomatoes dataset.

	steps	LO ↓	Comp ↑	Suff ↓	Delta	Time
IG	50	-0.981	0.345	0.352	0.304	t
SIG	50	**-1.19**	**0.392**	**0.312**	4.82	N × t
IG	250	-0.999	0.352	0.355	0.055	t'
IG	10 × N	-0.998	0.351	0.352	0.066	N × t' / 25
SIG	10	**-1.14**	**0.373**	**0.322**	4.93	N × t' / 25

Table 4: Comparison of IG and SIG with different numbers of interpolations on BERT fine-tuned on the SST2 dataset.
t and t' represent the amount of time to calculate IG with 50 and 250 steps respectively, and N represents the number of words on the input data (for instance in one sentence). On the SST2 dataset, we have an average of: N $\approx$ 25 words per sentence.
On top of the table, we compare IG and SIG using a fixed number of steps. On the bottom of the table, we compare IG with 250 steps against SIG with 10 steps. Since N $\approx$ 25, we have N $\times$ t' / 25 $\approx$ t'. For a fairer comparison, we also compare IG with a variable number of steps: 10 $\times$ N for each sentence, against SIG with 10 steps. These two methods have the same time complexity.
Delta is defined as $\sum_{ij} Attr_{ij}(\mathbf{x}) - (\mathrm{F}(\mathbf{x}) - \mathrm{F}(\bar{\mathbf{x}}))$. Contrary to IG, SIG has a high delta value, as in general $\sum_{ij} SIG_{ij}(\mathbf{x}) \neq \mathrm{F}(\mathbf{x}) - \mathrm{F}(\bar{\mathbf{x}})$.

et al., 2019). We also use the following datasets: SST2 (Socher et al., 2013), IMDB (Maas et al., 2011) and Rotten Tomatoes (Pang and Lee, 2005), which classify sentences into positive or negative sentiments or reviews. Moreover, we use the HuggingFace library to recover processed data and pretrained models (Wolf et al., 2019).

Following (Sanyal and Ren, 2021), we use the following evaluation metrics: Log-Odds (Shrikumar et al., 2017), Comprehensiveness (DeYoung et al., 2019) and Sufficiency (DeYoung et al., 2019). These metrics mask the top or bottom 20 % important features, according to an attribution method, and measure by how much the prediction of the language model changes using this masked data, compared with the original one. For more details on these metrics, please see Sanyal and Ren (2021).

Finally, we use the following feature attribution methods to compare our methods against: Grad*Inp (Shrikumar et al., 2016), Integrated Gradients (Sundararajan et al., 2017), DeepLift (Shrikumar et al., 2017), GradientShap (Lundberg and Lee, 2017) and Discretized IG (DIG) (Sanyal and Ren, 2021) using the GREEDY heuristics. Moreover, as in Sanyal and Ren (2021), we use 50 interpolation steps for all methods expect from DIG, for which we use 30 steps.

3.2 Results

Comparison with other feature attribution methods We present of Tables 1, 2 and 3 a comparison of the performance of SIG with the attribution methods listed in 3.1. We observe that SIG significantly outperforms all other methods across most datasets and language models we used. This tends to confirm that the change of overall meaning of a sentence by combining interpolations simultaneously is an important issue which needs to be tackled.

Comparison between IG and DIG Although results in Sanyal and Ren (2021) show that DIG outperforms other methods, including IG, this is not the case when using "mask" as a token. This result seems to undermine the intuition of Sanyal and Ren (2021) that the discrete nature of the embedding space is an important factor when explaining a language model. We also show in Appendix C that the requirement of having a monotonic path, stressed by Sanyal and Ren (2021), is not necessary.

Method	Example
IG	“a well-made and often **lovely** **depiction** of the **mysteries** of friendship.
SIG	“a **well-made** and often **lovely** depiction of the mysteries of friendship.
IG	“**a** hideous , **confusing** spectacle , **one** that may well put **the** nail in the coffin of any future rice adaptations.”
SIG	“a **hideous** , **confusing** spectacle , **one** that may well put the nail in the coffin of any future **rice** adaptations.”
IG	"this is **junk** food cinema at its **gr**easiest."
SIG	"this is **junk** food cinema at its gr**ea**siest."
IG	"a **remarkable** 179-minute **meditation** on the nature of revolution."
SIG	"a **remarkable** 179-minute **meditation** on the nature of revolution."

Table 5: Examples of attributions on several sentences of the SST2 dataset. The **underlined bold** tokens represent the most important token in the sentence, while **bold** tokens represent the top 20 % tokens in the sentence, according to each attribution method.

Choice of the baseline token We also provide in Appendix B results using "pad" as a baseline. Comparison between Tables 1, 2 and 3 on one hand, and Tables 6, 7, 8 on the other hand show that IG greatly improves using the "mask" token as a baseline. This seems to confirm our intuition of using this token instead of "pad". Moreover, SIG performs similarly using either token, which demonstrates the robustness of this method across these two baseline tokens.

Time complexity of SIG One important drawback of SIG is its time complexity, which is dependent on the number of words in the input data. In Table 4, we compare the original IG with SIG, using different numbers of steps. We define t and t' as the time complexity of computing IG with respectively 50 and 250 steps, and N the number of words in the input data. This table shows that, although reducing the number of steps results in a decrease of performance, SIG with 10 steps still performs better than both IG with 250 steps and IG with 10 $\times$ N steps, while having the same time complexity.

Moreover, as noted in Sanyal and Ren (2021), using IG with a large number of steps decreases Delta = $\sum_{ij} IG_{ij}(\mathbf{x}) - (\mathrm{F}(\mathbf{x}) - \mathrm{F}(\overline{\mathbf{x}}))$, while not improving performance. As a result, when computing attributions on long sentences or large texts, we recommend using SIG with a reduced number of steps instead of IG.

Comparison of IG and SIG on several examples We provide on Table 5 several examples of explained sentences, using IG and SIG. Both methods tend to agree on short sentences, while more disagreements appear on larger ones. For each example, we display in underlined bold the most important token, and in bold the top 20 % most important tokens, according to each method.

4 Conclusion

In this work, we have defined an attribution method specific to text data: Sequential Integrated Gradients (SIG). We have shown that SIG yields significantly better results than the original Integrated Gradients (IG), as well as other methods specific to language models, such as Discretized Integrated Gradients (DIG). This suggests that keeping the meaning of interpolated sentences close to the original one is key to producing good explanations. We have also shown that, although SIG can be computationally intensive, reducing the number of interpolations still yields better results than IG with a greater number of interpolations.

We have also highlighted in this work the benefit of using the token "mask" as a baseline, instead of "pad". Although SIG seems to be robust across both tokens, this is especially important when using IG, as it significantly improves the quality of explanations. Using the trainable token "mask" is indeed closer to the training procedure of language models, and should yield better interpolations as a result. We recommend therefore using this token as a baseline, when possible, when explaining predictions made by a language model.

Moreover, while this study was conducted on bidirectional language models such as BERT, SIG could also be used on auto-regressive models such as GPT-2 (Radford et al., 2019), by iteratively computing the attribution of a token, while keeping previous tokens fixed, and masking future tokens if any has been already computed.

Limitations

We see two main limitations of this work. The first one concerns the diversity of the language models and datasets used. BERT, DistilBERT and RoBERTa have similar architecture, and SST2, IMDB and Rotten Tomatoes are datasets designed to evaluate the sentiment of English text. It would therefore be interesting to validate the robustness of our results on more diverse languages, tasks and language models. In this short paper, we decided for brevity to follow the experiment design of Sanyal and Ren (2021), while being aware of its inherent limitations.

The second limitation of this work concerns the time complexity of SIG. As it needs to compute explanations for each word individually, this method can become very computationally expensive when applied on large text data. To alleviate this issue, we first made it possible to compute gradients in parallel, using an internal batch size similar to how Captum (Kokhlikyan et al., 2020) implemented the Integrated Gradients method. Secondly, as discussed in 3.2, it is possible to reduce the number of interpolated points, which makes the computation faster while retaining better performance than the original IG.

In this work, we ran our experiments on a machine with 16 CPUs, and one Nvidia Tesla T4 GPU. With this setting, computing SIG on SST2 and Rotten Tomatoes takes around one hour for each model. On the larger IMDB, computing SIG, on 2000 randomly sampled inputs, takes around 5 days for BERT and RoBERTa, and 2 days for DistilBERT.

Ethics Statement

The methods presented in this work aim to explain language models, and can as such present ethical issues related to this task. Discriminating biases can indeed be present in text data on which a language model is trained, and such a model can acquire and propagate these biases (Sap et al., 2019). As the presented methods aim to explain a language model without additional knowledge, these methods could also display discriminating biases learnt by a language model.

Moreover, common explanation methods such as Integrated Gradients has proved to be prone to adversarial attacks (Dombrowski et al., 2019), and can be misleading when used on out of sample data (Slack et al., 2021). There is no reason to believe our methods would be more robust compared to existing methods such as IG.

The proposed methods can also be characterised as gradient-based, as they rely on computing gradients on the input data, an uninformative baseline, or on interpolated points between them. As noted by (Mittelstadt et al., 2019), such methods are only local and may not give a clear explanation of the model globally.

Acknowledgement

The author would like to thank Vitalii Zhelezniak for his thoughtful comments and suggestions, including using the "mask" token as a baseline. We also thank Anthony Hu for his detailed initial review of this paper.

References

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. 2017. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. 2019. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*.

Ann-Kathrin Dombrowski, Maximillian Alber, Christopher Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. 2019. Explanations can be manipulated and geometry is to blame. *Advances in Neural Information Processing Systems*, 32.

Mickel Hoang, Oskar Alija Bihorac, and Jacobo Rouces. 2019. Aspect-based sentiment analysis using BERT. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 187–196, Turku, Finland. Linköping University Electronic Press.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for pytorch.

Yang Liu. 2019. Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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

Daniel D Lundstrom, Tianjian Huang, and Meisam Razaviyayn. 2022. A rigorous study of integrated gradients method and extensions to internal neuron attributions. In *International Conference on Machine Learning*, pages 14485–14508. PMLR.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Brent Mittelstadt, Chris Russell, and Sandra Wachter. 2019. Explaining explanations in ai. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 279–288.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Soumya Sanyal and Xiang Ren. 2021. Discretized integrated gradients for explaining language models. *arXiv preprint arXiv:2108.13654*.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1668–1678.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR.

Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. 2016. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*.

Dylan Slack, Anna Hilgard, Himabindu Lakkaraju, and Sameer Singh. 2021. Counterfactual explanations can be manipulated. *Advances in Neural Information Processing Systems*, 34:62–75.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating bert into neural machine translation. *arXiv preprint arXiv:2002.06823*.

A On the symmetry-preserving axiom of Sequential Integrated Gradients

This section is divided into two parts. First, we show that SIG preserves symmetry for each word along the embedding dimension. Second, we provide a counterexample to show that symmetry does not hold in general.

Symmetry of SIG_i Let us use the same notations as in Section 2. We want to compute the attribution of a word $\mathbf{x}_i$ on a model F, using the baseline $\overline{\mathbf{x}}^i$. Let's define the function:

$$\mathrm{F}_i(\mathbf{x}) := \mathrm{F}(\mathbf{x}_1, ..., \mathbf{x}, ..., \mathbf{x}_m)$$

F_i corresponds to F where only the i^{th} word is not fixed. Here, $\mathbf{x}$ corresponds to a word, not a sentence.

For such a function, SIG corresponds to the regular IG method: the baseline is $<$ mask $>$ and SIG constructs a straight line between this baseline and $\mathbf{x}_i$. As a result, if F_i is symmetric on two embedding features j_1 and j_2, SIG preserves this symmetry: $\mathrm{SIG}_{ij_1}(\mathbf{x}) = \mathrm{SIG}_{ij_2}(\mathbf{x})$.

Non symmetry of SIG The fact that SIG does not preserve symmetry in general is due to the choice of the baseline. As a counterexample, let's define a language F which takes as an input two words $\mathbf{x}_1$ and $\mathbf{x}_2$. This language model is moreover symmetric: $\mathrm{F}(\mathbf{x}_1, \mathbf{x}_2) = \mathrm{F}(\mathbf{x}_2, \mathbf{x}_1)$.

Here, the original IG method would preserve the symmetry: as the baseline is (`<mask>`, `<mask>`), when $\mathbf{x}_1 = \mathbf{x}_2$, we have $IG(\mathbf{x})_1 = IG(\mathbf{x})_2$. However, SIG doesn't preserve the symmetry due to its baseline: we would have: $\overline{\mathbf{x}}^1 = (\texttt{<mask>}, \mathbf{x}_2)$ and $\overline{\mathbf{x}}^2 = (\mathbf{x}_1, \texttt{<mask>})$. As a result, $\mathrm{SIG}(\mathbf{x})_1 = \mathrm{SIG}(\mathbf{x})_2$ only if $\mathbf{x}_1 = \mathbf{x}_2 = \texttt{<mask>}$.

B Additional results using the "pad" token

We present in this section results using the "pad" token instead of the "mask" one. These results for the three datasets: SST2, IMDB and Rotten Tomatoes can be found respectively on Tables 6, 7 and 8.

When using the "pad" token as a baseline, SIG seems to perform similarly compared with using the "mask" one, while other methods perform significantly worse. This demonstrates both the need to use "mask" as a token, and the robustness of the SIG method across different baselines.

C Challenge of the monotonic assumption of the path

Sanyal and Ren (2021) stipulate that the path between a baseline and an input needs to be monotonic to allow approximating the integral in IG using Riemann summation. However, while this is true for a Riemann integral, it is also possible to approximate the Riemann–Stieltjes integral, which is a generalisation of Riemann integral, and does not need a monotonic path. We define the Riemann–Stieltjes integral of $f : [\mathbf{a}, \mathbf{b}] \to \mathbb{R}$ as:

$$\int_{\mathbf{x}=\mathbf{a}}^{\mathbf{b}} f(\mathbf{x})\, dg(\mathbf{x})$$

where $g : [0, 1] \to [\mathbf{a}, \mathbf{b}]$ designates a path. Let us define a partition over $[0, 1]$ as t_k such as $0 \leq t_1 \leq ... \leq t_n \leq 1$. We can then approximate the integral with the sum:

$$\sum_{i=0}^{n-1} f(g(c_i)) \times [g(t_{i+1}) - g(t_i)]$$

where $c_i \in [t_i, t_{i+1}]$. As such, while the partition t_i, $i \in \{1, ..., n\}$ needs to be monotonic, the function g does not need to have this constraint. As a result, we could define a path-based IG method as:

$$IG_\gamma(\mathbf{x})_i := \int_\gamma \frac{\partial \mathrm{F}(\mathbf{x})}{\partial x_i}\, dx_i$$

where γ is not necessarily monotonic.

(Lundstrom et al., 2022) provide more insights on this topic, and in particular show that the implementation invariance, completeness and sensitivity axioms hold for non-monotonic paths.

For this reason, we decided not to include a combination of DIG and SIG in this study. However, an implementation of this method and the corresponding results can be found in the repository published with this paper.

Method	DistilBERT			RoBERTa			BERT		
	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓
Grad*Inp	-0.402	0.112	0.375	-0.318	0.085	0.398	-0.454	0.092	0.439
DeepLift	-0.196	0.053	0.489	-0.270	0.0784	0.439	-0.283	0.061	0.463
GradientShap	-0.753	0.191	0.328	-0.514	0.146	0.386	-0.471	0.146	0.425
IG	-0.954	0.251	0.273	-0.726	0.227	0.315	-0.658	0.235	0.398
DIG	-1.222	0.310	0.237	-0.812	0.249	0.287	-0.879	0.292	0.374
SIG	**-1.993**	**0.466**	**0.108**	**-1.346**	**0.398**	**0.244**	**-1.30**	**0.393**	**0.331**

Table 6: Comparison of SIG with several baselines on three language models fine-tuned on the SST2 dataset. For ↑ metrics, the higher the better, while for ↓ ones, the lower the better.

Method	DistilBERT			RoBERTa			BERT		
	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓
Grad*Inp	-0.189	0.082	0.209	-0.216	0.047	0.315	-0.654	0.087	0.299
DeepLift	-0.032	-0.005	0.515	-0.149	0.031	0.374	-0.519	0.027	0.465
GradientShap	-0.315	0.117	0.302	-0.351	0.110	0.213	-0.622	0.088	0.358
IG	-0.474	0.186	0.201	-0.499	0.169	0.114	-0.577	0.117	0.288
DIG	-0.812	0.297	0.153	-0.626	0.187	0.099	-0.971	0.192	0.229
SIG	**-2.157**	**0.585**	**0.0062**	**-0.856**	**0.291**	**0.0207**	**-1.96**	**0.352**	**0.152**

Table 7: Comparison of SIG with several baselines on three language models fine-tuned on the IMDB dataset.

Method	DistilBERT			RoBERTa			BERT		
	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓	LO ↓	Comp ↑	Suff ↓
Grad*Inp	-0.152	0.068	0.315	-0.211	0.062	0.363	-0.806	0.143	0.438
DeepLift	-0.077	0.017	0.372	-0.198	0.056	0.370	-0.457	0.076	0.474
GradientShap	-0.326	0.147	0.250	-0.264	0.103	0.348	-0.697	0.161	0.429
IG	-0.424	0.208	0.190	-0.360	0.151	0.312	-0.795	0.201	0.414
DIG	-0.501	0.257	0.184	-0.346	0.153	0.310	-1.06	0.267	0.416
SIG	**-0.753**	**0.378**	**0.109**	**-0.771**	**0.318**	**0.266**	**-1.55**	**0.360**	**0.393**

Table 8: Comparison of SIG with several baselines on three language models fine-tuned on the Rotten Tomatoes dataset.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section at the end of the paper, before references

☑ A2. Did you discuss any potential risks of your work?
In the introduction and limitation sections

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Left blank.

☑ B1. Did you cite the creators of artifacts you used?
I created my own and used code from https://github.com/INK-USC/DIG

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Open source

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
In the appendix

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
In the experiment section

C ☑ Did you run computational experiments?

In the experiment section

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
The models used are pre-trained, standard language models. The computation budget and infrastructure used is discussed in the ethics statement.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Not applicable. The is no training and hyperparameter search

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
It is just a single run - the results are not stochastic

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Not applicable. Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.