Interactively Generating Counterfactual Examples with AI Assistance

Afra Amini, Frederic Boesel, Robin Chan, Steven H. Wang

June 13, 2022

Abstract

Counterfactual examples (CFs) can be helpful for training and analyzing NLP models, especially in commonsense reasoning tasks like the widely used Natural Language Inference benchmark.

CFs are typically written manually by lay crowdworkers, but recent work has explored machine-in-the-loop approaches where an AI assistant can offer CF suggestions. We contribute an interactive dashboard for lay users writing CFs for Natural Language Inference and integrate AI assistance into the CF generation process. In the same dashboard we also compactly visualize the corpus of user-submitted CFs with a custom text-variant graph, showing how small variations in the CF hypothesis lead to different classification labels. We hope that future work stemming from this dashboard can reduce the labor costs and overall tedium of generating diverse and high-quality CF examples for NLI.

1 Introduction

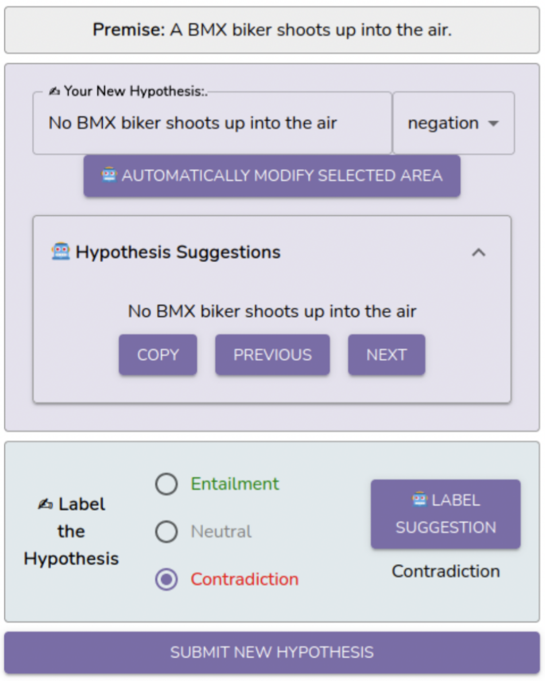
Natural Language Inference (NLI) is a popular commonsense reasoning task, where given a premise and a hypothesis sentence as text input, a classifier decides whether the premise implies the hypothesis ("entailment"), contradicts the hypothesis ("contradiction"), or has no bearing on the hypothesis ("neutral") [1].

A very commonly used dataset to train models in an NLI task is the Stanford NLI dataset [1], originally consisting of 550'152 manually labeled training sentence pairs, with an additional 10'000 used for training and testing each. A subsampled, manually-labeled augmented dataset has been published containing 6'600 revised samples combined with the 2'700 original sentences [2]. Using the same set of 2'700 original sentences, we aim to generate a complimentary set of revised samples generated in our user-interface.

Spurious Associations and Counterfactual Examples in NLI Previous work has shown that due to annotation artifacts (such as neutral hypotheses being more likely to clauses with the word "because"), models trained on only hypotheses perform surprisingly well, even though inference in this case should be equivalent to random chance [3, 4]. This is because annotation artifacts allow the NLI models to rely on spurious associations with words (like "because") that are unrelated to the general inference task itself, thus hurting model performance and generalization. Indeed, it has been shown that adding new CF examples to the test set reduces NLI performance in a variety of models, and that in some cases introducing these CF examples to the training set can even improve performance on the original dataset without CF examples [5].

AI Assistance for NLI Counterfactual Generation Polyjuice [6] is a fine-tuned GPT-2 [7] model that automatically perturbs an NLI example to generate suggested counterfactuals. Polyjuice adds ten new special tokens to GPT-2. One of these special tokens is `[BLANK]` which can be used to mark particular regions of an NLI hypothesis for perturbation. Other special tokens, called "control codes", are used to specify particular types of perturbations, like `[NEGATION]` and `[RESTRUCTURE]`.

Though the Polyjuice setup still requires user-input in the form of selecting fluent sentences and labeling the revised sentences, adding Polyjuice assistance to the CF generation process enables a higher sentence throughput than the entirely manual process in [5].


Premise: A BMX biker shoots up into the air.

Your New Hypothesis: No BMX biker shoots up into the air

negation

Hypothesis Suggestions

No BMX biker shoots up into the air

Label the Hypothesis: Entailment [✓ Contradiction] Neutral

LABEL SUGGESTION: Contradiction

Figure 1: Interface for creating and submitting a new counterfactual. The three buttons marked with robot emojis provide different types of AI assistance to the lay user.

Our Contribution In [6], the user interacts with Polyjuice only indirectly through labelling precomputed CF suggestions from Polyjuice. In our work, in addition to providing precomputed Polyjuice suggestions, we also create an interactive interface that enables the lay user to directly interface with Polyjuice by choosing the subspan of the hypothesis for Polyjuice to perturb and selecting the type of counterfactual modification. We also show the user AI classifier predictions from `roberta-large-mnli`, a Transformer-based NLI classifier [8].

Furthermore, we provide in our interface two visualizations of the counterfactuals that have been submitted so far: (1) a custom-coded Text Variant Graph [9] which compactly shows how small changes to the CF hypothesis lead to changes in the NLI label and (2) a simple Table View of the CF hypotheses which allows the user to copy and delete hypotheses.

2 User Interface

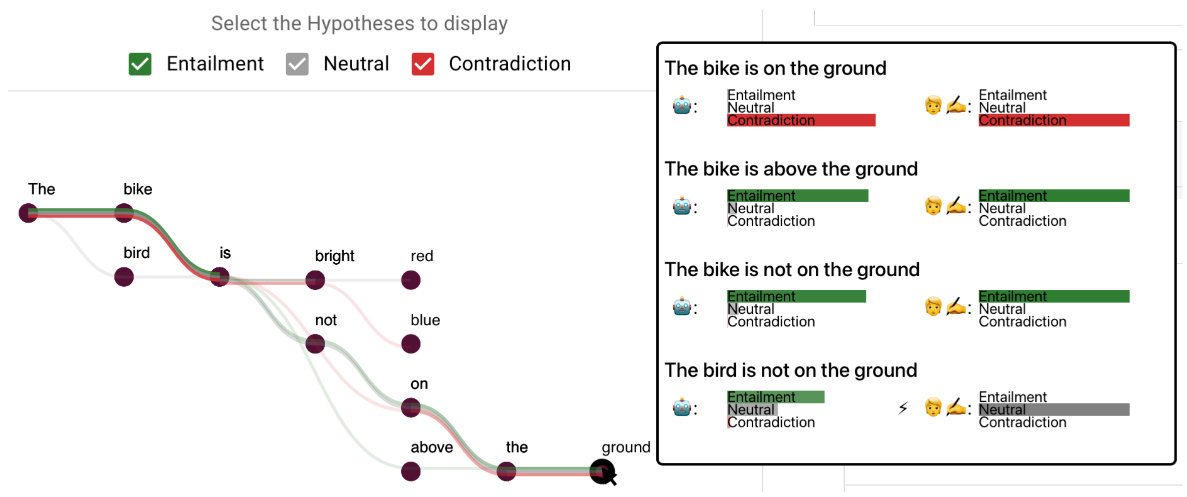
Dashboard Overview Our dashboard is divided into a top and bottom half. The top half contains components for switching between sentence pairs (premise and hypothesis) from the original SNLI dataset (see Figure 1). The bottom half provides two visualizations of all the hypotheses submitted so far for the currently selected sentence pair. The first visualization is a custom Text Variant Graph (see Figure 2) with edge colors indicating the user label (entailment, neutral, contradiction). The second visualization is a basic table view of the submitted hypotheses that provides copy and delete functionality.

See Section B in the appendix for some visualizations that we scrapped.

Guided Tour Upon loading the front-end, the user is presented with a series of pop-up windows that guide the user through the labelling process and that explain the usage of each component.

2.1 Top Half – Sentence Pair Selection and AI-Assisted CF Generation

Sentence Pair Selection Component The top-left component has left and right buttons that allow the user to select a sentence pair (premise and hypothesis) from the original dataset, and to see the original label of the sentence pair. The selected sentence pair changes the subset of the NLI dataset



The bike is on the ground
🤖: Entailment Neutral Contradiction
🧑✍: Entailment Neutral Contradiction

The bike is above the ground
🤖: Entailment Neutral Contradiction
🧑✍: Entailment Neutral Contradiction

The bike is not on the ground
🤖: Entailment Neutral Contradiction
🧑✍: Entailment Neutral Contradiction

The bird is not on the ground
🤖: Entailment Neutral Contradiction
⚡ 🧑✍: Entailment Neutral Contradiction

Figure 2: Text Variant Graph visualization of submitted CF examples for the original NLI premise and hypothesis pair – **Premise:** "A BMX biker shoots up into the air." **Hypothesis:** "The bike is bright red." Three colored checkboxes at the top of the display allow the user to filter hypotheses by their user label. A large tooltip window appears when the user hovers over the node "ground" and displays all hypotheses that pass through the "ground" node, along with the respective user labels and `roberta-large-mnli` probability distributions. The human annotator managed to fool the model on the last "The bird is not on the ground" CF example, possibly due to the model relying on the "not on the ground" part of the sentence.

that is visualized in the Graph and Table views and which is perturbed in the Polyjuice-Assisted CF Submission Componeent.

Polyjuice-Assisted CF Submission Component The CF Submission Component (Figure 1) allows the user to submit new CF hypotheses with three types of AI assistance. (1) The "Automatically Modify Selected Area" button and a dropdown menu of Polyjuice control codes allows the user to directly perturb a highlighted portion of the hypothesis with an instance of Polyjuice running the application backend. (2) The collapsable Hypothesis Suggestion window contains randomly generated cached Polyjuice hypothesis suggestions that the user can copy to the clipboard. (3) The Label Suggestion button allows the user to view the label predicted by a `roberta-large-mnli` model. The most valuable CFs are those where the user and the model disagree, and seeing the AI label can teach the user adversarial patterns that trick the AI.

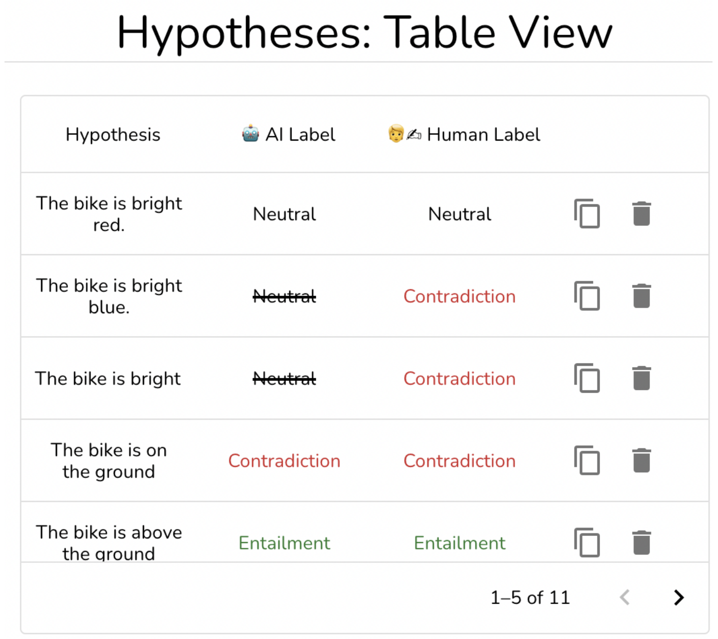
Users write hypotheses and select portions of text to perturb in the "Your New Hypothesis" input box. To ease the Polyjuice text selection process, text selections automatically snap to word boundaries. After labelling the hypothesis, the user submits the hypothesis via the "Submit New Hypothesis" button at the bottom of this component.

2.2 Bottom Half – Graph and Table Visualizations of CFs

Interactive Variance Graph Visualization The user is able to get a compact of all created CFs for the currently displayed sentence pair via our custom-coded Variance Graph (Figure 2), inspired by [9]. Our application backend joins all the submitted hypotheses in nodes and edges using `collateX` [10] for text sequence alignment. In the frontend, we use D3 to render an SVG graph is displayed with each node representing a word and a continuous line through the nodes for each sentence. Zoom and panning functionality allows the user to focus on particular nodes and edges and allows the display to accommodate large numbers of counterfactual examples.

The edges in the graph are color-coded by the user label (green, grey, and red for entailment, neutral, and contradiction labels) and vary in opacity based on the number of hypotheses passing through that link words in the corpus.

When the cursor hovers over a node or link, a large tooltip appears, displaying a list of all hypotheses that pass through the node or link along with with the user label and a horizontal bar plot of the



Hypotheses: Table View

Hypothesis	AI Label	Human Label
The bike is bright red.	Neutral	Neutral
The bike is bright blue.	~~Neutral~~	Contradiction
The bike is bright	~~Neutral~~	Contradiction
The bike is on the ground	Contradiction	Contradiction
The bike is above the ground	Entailment	Entailment

1–5 of 11

Figure 3: Table view of submitted counterfactual hypotheses. Delete and copy buttons allow the user to remove erroneous CFs and to make a new CF starting from a previously submitted hypothesis. When the user disagrees with the AI label, the AI label is strikedthrough.

`roberta-large-mnli` probability distribution. This enables the user to explore the decision boundary of the model by doing a what-if analysis. The user could add an additional CF with a small change and observe how the output probability distribution from RoBERTa changes. A lightning symbol is displayed in the hover tooltip next to under CF hypotheses where the user and AI model's labels contradict each other, encouraging the user to find such valuable examples. The Variance Graph aims at helping the user in creating more diverse CFs, by providing an overview of the patterns how the existing CFs relate the the original hypothesis. This can help the user break these patterns or explore new possibilities of changing the original hypothesis.

The visualization gives the lay user increased understanding of the underlying classification model because the colored edges in the graph show how small perturbations to the sentence change the model's classification, and the hover tooltip shows which sentences the model is more or less certain about.

Interactive Table Visualization The final component of the dashboard is a Table Visualization of the submitted counterfactuals (Figure 3 to the graph the user can also take a look at existing CFs in a table. The user can interact with the table by copying existing CFs into the submission box or by deleting erroneous CFs.

Originally, we wanted to add the copy and delete functionality to the graph visualization. However, there didn't seem to be an easy way to select a particular hypothesis for copy or deletion when multiple sentences traversed through each node and edges, so we chose to create a separate table view.

3 Discussion and Outlook

In prior work, the user interacted with Polyjuice only indirectly through labelling precomputed CF suggestions from Polyjuice. We expand on previous work by also creating an interactive interface that enables the lay user to directly interface with Polyjuice by choosing the subspan of the hypothesis for Polyjuice to perturb and selecting the type of counterfactual modification, and a graph visualization that increases lay user understanding of the underlying classification model.

In a next step, evaluation is necessary to assess the quality of the generated counterfactuals. The dataset itself then needs to be assessed on diversity and closeness, for which the self-BLEU score

[11] and the Levenshtein edit distance are commonly used. Benchmark evaluation metrics for the finetuned model could be taken from the Polyjuice paper, where in-domain test set accuracy, as well as generalizability on out-of-domain datasets, contrast sets and challenge sets are assessed.

Regarding human-trust-modeling, there is further work to be done when moving towards evaluation. Since each labeler generates their own unique set of counterfactuals, we can't apply majority voting to determine the correct label or use a standard SNLI model to assess label quality, as it doesn't make sense for this dataset, since the most useful counterfactuals lie near the decision boundary or are classified wrongly by a such a model. Most effective would be to manually label some cached polyjuice suggestions and test labelers on those, as they are most likely to be chosen.

We are also interested in creating GPT-3 prompts and queries for generating counterfactuals via OpenAI API, because we were surprised to find that many Polyjuice suggestions were of dubious quality despite the paper's dozens of citations.

References

[1] S. R. Bowman, G. Angeli, C. Potts, C. D. Manning, A large annotated corpus for learning natural language inference, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 632–642. doi:10.18653/v1/D15-1075.
URL https://aclanthology.org/D15-1075

[2] D. Kaushik, A. Setlur, E. Hovy, Z. C. Lipton, Explaining the efficacy of counterfactually augmented data, International Conference on Learning Representations (ICLR) (2021).

[3] A. Poliak, J. Naradowsky, A. Haldar, R. Rudinger, B. Van Durme, Hypothesis only baselines in natural language inference, in: Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 180–191. doi:10.18653/v1/S18-2023.
URL https://aclanthology.org/S18-2023

[4] S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. Bowman, N. A. Smith, Annotation artifacts in natural language inference data, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 107–112. doi:10.18653/v1/N18-2017.
URL https://aclanthology.org/N18-2017

[5] D. Kaushik, E. Hovy, Z. C. Lipton, Learning the difference that makes a difference with counterfactually-augmented data (2019). doi:10.48550/ARXIV.1909.12434.
URL https://arxiv.org/abs/1909.12434

[6] T. Wu, M. T. Ribeiro, J. Heer, D. S. Weld, Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models (2021). doi:10.48550/ARXIV.2101.00288.
URL https://arxiv.org/abs/2101.00288

[7] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners (2019).

[8] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692 (2019).

[9] S. Jänicke, B. Marco, G. Scheuermann, Text variant graph: Improving the layout for text variant graphs, VisLR: Visualization as Added Value in the Development, Use and Evaluation of Language Resources (2014).

[10] R. Dekker, G. Middell, Computer-supported collation with collatex. managing textual variance in an environment with varying requirements, in: Supporting Digital Humanities 2011, 2011.
URL http://hnk.ffzg.hr/bibl/SDH-2011/submissions/sdh2011_submission_54.pdf

[11] Y. Zhu, S. Lu, L. Zheng, J. Guo, W. Zhang, J. Wang, Y. Yu, Texygen: A benchmarking platform for text generation models (2018). doi:10.48550/ARXIV.1802.01886.
URL https://arxiv.org/abs/1802.01886

A Contribution Statement

A.1 Afra Amini

- Refactoring UI and rewriting all the components with material-ui
- Creating cached suggestions for polyjuice
- Implementing span selection interactive feature for polyjuice
- Contributions to creating the poster

A.2 Frederic Boesel

- Implementation and exploration of different variants of the Text Variant Graph
- Implementation of the graph creation from the submitted CFs for the variant graph
- Implementation of the handling of the submitted CFs

A.3 Robin Chan

- Contributions to refactoring the UI and interaction-workflow.
- Creating a guided tour through the interface.
- Implementing roBERTa label suggestions.

A.4 Steven H. Wang

- Dockerization.
- Contributions to text variants graph.
- Table view.
- Preliminary scripts for fine-tuning MNLI models on augmented dataset, OpenAI queries for improved models.
- Scrapped UMAP embedding visualization.
- Contributions to creating the poster

B Scrapped Solutions

B.1 UMAP embeddings

We attempted a UMAP visualization (Figure 4) of `roberta-large-mnli` activations as an explainability tool for suggested AI labels, but scrapped this in favor of the Text Variant Graph, which we believed was a more useful tool for lay users.

B.2 An even more custom text variant graph

We began prototyping a text variant graph based on D3 Force and a vague ”principle of minimum energy” idea, but scrapped that in favor of a static graph. The idea behind this graph was that entailment links and nodes would float to the top and the contradiction links and nodes would sink to the bottom, allowing for more effective positional encodings.

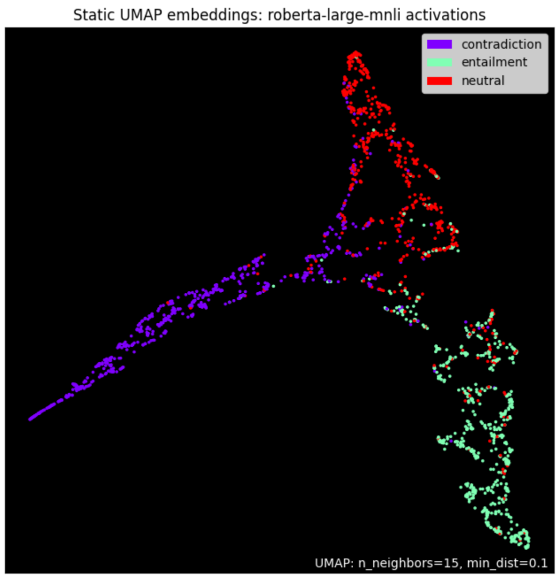


Figure 4: UMAP embedding visualization attempt.