Towards Stronger Adversarial Baselines Through Human-AI Collaboration

Wencong You and **Daniel Lowd**
University of Oregon
Eugene, OR
{wyou, lowd}@cs.uoregon.edu

Abstract

Natural language processing (NLP) systems are often used for adversarial tasks such as detecting spam, abuse, hate speech, and fake news. Properly evaluating such systems requires dynamic evaluation that searches for weaknesses in the model, rather than a static test set. Prior work has evaluated such models on both manually and automatically generated examples, but both approaches have limitations: manually constructed examples are time-consuming to create and are limited by the imagination and intuition of the creators, while automatically constructed examples are often ungrammatical or labeled inconsistently. We propose to combine human and AI expertise in generating adversarial examples, benefiting from humans' expertise in language and automated attacks' ability to probe the target system more quickly and thoroughly. We present a system that facilitates attack construction, combining human judgment with automated attacks to create better attacks more efficiently. Preliminary results from our own experimentation suggest that human-AI hybrid attacks are more effective than either human-only or AI-only attacks. A complete user study to validate these hypotheses is still pending.

1 Introduction

Humans have used language to deceive each other for millennia. With the advent of NLP systems, humans now work to deceive models and algorithms, from evading email spam filters in the early 2000s to defeating classifiers for social network spam, abusive language, misinformation, and more. More recently, humans have developed automated adversarial attacks that minimally modify text while changing the output of a classifier or other NLP systems (Ebrahimi et al., 2018). These automated attacks have the potential to be much more efficient than humans, helping attackers to find weaknesses in models and helping defenders find and patch those same weaknesses (Xie et al., 2021; Zhou et al., 2019).

Attack	Original → Perturbed Text	Label
PSO	city by the sea swings from one approach to the other , but in the end , it stays in formula – which is a [waste → moor] of de niro , mcdormand and the other good actors in the cast .	Neg. (98%) → Pos. (93%)
BAE	When a set of pre-shooting guidelines a director came up with for his actors turns out to be cleverer , better written and of considerable more interest than the finished film , that 's a [bad → good] sign .	Neg. (97%) → Pos. (95%)
PWWS	[A refreshing → axerophthol review] Korean film about five [female → distaff] high school friends who face an uphill battle when they try to take their relationships into deeper waters.	Pos. (99%) → Neg. (73%)

Table 1: Attack Samples on SST-2

The number of automated attacks continues to grow but their effectiveness remains low — Wang et al. (2021a) found that 90% of automated adversarial attacks changed the semantics of the original input or confused human annotators. We have observed similar behavior, as shown in Table 1. These examples are generated by word-level attack algorithms PSO (Zang et al., 2020), BAE (Garg and Ramakrishnan, 2020), and PWWS (Ren et al., 2019), as implemented in the TextAttack framework (Morris et al., 2020), on the sentiment dataset SST-2 (Socher et al., 2013) against BERT model (Devlin et al., 2019). Although all perturbations change the predicted label, PSO chooses a synonym that is inappropriate in the context, BAE selects a complete antonym, and PWWS picks some rare substitutes that are nonsensical and possibly offensive.

Doubtless, humans can be more effective than these attacks, given their effectiveness against real-world spam and abuse filters. We believe that the next step for adversarial attacks and robust NLP is human-AI collaboration, in which humans work with automated adversarial algorithms to pro-

Proceedings of NLP Power! The First Workshop on Efficient Benchmarking in NLP, pages 11 - 21

duce effective attacks efficiently. Furthermore, real-world attackers are already doing this. Spammers already use many different technologies to accomplish their tasks, including text spinners to rewrite text, HTML tricks to conceal suspicious text, botnets to scale up and avoid IP bans, and more. A typical spammer does not craft every message individually, but uses semi-automated techniques to generate many different messages[1]. In response, a growing amount of NLP research is now using human expertise through human-in-the-loop (HITL) methods to create new benchmarking datasets for evaluating and improving the robustness of NLP systems to adversarial inputs.

Thus far, human expertise in adversarial NLP tasks has been limited. There is a growing body of work in which humans are asked to craft inputs where a given model will perform poorly, but they receive little support in doing so — sometimes word saliences (Mozes et al., 2021), sometimes model predictions (Kiela et al., 2021), and sometimes even less. Overall, the effort between humans and machines is still largely separate; that is, humans generate adversarial examples alone based on model interpretations, without directly interacting with any attack algorithms.

In this paper, we study the potential of direct human-AI interaction for generating higher-quality adversarial examples for NLP tasks. We work with the state-of-the-art word-level attacks on benchmark datasets for sentiment analysis and abuse detection. We choose word-level attacks as they can be more subtle than character-level attacks, which have obvious misspellings. We design an interactive user interface that enables four types of attacks, including two human-AI collaboration methods. Instead of a pure black-box environment, our interface explains the algorithm's search space and allows humans to modify and improve the perturbations while giving humans immediate feedback from the target NLP model. Along with generated attacks, we collect data for user experience and user preference with regard to different attack approaches. We then further study the collected data and analyze the impact of proposed human-AI collaboration methods and the degree of improvement on the adversarial examples. At present, we have pilot data from using the system ourselves; a full user study is pending IRB approval.

[1]For an example of a spammer script that does this, see https://alexking.org/blog/2013/12/22/spam-comment-generator-script.

We summarize our contributions as follows:

- We propose a novel human-AI collaboration strategy to enable direct human and AI interaction for generating word-level adversarial examples for NLP tasks effectively and efficiently.
- We design a framework with friendly user interface to realize four types of attack methods on benchmark datasets against state-of-the-art NLP models. In addition to helping generate adversarial examples, the framework also collects self- and peer-evaluation of example quality and user feedback about the interface.
- We share initial results based on our own use of the system, while IRB approval for a full study is pending.

The rest of the paper is structured as follows: Section 2 discusses work related to our research. Section 3 introduces our framework, the human-AI collaboration methods and the evaluation metrics. Section 4 gives preliminary results and some brief analysis for our findings. Section 5 explains the stages of experiments for generating and collecting quality data. Finally, we conclude and discuss future work in Section 6.

2 Related Work

We review prior work on automated adversarial attacks for NLP, and HITL in adversarial learning.

Automated adversarial attacks for NLP: With the growth of research that studies adversarial learning in NLP, a variety of attack methods have been developed on multiple levels. From character-level modifications such as HotFlip (Ebrahimi et al., 2018), DeepWordBug (Gao et al., 2018), and VIPER (Eger et al., 2019), to word-level perturbations such as BAE (Garg and Ramakrishnan, 2020), PSO (Zang et al., 2020), PWWS (Ren et al., 2019), and TextFooler (Jin et al., 2020). Many of them have been aggregated and organized by toolchains like TextAttack (Morris et al., 2020) and OpenAttack (Zeng et al., 2021) for easy access to researchers.

For character-level attacks, although they show their effectiveness in many ways, they mainly fall in the following two categories: Some of the character-level modifications can be seen as typos if an algorithm simply influences the embedding space by replacing/inserting/deleting one or a few

characters in a word, such as DeepWordBug (Gao et al., 2018), then they may be easily detected by a grammar checker tool, like Grammarly [2]; the others can introduce some unique encoding/decoding methods and transform letters to another form, such as VIPER (Eger et al., 2019) that adds accent signs on top of each letter, and these modification may be easily identified by human. Overall, character-level perturbations tend to be more obvious.

On the other hand, the study of word-level attacks is more popular, as a substitute for a word may significantly impact the semantics of the text. Many attack methodologies have been investigated for searching for the optimal synonym substitutions, including BERT-based contextual prediction (Garg and Ramakrishnan, 2020; Li et al., 2020), gradient-based word swap (Ebrahimi et al., 2018; Wallace et al., 2019), particle swarm optimization (Zang et al., 2020), and greedy word search with saliency scores (Ren et al., 2019).

We summarize three attacks that are included in our framework. **BAE:** BERT-based Adversarial Examples (BAE), a black-box contextual perturbation algorithm based on a BERT masked language model (MLM). BAE masks some part of the text, then replaces and inserts tokens into the text, using the BERT-MLM to generate adversarial examples. **PWWS:** Probability Weighted Word Saliency (PWWS), a black-box greedy algorithm that ranks the importance of words based on the saliency score and calculates the classification probability that are used to determine the synonym substitution. **TextFooler:** TextFooler, a black-box greedy algorithm identifies the important words and replaces them with the words that are most semantically similar and grammatically correct with a higher priority until the prediction is altered.

These automated word-level attacks mostly rely on the knowledge of existing target models and algorithms' intensive search to locate the best synonym substitutions. However, recent work (Xie et al., 2021, 2022) shows that the quality of generated adversarial examples is actually far from satisfactory, with respect to the low attack success rate across domains, incorrect grammar, and distorted meaning.

HITL in adversarial learning: As the capacity of automated algorithms may be limited, many researchers propose incorporating crowd-sourcing into generating and annotating adversarial examples. The Dynabench framework asks humans to manually construct examples where an NLP system would perform poorly (Kiela et al., 2021). A HITL QA system that asks humans to write adversarial questions that break a QA system while remaining answerable by humans (Wallace and Boyd-Graber, 2018). The Adversarial NLI project asks humans to annotate mislabeled data and uses humans as adversaries to create a benchmark natural language inference (NLI) dataset for a more robust NLP model (Nie et al., 2020). The most related work compares the performance of human- and machine-generated word-level adversarial examples for NLP classification tasks (Mozes et al., 2021).

However, existing work falls short of direct collaboration between humans and AI. The advantages of human crowd-sourcing and that of automated algorithms are still quite distinct.

3 Framework

In our framework, we study the potential of direct human-AI collaboration for generating higher-quality adversarial examples. At the time of submission, we have completed the design of the framework, confirmed the details for human-AI collaboration, and implemented the interactive user interface.

3.1 Components & Workflow

Our task is divided into two parts: *generating adversarial examples* and *evaluating adversarial examples*. Figure 1 depicts the workflow. First we feed the input samples to the attack phase where four attack methods are implemented. Human participants then use these attack methods to generate adversarial examples aiming to fool the target model's predictions. Participants are asked to self-evaluate the quality of generated adversarial examples based on grammatical properties, the difficulty of generating those examples, and their experiences with the system in terms of the helpfulness of different HITL strategies. Peer-evaluation is also included for evaluating the grammatical properties, and identifying the source of any given text.

We implement three word-level attacks — BAE, PWWS, and TextFooler from the TextAttack library on sentiment dataset SST-2 and abuse comment dataset Hatebase (Davidson et al., 2017) against the RoBERTa target models (Liu et al., 2019) that are trained on these datasets separately. We use RoBERTa as the target model because it outper-

[2]Grammarly, https://www.grammarly.com/.

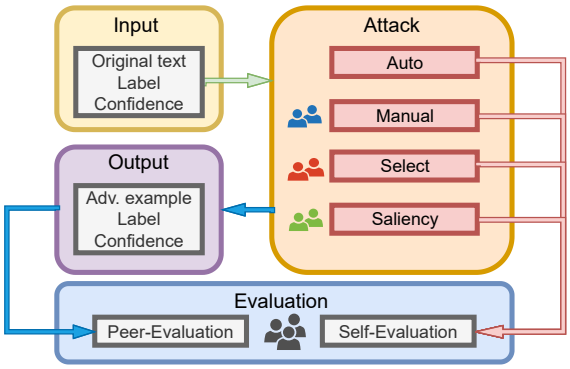

Figure 1: System & Workflow. Human figures in attack phase indicate that there is direct human-AI interaction. Human figures in evaluation phase indicate that humans are involved in both self-evaluation and peer-evaluation.

Attack	Transformation	Operation
BAE	BERT Masked Token Prediction	Replace & Insert
PWWS	WordNet-based synonym swap	Replace
TF	Counter-fitted word embedding swap	Replace

Table 2: A Summary of automated attack algorithms. TF is short for TextFooler.

forms BERT (Devlin et al., 2019) and XLNet (Yang et al., 2019) on various datasets across domains for classification in recent work (Xie et al., 2022). We summarize the characters of these attacks in Table 2. Please refer to Section 2 for a detailed description of them. All attacks share the same Greedy-WIR search method implemented in TextAttack. We make certain modifications to the scripts in the TextAttack library to generate desired intermediate attack results, which are used as interpretable information for HITL adversarial attacks.

3.2 Generating Adversarial Examples

For attack generation, we design an interactive user interface introducing four attack methods:

- **Auto:** Black-box. Participants simply read and evaluate adversarial examples generated by one of the automated attack algorithms. Participants are not provided with any insight on how an automated attack algorithm modifies a sample, but the perturbed example itself. This method is considered as the baseline.
- **Manual:** Black-box. Participants rely on their judgment solely to attack a given sample. The only information they receive is the immediate target model prediction. Once an adversarial example is entered, the target model returns the prediction result to show whether or not the crafted example has successfully flipped the predictive label.
- **Select:** Gray-box. Participants are given intermediate perturbation results from the automated algorithm — specifically, keywords and potential substitution candidates for each keyword. Participants can select the best word substitute using dropdown lists, or enter an alternative word in a text input box. See Figure 5 for the interface. Basically, the Select method relaxes the constraints from the automated algorithm, and allows humans to modify up to five keywords. The immediate predictive label and probability of the selected word combination from the target model is also provided to show whether the chosen words have successfully changed the prediction.
- **Saliency:** Gray-box. Participants are shown a dynamic saliency map as they craft their adversarial examples. A saliency map shows what words the target model identifies as most important that are most likely to affect the prediction, and then marks those words with colors with different intensities. Unlike (Mozes et al., 2021), where the interface displays word saliencies calculated by replacing the word with an out-of-vocabulary token, we implement the built-in method in each automated attack to calculate the saliency score. For example, BAE and TextFooler simply delete the word and calculate the word saliencies, while PWWS replaces each word with an unknown token and calculates the weighted saliency. The corresponding mathematical expressions are provided in A.2 of the Appendix. Overall, the Saliency method grants even more flexibility by allowing humans to change more words if necessary in order to preserve correct grammar and semantics. Participants can adjust their perturbation based on the dynamic saliency map and the target model's immediate prediction, see Figure 6 for the interface.

For each method, participants are given a small number of original samples selected from one of the datasets, perform adversarial attacks on those samples with or without the assistance of the automated algorithms.

3.3 Evaluating Adversarial Examples

To evaluate generated adversarial examples, we consider the following properties:

- **Grammar:** measures whether or not the text contains any syntax errors, and retains the original or similar semantics. This is crucial for identifying if an adversarial attack is successful, as if the perturbation is fundamentally wrong by making the sentence unreadable or flipping the emotion of the message completely, we consider it as a failed attack.
- **Plausibility:** measures whether or not the text is naturally crafted by native speakers. A piece of text is highly plausible if it is natural, logically correct, appropriately worded, and preserving meaningful messages (Wang et al., 2021b). These properties appear as naturalness, correctness, appropriateness and meaningfulness in our user interface.
- **Effort:** reflects the difficulty level for participants to successfully perform adversarial attacks using different attack methods.
- **Helpfulness:** collects the degree of helpfulness of the information provided to participants to assist with generating adversarial examples in different attack methods (i.e., intermediate search results, lists of candidates, saliency maps, and more).

All properties are evaluated on a scale from 1 to 5 where 5 indicates the best quality, the most difficult, or the most helpful, depending on the specific property; see Figure 7.

Participants are required to self-evaluate their own constructed examples using each of the attack methods. Since self-evaluation can be very subjective, to ensure the fairness and to yield a more balanced and less biased analysis and outcome, we also plan to include anonymous peer-evaluation using Amazon Mechanical Turk (AMT) [3] with a group of AMT workers who are excluded from previous attack tasks. Each AMT worker reads a random subset of the adversarial examples, identifies what source an example may come from, and evaluates the grammatical quality (i.e. grammar and plausibility) of that example on the same scales.

[3] Amazon Mechanical Turk, see https://www.mturk.com/

4 Preliminary Results

Our hypotheses are that with minimal human collaboration, compared to automated attacks alone, the attacks would yield more promising results that are meaningful while holding correct grammar and semantics. In our preliminary work, we already see promise for this direction. Table 3 shows an example where PWWS on its own failed to come up with a good attack example, but succeeded in identifying the key text to modify. A human was then able to propose alternative text, which tricked the classifier while maintaining the correct semantics.

OR. Txt	Auto Txt	HITL Txt
4 friends , 2 couples , 2000 miles , and all the Pabst Blue Ribbon beer they can drink - it 's the ultimate road-trip . (Pos. 62%)	4 friends , 2 couples , 2000 miles , and all the Pabst disconsolate Ribbon beer they can drink - it 's the ultimate road-trip . (Neg. 84%)	4 friends , 2 couples , 2000 miles , and all the Pabst cheap beer they can drink - it 's the ultimate road-trip . (Neg. 83%)

Table 3: Original vs. automated attack vs. HITL attack

As a pilot experiment, to test the viability of the framework before recruiting participants, the authors used the framework on themselves to collect 532 unique adversarial examples generated from the SST-2 dataset. By studying these examples, we have seen the following patterns (which we hypothesize will extend to the full experiments):

Success Rate: Figure 2 shows the attack success rate across all attack methods. Though an automated attack may have a higher attack success rate due to the advantage of intensive search and the NLP model-oriented design, humans can achieve comparable attack success rate if provided with better human-AI interaction. Additionally, manually crafted attacks without any assist cannot compete with the those generated through other methods.

Grammar and Plausibility: Figure 3 presents the average scores for grammar and plausibility, where the error bars denote the standard errors of the scores. The scores are aggregated and averaged per the attack method from the self-evaluation results over the 532 adversarial examples. It is obvious that human-generated adversarial examples on average have higher scores considering the grammatical properties and plausibility. Manual attack and HITL methods seem to produce higher-quality adversarial examples with the assistance of automated algorithms, as compared to automated

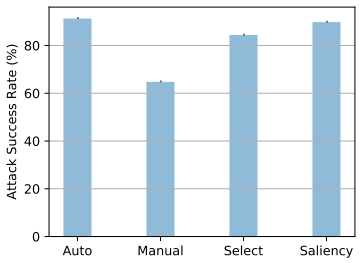

Figure 2: Attack success rate

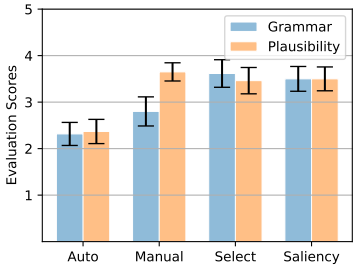

Figure 3: Grammar & plausibility

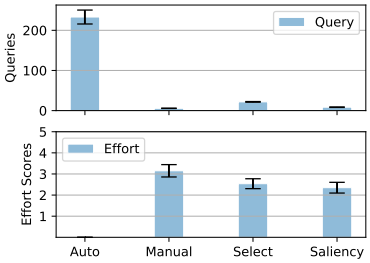

Figure 4: Queries & Effort

attacks, these methods loosen the constraints on various degrees and grant humans more freedom to make more modifications if needed. Therefore humans have more flexibility crafting grammatically correct and plausible adversarial examples.

Queries and Human Effort: The top of Figure 4 displays the number of queries it takes for an automated algorithm or a human to choose their word substitutions. The bottom of the figure gives the average effort scores for each attack method. The error bars denote the standard errors of the scores. The results illustrate that humans are able to perturb an NLP model with more effort but fewer queries, and the gray-box setting, which includes additional information for the participants, is easier to attack than the black-box settings. The extra information provides some insight and explanation about how an automate algorithm understands the NLP model and how an NLP model decides the predictions.

5 Planned Experiments

We plan to hire approximately 54 adult native English speakers, of whom we expect a subset to be experts in NLP or linguistics, from our local university to generate adversarial examples, and additional adult native English speaker AMT workers for peer-evaluation.

Unlike the recent work of Mozes et al. (2021), which relies entirely on online crowd-sourcing on AMT, we carry on in-person experiments for attack generation, where we provide a few examples and detailed instructions to the participants to show how our interface operates, and what the standards/baselines are for evaluating the adversarial examples. We expect to obtain higher-quality data by bringing participants into a more controlled environment where it's easier to provide instruction, answer questions, and receive feedback.

To motivate participants through the process, we have designed an incentive payment plan. Details are included in A.3 of the Appendix.

Stage 1: adversarial example generation and self-evaluation. In each task, each participant is asked to work with approximately 15 examples from a source dataset, generating adversarial examples based on the source examples. We show the same examples to three different participants, who work independently to find their own adversarial examples. This gives us a chance to observe how varied the solutions are; if solutions vary substantially, then a larger group of people may have a better chance to find a good attack.

To increase the quality of the adversarial examples, we plan to have each participant complete the Auto and Manual methods before moving on to our proposed HITL methods. This also serves the purpose of training participants in these tasks, similar to tasks 1-3 by Mozes et al. (2021). By doing so, participants have the chance to get familiar with our user interface, and get a better understanding of the capacity of an automated attack algorithm versus a human, in terms of influencing the target model's predictions. They then closely interact with the automated algorithms and the target model, where they obtain extra interpretable information from both parties that could assist them with more effective perturbations.

To increase the independence of the factors that may potentially impact the experiment results statistically, such as the order of samples and attack tasks being presented to an participant, we mix up the order of samples in each attack method, and we switch the order of attack methods before giving them to the participants.

Each participant at our local university is expected to submit about 45 adversarial examples if they successfully complete all four tasks (the examples are not necessarily all successful attacks). We also collect all the attempts they make between two submissions and consider the total number of attempts as the number of queries. We are hoping to

gather at least 2000 unique and quality adversarial examples among participants from all tasks.

Stage 2: peer-evaluation After collecting and organising generated adversarial examples, we will recruit an independent group of AMT workers to annotate the data. Similar to (Mozes et al., 2021), we plan to select AMT workers based on their historical performance. That is, AMT workers who have successfully completed more than 1000 human intelligence tasks, and have an approval rate that is higher than 98% would be selected for peer-evaluation. We present AMT workers with a few adversarial examples (approximately 50 examples) generated by humans and/or automated algorithms, randomly and anonymously. Each example is evaluated by three AMT workers to reduce variance.

We aim to recruit 30 qualified AMT workers and hope to gather 1500 unique peer-evaluation results from them for about 500 examples.

6 Conclusion & Future Work

Humans have excellent intuition about language, but weak intuition about deep networks; automated attacks are often the opposite. Given the weak performance of manual attacks and automated attacks against NLP systems, some type of human-AI collaboration is essential to truly evaluate their robustness, and to be prepared for the inevitable attacks from real-world adversaries.

In the future, we will carry out the experiments as designed, and further include the IMDB movie review dataset curated by (Maas et al., 2011). As the texts in the IMDB dataset are often longer, this dataset may provide participants greater flexibility in modifying the examples.

We believe that further study into collaboration methods will lead to a better understanding of adversarial attacks and more robust NLP models. We hope to provide a new benchmark for HITL adversarial learning while we continue exploring other effective human-AI collaboration methods. We hope that our framework will help researchers and practitioners better evaluate the robustness of NLP models to the best attacks that humans and algorithms can construct, and then improve their models by training on these adversarial examples.

Acknowledgement

This work was supported by a grant from the Defense Advanced Research Projects Agency (DARPA), agreement number HR00112090135. This work benefited from access to the University of Oregon high-performance computer, Talapas.

References

Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *ICWSM*, pages 512–515.

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

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647, Minneapolis, Minnesota. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6174–6181, Online. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, 32nd Innovative Applications of Artificial Intelligence Conference, and 10th AAAI Symposium on Educational Advances in Artificial Intelligence, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

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. *CoRR*, abs/1907.11692.

Andrew L. 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, Portland, Oregon, USA. Association for Computational Linguistics.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

Maximilian Mozes, Max Bartolo, Pontus Stenetorp, Bennett Kleinberg, and Lewis Griffin. 2021. Contrasting human- and machine-generated word-level adversarial examples for text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8258–8270, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. In *ACL*, pages 4885–4901.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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, Seattle, Washington, USA. Association for Computational Linguistics.

Eric Wallace and Jordan Boyd-Graber. 2018. Trick me if you can: Adversarial writing of trivia challenge questions. In *Proceedings of ACL 2018, Student Research Workshop*, pages 127–133, Melbourne, Australia. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China, November 3-7, 2019*, pages 2153–2162. Association for Computational Linguistics.

Boxin Wang, Chejian Xu, Shuohang Wang, Zhe Gan, Yu Cheng, Jianfeng Gao, Ahmed Hassan Awadallah, and Bo Li. 2021a. Adversarial GLUE: A multi-task benchmark for robustness evaluation of language models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Xiao Wang, Qin Liu, Tao Gui, Qi Zhang, Yicheng Zou, Xin Zhou, Jiacheng Ye, Yongxin Zhang, Rui Zheng, Zexiong Pang, Qinzhuo Wu, Zhengyan Li, Chong Zhang, Ruotian Ma, Zichu Fei, Ruijian Cai, Jun Zhao, Xingwu Hu, Zhiheng Yan, Yiding Tan, Yuan Hu, Qiyuan Bian, Zhihua Liu, Shan Qin, Bolin Zhu, Xiaoyu Xing, Jinlan Fu, Yue Zhang, Minlong Peng, Xiaoqing Zheng, Yaqian Zhou, Zhongyu Wei, Xipeng Qiu, and Xuanjing Huang. 2021b. TextFlint: Unified multilingual robustness evaluation toolkit for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 347–355, Online. Association for Computational Linguistics.

Zhouhang Xie, Jonathan Brophy, Adam Noack, Wencong You, Kalyani Asthana, Carter Perkins, Sabrina Reis, Zayd Hammoudeh, Daniel Lowd, and Sameer Singh. 2021. What models know about their attackers: Deriving attacker information from latent representations. In *Proceedings of the 4th BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 69–78, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhouhang Xie, Jonathan Brophy, Adam Noack, Wencong You, Kalyani Asthana, Carter Perkins, Sabrina Reis, Sameer Singh, and Daniel Lowd. 2022. Identifying adversarial attacks on text classifiers.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080, Online. Association for Computational Linguistics.

Guoyang Zeng, Fanchao Qi, Qianrui Zhou, Tingji Zhang, Bairu Hou, Yuan Zang, Zhiyuan Liu, and Maosong Sun. 2021. Openattack: An open-source textual adversarial attack toolkit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 363–371.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019. Learning to discriminate perturbations for blocking adversarial attacks in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4904–4913, Hong Kong, China. Association for Computational Linguistics.

A Appendix

A.1 User Interface

See Figures 5, 6, and 7 on the next few pages.

A.2 Word Saliency for BAE, TextFooler, and PWWS

We now describe the word salience methods used by BAE, TextFooler, and PWWS. These approaches are first described by (Jin et al., 2020; Ren et al., 2019); we summarize their methods below.

Considering a sentence X consisting of n words $X = \{w_1, w_2, \ldots, w_n\}$, and its true label y, BAE and TextFooler simply delete a word w_i and measure the word importance $I_{w_i}, \forall w_i \in X$ for contributing to the model predictive score $P(X)$. Denote the sentence without w_i as $X_{\setminus w_i}$, where

$$X_{\setminus w_i} = X \setminus \{w_i\} = \{w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n\}.$$

The importance score I_{w_i} is calculated as the difference between the predictive scores before and after deleting word w_i, i.e.

$$I_{w_i} = P(X) - P(X_{\setminus w_i}),$$

if $P(X) = P(X_{\setminus w_i}) = y$;

$$I_{w_i} = (P(y|X) - P(y|X_{\setminus w_i})) + (P(\hat{y}|X_{\setminus w_i}) - P(\hat{y}|X)),$$

if $P(X) = y$ and $P(X_{\setminus w_i}) = \hat{y}$, where $y \neq \hat{y}$.

PWWS first replaces a word w_i with a candidate word w_i^* to form a new sentence $X^* = \{w_1, \ldots, w_i^*, \ldots, w_n\}$, where w_i^* is the best candidate that changes the predictive probability the most, calculated by

$$w_i^* = argmax_{w_i' \in C} P(y|X) - P(y|X'),$$

where $X' = \{w_1, \ldots, w_i', \ldots, w_n\}$, and w_i' is a candidate token among all substitute candidates C for word w_i. Therefore, the most significant predictive probability change is obtained by

$$\Delta P_i^* = P(y|X) - P(y|X^*).$$

PWWS then calculates the standard saliency by replacing w_i with an unknown token via

$$S(X, w_i) = P(y|X) - P(y|\hat{X})$$

where $\hat{X} = \{w_1, \ldots, unknown, \ldots, w_n\}$. A saliency vector $\mathbf{S}(X)$ is obtained by calculating the saliency for every word in the sentence. PWWS finally combines the predictive probability and the saliency vector through a dot product to get a probability weighted saliency score (Ren et al., 2019). That is

$$H(X, X^*, w_i) = \phi \mathbf{S}(X) \cdot \Delta P_i^*,$$

where ϕ is a softmax function. $H(X, X^*, w_i)$ eventually determines the word importance for PWWS.

A.3 Incentive Payment Plan

Each participant at the university is expected to complete the adversarial example generation tasks using all four attack methods for consistency. Therefore, we create an incentive payment plan to motivate participants to work through the four tasks: Auto, Manual, Select, and Saliency. The Auto setting is fairly simple, which we expect participants to finish the task in less than 30 minutes, and we pay $12/person. The Manual setting is slightly more time-consuming and more difficult,

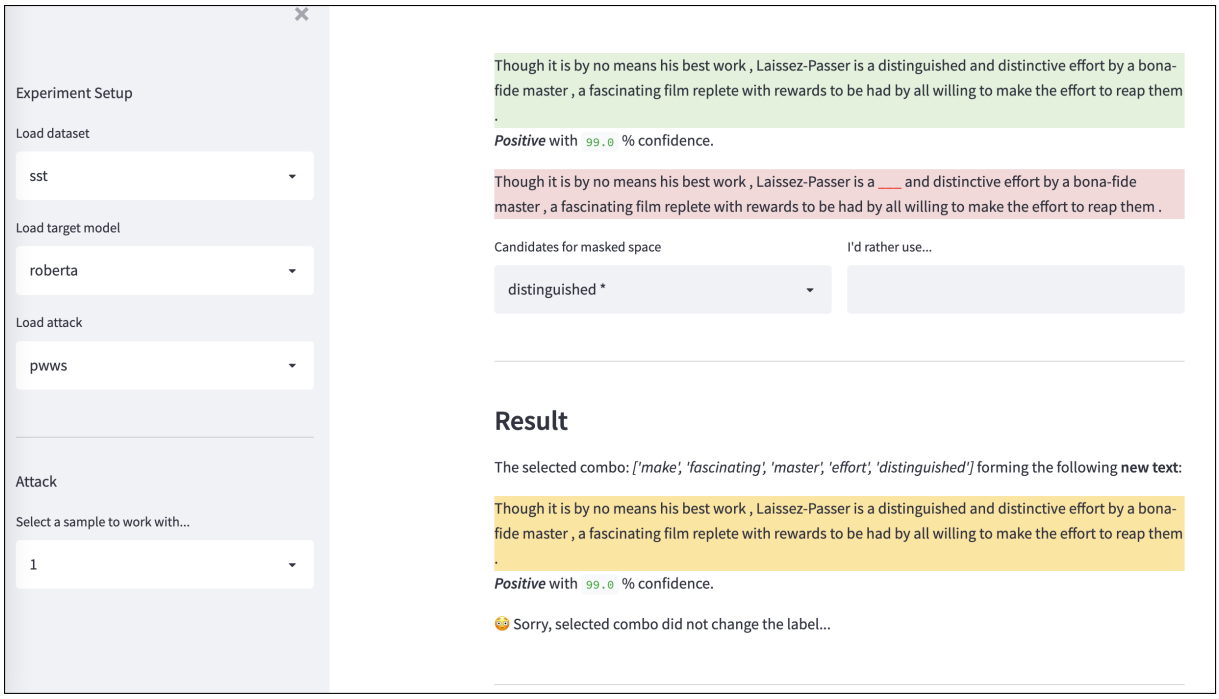

Figure 5: The interface for the Select task

we expect them to finish the task in 60 minutes, and we pay $28/person. The Select and Saliency may also require some effort and attempts so that we expect them to complete the tasks in 90 minutes, and we pay $40/person for each task. By doing so, we hope to keep participants interested and motivated throughout the whole process.

We also plan to reward ten participants $10 who give constructive feedback for our user interface or experiment design through a drawing system. Additionally, we will double the pay for the top three participants who provide the most quality adversarial examples, where the quality is evaluated anonymously on AMT during the peer-evaluation phase.

For peer-evaluation performed on AMT, We will match the market prices and pay $0.2∼0.25/example to the AMT workers. Peer-evaluation is fairly straightforward, and we estimate that it takes no more than 90 minutes for each AMT worker to complete the task.

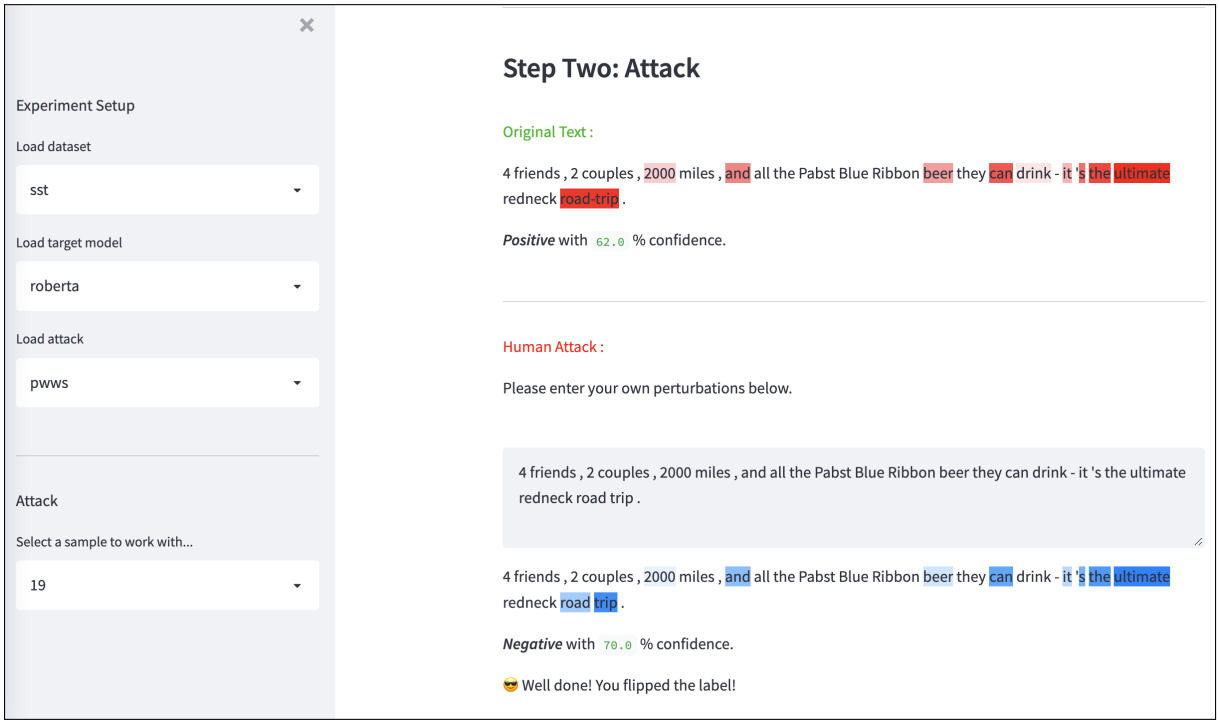

Figure 6: The interface for the Saliency task

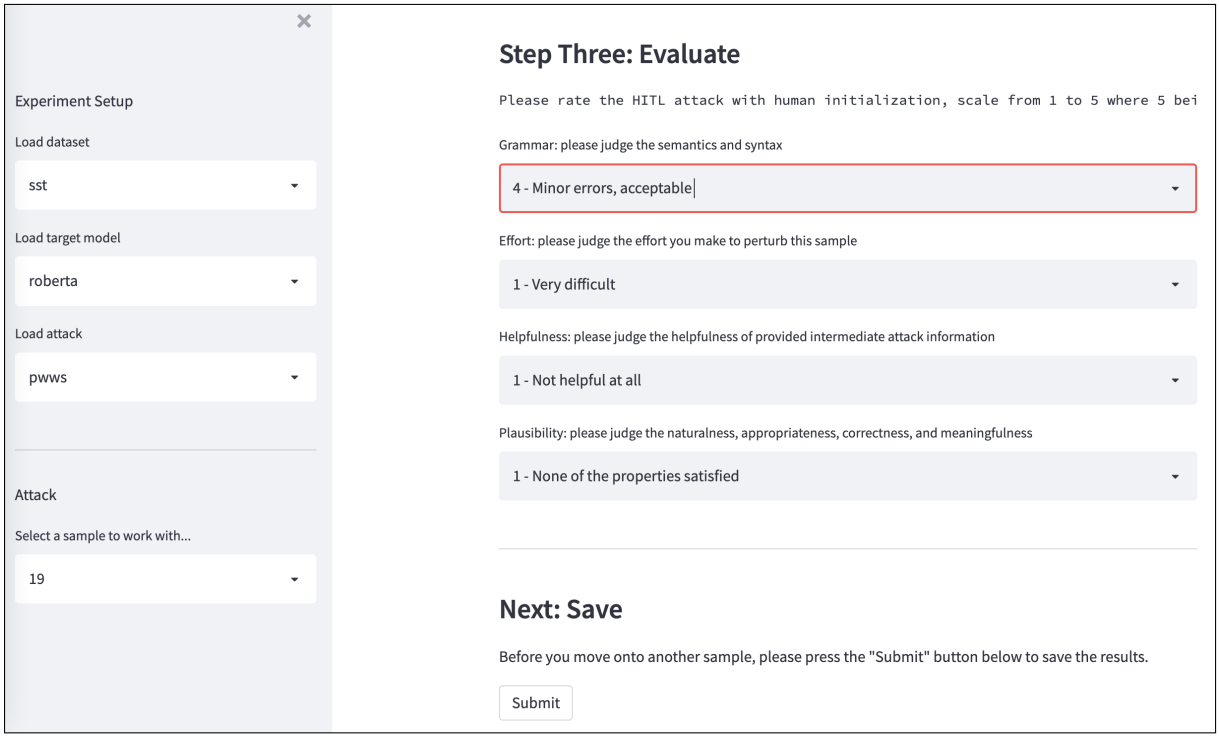

Figure 7: The interface for self-evaluation