Make Up Your Mind! Adversarial Generation of Inconsistent Natural Language Explanations

Oana-Maria Camburu[1,4] **Brendan Shillingford**[1,2] **Pasquale Minervini**[3]
Thomas Lukasiewicz[1,4] **Phil Blunsom**[1,2]
[1] University of Oxford [2] DeepMind, London
[3] University College London [4] Alan Turing Institute, London
firstname.lastname@cs.ox.ac.uk p.minervini@ucl.ac.uk

Abstract

To increase trust in artificial intelligence systems, a promising research direction consists of designing neural models capable of generating natural language explanations for their predictions. In this work, we show that such models are nonetheless prone to generating mutually inconsistent explanations, such as "*Because there is a dog in the image.*" and "*Because there is no dog in the* [same] *image.*", exposing flaws in either the decision-making process of the model or in the generation of the explanations. We introduce a simple yet effective adversarial framework for sanity checking models against the generation of inconsistent natural language explanations. Moreover, as part of the framework, we address the problem of adversarial attacks with *full* target sequences, a scenario that was not previously addressed in sequence-to-sequence attacks. Finally, we apply our framework on a state-of-the-art neural natural language inference model that provides natural language explanations for its predictions. Our framework shows that this model is capable of generating a significant number of inconsistent explanations.

1 Introduction

In order to explain the predictions produced by accurate yet black-box neural models, a growing number of works propose extending these models with natural language explanation generation modules, thus obtaining models that explain themselves in human language (Hendricks et al., 2016; Camburu et al., 2018; Park et al., 2018; Kim et al., 2018; Ling et al., 2017).

In this work, we first draw attention to the fact that such models, while appealing, are nonetheless prone to generating inconsistent explanations. We define two explanations to be inconsistent if they provide contradictory arguments about the instances and predictions that they aim to explain. For example, consider a visual question answering (VQA) task (Park et al., 2018) and two instances where the image is the same but the questions are different, say "*Is there an animal in the image?*" and "*Can you see a Husky in the image?*". If for the first instance a model predicts "*Yes.*" and generates the explanation "*Because there is a dog in the image.*", while for the second instance the *same* model predicts "*No.*" and generates the explanation "*Because there is no dog in the image.*", then the model is producing inconsistent explanations.

Inconsistent explanations reveal at least one of the following undesired behaviors: *(i)* at least one of the explanations is not faithfully describing the decision mechanism of the model, or *(ii)* the model relied on a faulty decision mechanism for at least one of the instances. Note that, for a pair of inconsistent explanations, further investigation would be needed to conclude which of these two behaviors is the actual one (and might vary for each instance). Indeed, a pair of inconsistent explanations does not necessarily imply at least one unfaithful explanation. In our previous example, if the image contains a dog, it is possible that the model identifies the dog when it processes the image together with the first question, and that the model does not identify the dog when it processes the image together with the second question, hence both explanations would faithfully reflect the decision mechanism of the model even if they are inconsistent. Similarly, a pair of inconsistent explanations does not necessarily imply that the model relies on a faulty decision mechanism, because the explanations may not faithfully describe the decision mechanism of the model. We here will not investigate the problem of identifying which of the two undesired behaviors is true for a pair of inconsistent explanations.

In this work, we introduce a framework for

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4157–4165
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

PREMISE: A guy in a red jacket is snowboarding in midair.	
ORIGINAL HYPOTHESIS: A guy is outside in the snow. PREDICTED LABEL: entailment ORIGINAL EXPLANATION: **Snowboarding is done outside.**	REVERSE HYPOTHESIS: The guy is outside. PREDICTED LABEL: contradiction REVERSE EXPLANATION: **Snowboarding is not done outside.**
PREMISE: A man talks to two guards as he holds a drink.	
ORIGINAL HYPOTHESIS: The prisoner is talking to two guards in the prison cafeteria. PREDICTED LABEL: neutral ORIGINAL EXPLANATION: **The man is not necessarily a prisoner.**	REVERSE HYPOTHESIS: A prisoner talks to two guards. PREDICTED LABEL: entailment REVERSE EXPLANATION: **A man is a prisoner.**
PREMISE: Two women and a man are sitting down eating and drinking various items.	
ORIGINAL HYPOTHESIS: Three women are shopping at the mall. PREDICTED LABEL: contradiction ORIGINAL EXPLANATION: **There are either two women and a man or three women.**	REVERSE HYPOTHESIS: Three women are sitting down eating. PREDICTED LABEL: neutral REVERSE EXPLANATION: **Two women and a man are three women.**

Table 1: Examples of detected inconsistent explanations – the reverse hypotheses generated by our method (right) are realistic.

checking if models are robust against generating inconsistent natural language explanations. Given a model m that produces natural language explanations for its predictions, and an instance $\mathbf{x}$, our framework aims to generate inputs $\hat{\mathbf{x}}$ that cause the model to produce explanations that are inconsistent with the explanation produced for $\mathbf{x}$. Thus, our framework falls under the category of *adversarial methods*, i.e., searching for inputs that cause a model to produce undesired answers (Biggio et al., 2013; Szegedy et al., 2014).

As part of our framework, we address the problem of adversarial attacks with *full* target sequences, a scenario that has not been previously addressed in sequence-to-sequence attacks, and which can be useful for other areas, such as dialog systems. Finally, we apply our framework on a state-of-the-art neural natural language inference model that generates natural language explanations for its decisions (Camburu et al., 2018). We show that this model can generate a significant number of inconsistent explanations.

2 Method

Given a model m that can jointly produce predictions and natural language explanations, we propose a framework that, for any given instance $\mathbf{x}$, attempts to generate new instances for which the model produces explanations that are inconsistent with the explanation produced for $\mathbf{x}$; we refer to the latter as $\mathbf{e}_m(\mathbf{x})$.

We approach the problem in two high-level steps. Given an instance $\mathbf{x}$, *(A)* we create a list of explanations that are inconsistent with the explanation generated by the model on $\mathbf{x}$, and *(B)* given an inconsistent explanation from the list created in A, we find an input that causes the model to generate this precise inconsistent explanation.

Setup. Our setup has three desired properties that make it different from commonly researched adversarial settings in natural language processing:

- At step (B), the model has to generate a *full target sequence*: the goal is to generate the *exact* explanation that was identified at step (A) as inconsistent with the explanation $\mathbf{e}_m(\mathbf{x})$.
- Adversarial inputs do not have to be a paraphrase or a small perturbation of the original input, since our objective is to generate inconsistent explanations rather than incorrect predictions — these can eventually happen as a byproduct.
- Adversarial inputs have to be realistic to the task at hand.

To our knowledge, this work is the first to tackle this problem setting, especially due to the challenging requirement of generating a full target sequence — see Section 4 for comparison with existing works.

Context-dependent inconsistencies. In certain tasks, instances consist of a context (such as an image or a paragraph), and some assessment to be made about the context (such as a question or a hypothesis). Since explanations may refer (sometimes implicitly) to the context, the assessment of whether two explanations are inconsistent may also depend on it. For example, in VQA, the inconsistency of the two explanations "*Because there is a dog in the image.*" and "*Because there is no dog in the image.*" depends on the image. However, if the image is the same, the two explanations are inconsistent regardless of which questions were asked on that image.

For such a reason, given an instance $\mathbf{x}$, we differentiate between parts of the instance that will remain fixed in our method (referred to as *context parts* and denoted as $\mathbf{x}_c$) and parts of the instance that our method will vary in order to obtain inconsistencies (referred to as *variable parts* and denoted as $\mathbf{x}_v$). Hence, $\mathbf{x} = (\mathbf{x}_c, \mathbf{x}_v)$. In our VQA example, $\mathbf{x}_c$ is the image, and $\mathbf{x}_v$ is the question.

Stand-alone inconsistencies. Furthermore, we note that there are cases for which explanations are inconsistent regardless of the input. For example, explanations formed purely of background knowledge such as "*A woman is a person.*" and "*A woman is not a person.*"[1] are always inconsistent (and sometimes outrageous), regardless of the instances that lead to them. For these cases, our method can treat the whole input as variable, i.e., $\mathbf{x}_c = \emptyset$ and $\hat{\mathbf{x}}_v = \mathbf{x}$.

Steps. Our adversarial framework consists of the following steps:

1. Reverse the explanation generator module of model m by training a REVEXPL model to map from the generated explanation and the context part of the input to the variable part of the input, i.e., $\text{REVEXPL}(\mathbf{x}_c, \mathbf{e}_m(\mathbf{x})) = \mathbf{x}_v$.

2. For each explanation $\mathbf{e} = \mathbf{e}_m(\mathbf{x})$:

   (a) Create a list of statements that are inconsistent with $\mathbf{e}$, we call it $\mathcal{I}_\mathbf{e}$.

   (b) Query REVEXPL on each $\hat{\mathbf{e}} \in \mathcal{I}_\mathbf{e}$ and the context $\mathbf{x}_c$. Get the new variable part $\hat{\mathbf{x}}_v = \text{REVEXPL}(\mathbf{x}_c, \hat{\mathbf{e}})$ of a *reverse input* $\hat{\mathbf{x}} = (\mathbf{x}_c, \hat{\mathbf{x}}_v)$, which *may* cause the m to produce inconsistent explanations.

   (c) Query m on each reverse input to get a *reverse explanation* $\mathbf{e}_m(\hat{\mathbf{x}})$.

   (d) Check if each reverse explanation $\mathbf{e}_m(\hat{\mathbf{x}})$ is indeed inconsistent with $\mathbf{e}$ by checking if $\mathbf{e}_m(\hat{\mathbf{x}}) \in \mathcal{I}_\mathbf{e}$.

To execute step (2a), note that explanations are by nature logical sentences. Hence, for any task, one may define a set of logical rules to transform an explanation into an inconsistent counterpart, such as negation or replacement of task-essential tokens with task-specific antonyms. For example, in explanations for self-driving cars (Kim et al., 2018), one can replace "*green light*" with "*red light*", or "*the road is empty*" with "*the road is crowded*" (which are task-specific antonyms), to get inconsistent (and hazardous) explanations such as "*The car accelerates because there is a red light.*".

Another strategy to obtain inconsistent explanations consists of swapping explanations from mutually exclusive labels. For example, assume a recommender system predicts that movie *X* is a bad recommendation for user *Y* "*because X is a horror movie.*", implying that user Y does not like horror movies. If it also predicts that movie *Z* is a good recommendation to the same user *Y* "*because Z is a horror movie.*", then we have an inconsistency, as the latter would imply that user Y likes horror movies.

While this step requires a degree of specific adjustment to the task at hand, we consider it a small price to pay to ensure that the deployed system is coherent. Also, note that this step can eventually be automated, for example, by training a neural network to generate task-specific inconsistencies after crowd-sourcing a dataset of inconsistent explanations for a task at hand — we leave this as future work.

Finally, to execute step (2d), our framework currently checks for an exact string match between a reverse explanation and any of the inconsistent explanations created at step (2a). Alternatively, one can train a model to identify if a pair of explanations forms an inconsistency, which we also leave as future work.

3 Experiments

We consider the task of natural language inference (NLI) (Bowman et al., 2015), which consists of detecting whether a pair of sentences, called *premise* and *hypothesis*, are in a relation of: *entailment*, if the premise entails the hypothesis; *contradiction*, if the premise contradicts the hypothesis; or *neutral*, if neither entailment nor contradiction holds. For example, a pair with premise "*Two doctors perform surgery on patient.*" and hypothesis "*Two doctors are performing surgery on a man.*" constitutes a neutral pair.

The SNLI corpus (Bowman et al., 2015) of $\sim$570K such human-written instances enabled a plethora of works on this task (Rocktäschel et al., 2015; Munkhdalai and Yu, 2016; Liu et al., 2016). Recently, Camburu et al. (2018) augmented SNLI with crowd-sourced free-form explanations of the ground-truth label, called e-SNLI. An explanation

[1] Which was generated by the model in our experiments.

from e-SNLI for the neutral pair above is "*Not every patient is a man.*".

Their best model for generating explanations, called EXPLAINTHENPREDICTATTENTION (hereafter called ETPA), is a sequence-to-sequence attention model that uses two bidirectional LSTM networks (Hochreiter and Schmidhuber, 1997) for encoding the premise and hypothesis, and an LSTM decoder for generating the explanation while separately attending over the tokens of the premise and hypothesis. Subsequently, they predict the label solely based on the explanation via a separately trained network, which maps an explanation to a label.

We show that our framework is able to make ETPA[2] generate a significant number of inconsistent explanations. We highlight that our final goal is not a label attack, even if, for this particular model in which the label is predicted solely from the explanation, we implicitly also have a label attack with high probability.[3]

In our experiments, we set $\mathbf{x}_c$ as the premise (as this represents the given context in this task) and $\mathbf{x}_v$ as the hypothesis. However, note that due to the nature of SNLI for which decisions are based mostly on commonsense knowledge, the explanations are most of the time independent of the premise, such as "*A dog is an animal.*" — hence, it would be possible to also reverse the premise and not just the hypothesis; we leave this as future work.

For the REVEXPL model, we use the same neural architecture and hyperparameters used by Camburu et al. (2018) for ETPA. REVEXPL takes as input a premise-explanation pair, and produce a hypothesis. Our trained REVEXPL model is able to reconstruct *exactly the same* (according to string matching) hypothesis with 32.78% test accuracy.

Creating $\mathcal{I}_\mathbf{e}$. To execute step (2a), we employ negation and swapping explanations. For negation, we simply remove the tokens "not" and "n't" if they are present. If these tokens appear more than once in an explanation, we create multiple inconsistencies by removing only one occurrence at a time. We do not attempt to add negation tokens, as this may result in grammatically incorrect sentences.

For swapping explanations, we note that the explanations in e-SNLI largely follow a set of label-specific templates. This is a natural consequence of the task and the SNLI dataset and not a requirement in the collection of the e-SNLI. For example, annotators often used "*One cannot X and Y simultaneously.*" to explain a contradiction, "*Just because X, doesn't mean Y.*" for neutral, or "*X implies Y.*" for entailment. Since any two labels are mutually exclusive, transforming an explanation from one template to a template of another label should automatically create an inconsistency. For example, for the explanation of the contradiction "*One cannot eat and sleep simultaneously.*", we match *X* to "*eat*" and *Y* to "*sleep*", and create the inconsistent explanation "*Eat implies sleep.*" using the entailment template "*X implies Y.*". Thus, for each label, we created a list of the most used templates that we manually identified among e-SNLI, which can be found in Appendix A. A running example of creating inconsistent explanations by swapping is given in Appendix A.1.

If there is no negation and no template match, we discarded the instance. In our experiments, we only discarded 2.6% of the SNLI test set.

We note that this procedure may result in grammatically or semantically incorrect inconsistent explanations. However, as we will see below, our REVEXPL performed well in generating correct and relevant reverse hypotheses even when its input explanations were not correct. This is not surprising, because REVEXPL has been trained to output ground-truth hypotheses.

The rest of the steps follow as described in (2b) - (2d).

Results and discussion. We identified a total of 1044 pairs of inconsistent explanations starting from the SNLI test set, which contains 9824 instances. First, we noticed that there are, on average, 1.93 ± 1.77 distinct reverse hypotheses giving rise to a pair of inconsistent explanation. Since the hypotheses are distinct, each of these instances is a separate valid adversarial inputs. However, if one is strictly interested in the number of distinct pairs of inconsistent explanations, then, after eliminating duplications, we obtain 540 pairs of such inconsistencies.

Secondly, since the generation of natural language is always best evaluated by humans, we manually annotated 100 random distinct pairs. We found that 82% of the reverse hypotheses form realistic instances together with the premise. We also found that the majority of the unrealistic instances

[2]We use the pretrained model from https://github.com/OanaMariaCamburu/e-SNLI.

[3]Their Explanation-to-Label component had 96.83% test accuracy.

are due to a repetition of a token in the hypothesis. For example, "*A kid is riding a helmet with a helmet on training.*" is a generated reverse hypothesis which is just one token away from a perfectly valid hypothesis.

Given our estimation of 82% to be inconsistencies caused by realistic reverse hypotheses, we obtained a total of $\sim$443 distinct pairs of inconsistent explanations. While this means that our procedure only has a success rate of $\sim 4.51\%$, it is nonetheless alarming that this very simple and under-optimized adversarial framework detects a significant number of inconsistencies on a model trained on $\sim$570K examples. In Table 1, we see three examples of detected inconsistencies. More examples can be found in Appendix B.

Manual scanning. We were curious to what extent one can find inconsistencies via a brute-force manual scanning. We performed three such experiments, with no success. On the contrary, we noticed a good level of robustness against inconsistencies when scanning through the generic adversarial hypotheses introduced by Carmona et al. (2018). The details are in Appendix C.

4 Related Work

An increasing amount of work focuses on providing natural language, free-form explanations (Camburu et al., 2018; Kim et al., 2018; Park et al., 2018; Hendricks et al., 2016) as a more comprehensive and user-friendly alternative to other forms of explainability, such as feature-based explanations (Ribeiro et al., 2016; Lundberg and Lee, 2017). In this work, we bring awareness to the risk of generating inconsistent explanations. Similarly, Hendricks et al. (2017) identify the risk of mentioning attributes from a strong class prior without any evidence being present in the input.

Generating adversarial examples. Generating adversarial examples is an active research area in natural language processing (Zhang et al., 2019; Wang et al., 2019). However, most works build on the requirement that the adversarial input should be a small perturbation of an original input (Belinkov and Bisk, 2017; Hosseini et al., 2017; Cheng et al., 2018), or should be preserving the semantics of the original input (Iyyer et al., 2018). Our setup does not have this requirement, and any pair of task-realistic inputs that causes the model to produce inconsistent explanations suffices. Most importantly, to our knowledge, no previous adversarial attack for sequence-to-sequence models generates *full* target sequences. For instance, Cheng et al. (2018) require the presence of pre-defined tokens anywhere in the target sequence: they only test with up to 3 required tokens, and their success rate dramatically drops from 99% for 1 token to 37% for 3 tokens for the task of summarization. Similarly, Zhao et al. (2018) proposed an adversarial framework for adding and removing tokens in the target sequence for the task of machine translation. Our scenario would require as many tokens as the desired adversarial explanation, and we also additionally need them to be in a given order, thus tackling a much challenging task. Finally, Minervini and Riedel (2018) attempted to find inputs where a model trained on SNLI violates a set of logical constraints. However, their method needs to enumerate and evaluate a potentially very large set of perturbations of the inputs. Besides the computational overhead, it also may easily generating ungrammatical inputs. Moreover, their scenario does not address the question of automatically producing undesired (inconsistent) sequences.

5 Summary and Outlook

We drew attention that models generating natural language explanations are prone to producing inconsistent explanations. This concern is general and can have a large practical impact. For example, users would likely not accept a self-driving car if its explanation module is prone to state that "*The car accelerates because there are people crossing the intersection.*". We introduced a generic framework for identifying such inconsistencies and showed that the best existing model on e-SNLI can generate a significant number of inconsistencies. Future work will focus on developing more advanced procedures for detecting inconsistencies, and on building robust models that do not generate inconsistencies.

Acknowledgments. This work was supported by a JP Morgan PhD Fellowship, the Alan Turing Institute under the EPSRC grant EP/N510129/1, the EPSRC grant EP/R013667/1, the AXA Research Fund, and the EU Horizon 2020 Research and Innovation Programme under the grant 875160.

References

Yonatan Belinkov and Yonatan Bisk. 2017. Synthetic and natural noise both break neural machine translation. *CoRR*, abs/1711.02173.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Srndic, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion attacks against machine learning at test time. In *ECML/PKDD (3)*, volume 8190 of *LNCS*, pages 387–402. Springer.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. *CoRR*, abs/1508.05326.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-SNLI: Natural language inference with natural language explanations. In *NeurIPS*, pages 9560–9572.

Vicente Iván Sánchez Carmona, Jeff Mitchell, and Sebastian Riedel. 2018. Behavior analysis of NLI models: Uncovering the influence of three factors on robustness. In *NAACL-HLT*, pages 1975–1985. Association for Computational Linguistics.

Minhao Cheng, Jinfeng Yi, Huan Zhang, Pin-Yu Chen, and Cho-Jui Hsieh. 2018. Seq2Sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. *CoRR*, abs/1803.01128.

Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. 2016. Generating visual explanations. In *ECCV (4)*, volume 9908 of *LNCS*, pages 3–19. Springer.

Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. 2017. Grounding visual explanations (extended abstract). *CoRR*, abs/1711.06465.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Hossein Hosseini, Baicen Xiao, and Radha Poovendran. 2017. Deceiving Google's cloud video intelligence API built for summarizing videos. In *CVPR Workshops*, pages 1305–1309. IEEE Computer Society.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *CoRR*, abs/1804.06059.

Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John F. Canny, and Zeynep Akata. 2018. Textual explanations for self-driving vehicles. In *ECCV (2)*, volume 11206 of *LNCS*, pages 577–593. Springer.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *CoRR*, abs/1705.04146.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional LSTM model and inner-attention. *CoRR*, abs/1605.09090.

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc.

Pasquale Minervini and Sebastian Riedel. 2018. Adversarially regularising neural NLI models to integrate logical background knowledge. In *CoNLL*, pages 65–74. Association for Computational Linguistics.

Tsendsuren Munkhdalai and Hong Yu. 2016. Neural semantic encoders. *CoRR*, abs/1607.04315.

Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. 2018. Multimodal explanations: Justifying decisions and pointing to the evidence. *CoRR*, abs/1802.08129.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you?": Explaining the predictions of any classifier. In *KDD*, pages 1135–1144. ACM.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomás Kociský, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *CoRR*, abs/1509.06664.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *ICLR (Poster)*.

Wenqi Wang, Benxiao Tang, Run Wang, Lina Wang, and Aoshuang Ye. 2019. A survey on adversarial attacks and defenses in text. *CoRR*, abs/1902.07285.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2019. Adversarial attacks on deep learning models in natural language processing: A survey.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *ICLR (Poster)*. OpenReview.net.

A e-SNLI Explanations Templates

Below we present the list of templates that we manually found to match most of the e-SNLI explanations (Camburu et al., 2018). We recall that during the collection of the dataset Camburu et al. (2018) did not impose any template, they were a natural consequence of the task and SNLI dataset.

Here, "*subphrase1/subphrase2/...*" means that a separate template is to be considered for each of the subphrases. X and Y are the key elements that we want to identify and use in the other templates in order to create inconsistencies. "[...]" is a placeholder for any string, and its value is not relevant. Subphrases placed between round parenthesis (for example, "(the)" or "(if)") are optional, and two distinct templates are formed one with and one without that subphrase.

Entailment Templates

- X is/are a type of Y
- X implies Y
- X is/are the same as Y
- X is a rephrasing of Y
- X is a another form of Y
- X is synonymous with Y
- X and Y are synonyms/synonymous
- X and Y is/are the same thing
- (if) X , then Y
- X so Y
- X must be Y
- X has/have to be Y
- X is/are Y

Neutral Templates

- not all X are Y
- not every X is Y
- just because X does not/n't mean/imply Y
- X is/are not necessarily Y
- X does not/n't have to be Y
- X does not/n't imply/mean Y

Contradiction Templates

- ([...]) cannot/can not/ca n't (be) X and Y at the same time/simultaneously
- ([...]) cannot/can not/ca n't (be) X and at the same time Y
- X is/are not (the) same as Y
- ([...]) is/are either X or Y
- X is/are not Y
- X is/are the opposite of Y
- ([...]) cannot/can not/ca n't (be) X if (is/are) Y
- X is/are different than Y
- X and Y are different ([...])

A.1 Running Example for Creating Inconsistencies by Swapping between Templates of Explanations

Consider the explanation $\mathbf{e}$ ="*Dog is a type of animal.*" which may arise from a model explaining the instance $\mathbf{x}$ = (premise: "*A dog is in the park.*", hypothesis: "*An animal is in the park.*"). We identify that $\mathbf{e}$ matches the template "X is/are a type of Y" with X = "dog" (we convert to lowercase) and Y = "animal". We generate the list $\mathcal{I}_{\mathbf{e}}$ by replacing X and Y in each of the neutral and contradictory templates listed above with the exception of those that contain "[...]" in order to avoid guessing the placeholder. We obtain $\mathcal{I}_{\mathbf{e}}$ as:

- not all dog are animal
- not every dog is animal
- just because dog does not/n't mean/imply animal
- dog is/are not necessarily animal
- dog does not/n't have to be animal
- dog does not/n't imply/mean animal
- cannot/can not/ca n't (be) dog and animal at the same time/simultaneously
- cannot/can not/ca n't (be) dog and at the same time animal
- dog is/are not (the) same as animal

- is/are either dog or animal
- dog is/are not animal
- dog is/are the opposite of animal
- cannot/can not/ca n't (be) dog if (is/are) animal
- dog is/are different than animal
- dog and animal are different

B More Examples of Detected Inconsistencies

In Table 2, we provide more examples of inconsistent explanations detected with our method.

C Manual Scanning

We performed three experiments of manually scanning. First, we manually analyzed the first 50 instances in the test set without finding any inconsistency. However, these examples were involving different concepts, thus decreasing the likelihood of finding inconsistencies. To account for this, in our second experiment, we constructed three groups around the concepts of *woman*, *prisoner*, and *snowboarding*, by simply selecting the explanations in the test set containing these words. We selected these concepts, because our framework detected inconsistencies about them — examples are listed in Table 1 and Table 2.

For *woman*, we obtained 1150 examples in the test set, and we looked at a random sample of 20, among which we did not find any inconsistency. For *snowboarding*, we found 16 examples in the test set and again no inconsistency among them. For *prisoner*, we only found one instance in the test set, so we had no way to find out that the model is inconsistent with respect to this concept simply by scanning the test set.

We only looked at the test set for a fair comparison with our method that was only applied on this set.

However, we highlight that, even if the manual scanning would have been successful, it should not be regarded as a proper baseline, since it does not bring the same benefits as our framework. Indeed, manual scanning requires considerable human effort to look over a large set of explanations in order to find if any two are inconsistent. Even a group of only 50 explanations required us a non-negligible amount of time. Moreover, restricting ourselves to the instances in the original dataset would clearly be less effective than being able to generate new instances from the dataset's distribution. Our framework addresses these issues and directly provides pairs of inconsistent explanations. Nonetheless, we considered this experiment useful for illustrating that the explanation module does not provide inconsistent explanations in a frequent manner.

In our third experiment of manual scanning, we experimented with a few manually created hypotheses from Carmona et al. (2018), which had been shown to induce confusion at the label level. We were pleased to notice a good level of robustness against inconsistencies. For example, for the neutral pair (premise: "*A bird is above water.*", hypothesis: "*A swan is above water.*"), we get the explanation "***Not all birds are a swan.***", while when interchanging bird with swan, i.e., for the pair (premise: "*A swan is above water.*", hypothesis: "*A bird is above water.*"), ETPA generates the explanation "***A swan is a bird.***", showing a good understanding of the relationship between the entities "swan" and "bird". Similarly, interchanging "child" with "toddler" in (premise: "*A small child watches the outside world through a window.*", hypothesis: "*A small toddler watches the outside world through a window.*") does not confuse the model, which outputs "***Not every child is a toddler.***" and "***A toddler is a small child.***", respectively. Further investigation on whether the model can be tricked on concepts where it seems to exhibit robustness, such as *toddler* or *swan*, are left for future work.

PREMISE: Biker riding through the forest.	
ORIGINAL HYPOTHESIS: Man riding motorcycle on highway. PREDICTED LABEL: contradiction ORIGINAL EXPLANATION: **Biker and man are different.**	REVERSE HYPOTHESIS: A man rides his bike through the forest. PREDICTED LABEL: entailment REVERSE EXPLANATION: **A biker is a man.**
PREMISE: A hockey player in helmet.	
ORIGINAL HYPOTHESIS: They are playing hockey PREDICTED LABEL: entailment ORIGINAL EXPLANATION: **A hockey player in helmet is playing hockey.**	REVERSE HYPOTHESIS: A man is playing hockey. PREDICTED LABEL: neutral REVERSE EXPLANATION: **A hockey player in helmet doesn't imply playing hockey.**
PREMISE: A blond woman speaks with a group of young dark-haired female students carrying pieces of paper.	
ORIGINAL HYPOTHESIS: A blond speaks with a group of young dark-haired woman students carrying pieces of paper. PREDICTED LABEL: entailment ORIGINAL EXPLANATION: **A woman is a female.**	REVERSE HYPOTHESIS:The students are all female. PREDICTED LABEL: neutral REVERSE EXPLANATION: **The woman is not necessarily female.**
PREMISE: The sun breaks through the trees as a child rides a swing.	
ORIGINAL HYPOTHESIS: A child rides a swing in the daytime. PREDICTED LABEL: entailment ORIGINAL EXPLANATION: **The sun is in the daytime.**	REVERSE HYPOTHESIS: The sun is in the daytime. PREDICTED LABEL: neutral REVERSE EXPLANATION: **The sun is not necessarily in the daytime.**
PREMISE: A family walking with a soldier.	
ORIGINAL HYPOTHESIS: A group of people strolling. PREDICTED LABEL: entailment ORIGINAL EXPLANATION: **A family is a group of people.**	REVERSE HYPOTHESIS: A group of people walking down a street. PREDICTED LABEL: contradiction REVERSE EXPLANATION: **A family is not a group of people.**

Table 2: More examples of inconsistent explanations detected with our method.