XMD: An End-to-End Framework for Interactive Explanation-Based Debugging of NLP Models

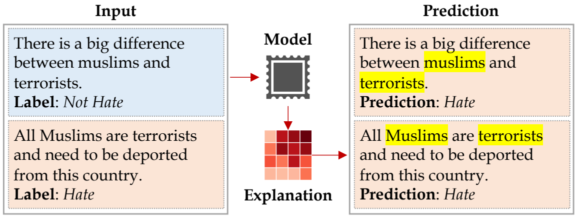
Dong-Ho Lee[1*] **Akshen Kadakia**[1*] **Brihi Joshi**[1] **Aaron Chan**[1] **Ziyi Liu**[1]
Takashi Shibuya[2] **Ryosuke Mitani**[2] **Toshiyuki Sekiya**[2] **Jay Pujara**[1] **Xiang Ren**[1]
[1]Department of Computer Science, University of Southern California
[2]R&D Center, Sony Group Corporation
{dongho.lee, akshenhe, brihijos, chanaaro, zliu2803, jpujara, xiangren}@usc.edu
{Takashi.Tak.Shibuya, Ryosuke.Mitani, Toshiyuki.Sekiya}@sony.com

Abstract

NLP models are susceptible to learning spurious biases (*i.e.,* bugs) that work on some datasets but do not properly reflect the underlying task. Explanation-based model debugging aims to resolve spurious biases by showing human users explanations of model behavior, asking users to give feedback on the behavior, then using the feedback to update the model. While existing model debugging methods have shown promise, their prototype-level implementations provide limited practical utility. Thus, we propose **XMD**: the first open-source, end-to-end framework for explanation-based model debugging. Given task- or instance-level explanations, users can flexibly provide various forms of feedback via an intuitive, web-based UI. After receiving user feedback, **XMD** automatically updates the model in real time, by regularizing the model so that its explanations align with the user feedback. The new model can then be easily deployed into real-world applications via Hugging Face. Using **XMD**, we can improve the model's OOD performance on text classification tasks by up to 18%.[1]

1 Introduction

Neural language models have achieved remarkable performance on a wide range of natural language processing (NLP) tasks (Srivastava et al., 2022). However, studies have shown that such NLP models are susceptible to learning spurious biases (*i.e.,* bugs) that work on specific datasets but do not properly reflect the underlying task (Adebayo et al., 2020; Geirhos et al., 2020; Du et al., 2021; Sagawa et al., 2020). For example, in hate speech detection, existing NLP models often associate certain group identifiers (*e.g.*, *black*, *muslims*) with hate speech, regardless of how these words are actually used (Kennedy et al., 2020b) (Fig. 1). This poses serious concerns about the usage of NLP models for high-stakes decision-making (Bender et al., 2021).



There is a big difference between muslims and terrorists.
Label: *Not Hate*

All Muslims are terrorists and need to be deported from this country.
Label: *Hate*

There is a big difference between muslims and terrorists.
Prediction: *Hate*

All Muslims are terrorists and need to be deported from this country.
Prediction: *Hate*

Figure 1: We make predictions on machine-generated examples (Brown et al., 2020; Hartvigsen et al., 2022) using BERT model fine-tuned on HateXplain (Mathew et al., 2021) and show its explanation using integrated gradients (Sundararajan et al., 2017). It shows spurious correlation between a word *muslims* and the label *hate*.

In response, many methods have been proposed for debiasing either the model or the dataset. Model debiasing can be done via techniques like instance reweighting (Schuster et al., 2019), confidence regularization (Utama et al., 2020), and model ensembling (He et al., 2019; Mahabadi and Henderson, 2019; Clark et al., 2019). Dataset debiasing can be done via techniques like data augmentation (Jia and Liang, 2017; Kaushik et al., 2020) and adversarial filtering (Zellers et al., 2018; Le Bras et al., 2020). However, these methods lack knowledge of which spurious biases actually impacted the model's decisions, which greatly limits their debiasing ability.

On the other hand, *explanation-based model debugging* focuses on addressing spurious biases that actually influenced the given model's decision-making (Smith-Renner et al., 2020; Lertvittayakumjorn and Toni, 2021; Hartmann and Sontag, 2022). In this paradigm, a human-in-the-loop (HITL) user is given explanations of the model's behavior (Sundararajan et al., 2017; Shrikumar et al., 2017) and asked to provide feedback about the behavior. Then, the feedback is used to update the model, in order to correct any spurious biases detected via the user feedback. While existing model debugging methods have shown promise (Idahl et al., 2021; Lertvittayakumjorn et al., 2020;

[*] Both authors contributed equally.

[1] Source code and project demonstration video are made publicly available at `http://inklab.usc.edu/xmd/`

Zylberajch et al., 2021; Ribeiro et al., 2016), their prototype-level implementations provide limited end-to-end utility (*i.e.,* explanation generation, explanation visualization, user feedback collection, model updating, model deployment) for practical use cases.

Given the interactive nature of explanation-based model debugging, it is important to have a user-friendly framework for executing the full debugging pipeline. To achieve this, we propose the E**X**planation-Based NLP **M**odel **D**ebugger (**XMD**). Compared to prior works, **XMD** makes it simple for users to debug NLP models and gives users significant control over the debugging process (Fig. 2). Given either task (model behavior over all instances) or instance (model behavior w.r.t. a given instance) explanations, users can flexibly provide various forms of feedback (*e.g., add* or *remove* focus on a given token) through an easy-to-use, web-based user interface (UI). To streamline user feedback collection, **XMD**'s UI presents intuitive visualizations of model explanations as well as the different options for adjusting model behavior (Fig. 3-4). After receiving user feedback, **XMD** automatically updates the model in real time, by regularizing the model so that its explanations align with the user feedback (Joshi et al., 2022). **XMD** also provides various algorithms for conducting model regularization. The newly debugged model can then be downloaded and imported into real-world applications via Hugging Face (Wolf et al., 2020). To the best of our knowledge, **XMD** is the first open-source, end-to-end framework for explanation-based model debugging. We summarize our contributions as follows:

- **End-to-End Model Debugging**: **XMD** packages the entire model debugging pipeline (*i.e.,* explanation generation, explanation visualization, user feedback collection, model updating, model deployment) as a unified system. **XMD** is agnostic to the explanation method, user feedback type, or model regularization method. **XMD** can improve models' out-of-distribution (OOD) performance on text classification tasks (*e.g.,* hate speech detection, sentiment analysis) by up to 18%.
- **Intuitive UI**: **XMD**'s point-and-click UI makes it easy for non-experts to understand model explanations and give feedback on model behavior.
- **Easy Model Deployment**: Given user feedback, **XMD** automatically updates the model in real time. Users can easily deploy debugged models into real-world applications via Hugging Face.

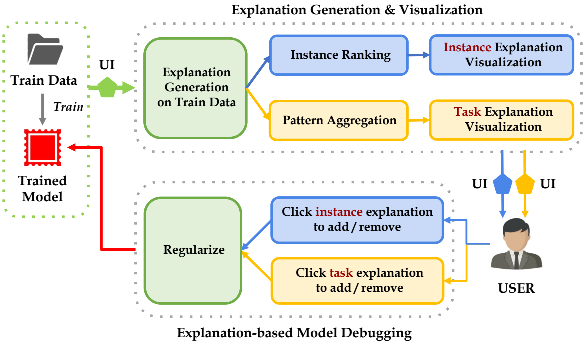

Explanation Generation & Visualization
Train Data
Train
Trained Model
UI
Explanation Generation on Train Data
Instance Ranking
Instance Explanation Visualization
Pattern Aggregation
Task Explanation Visualization
UI
UI
Regularize
Click instance explanation to add / remove
Click task explanation to add / remove
USER
Explanation-based Model Debugging

Figure 2: **System Architecture**

2 Framework Overview

As shown in Figure 2, our framework consists of three main components: *Explanation Generation*; *Explanation Visualization*; and *Explanation-based Model Debugging*. Here, the explanation generation and debugging process are done on the backend while visualizing explanations and capturing human feedback on them are done on front-end UI.

Explanation Generation (§3.1) Humans first input the train data and the model trained on the train data into the framework. On the backend, our framework uses a heuristic post-hoc explanation approach on the model to generate rationales for the train data.

Explanation Visualization (§3.2) The framework visualizes the generated rationales through UI in two different ways: an instance explanation, which shows the explanation for each train instance, and a task explanation, which shows words according to their importance to the prediction.

Explanation-Based Model Debugging (§3.3) The human then decides whether to select words for each instance (Instance Explanation) or words that apply to all instances (Task Explanation) to increase or decrease the word importance. When a human clicks a few words to debug and then decides to end the debugging process, the framework retrains the model with a regularization approach and makes the debugged model downloadable.

3 XMD Framework

In this section, we present each module of **XMD** in processing order. To start the process, the user needs to place a training dataset $\mathcal{D}_T$ and a classification model $\mathcal{M}$ that is trained on $\mathcal{D}_T$.

3.1 Explanation Generation

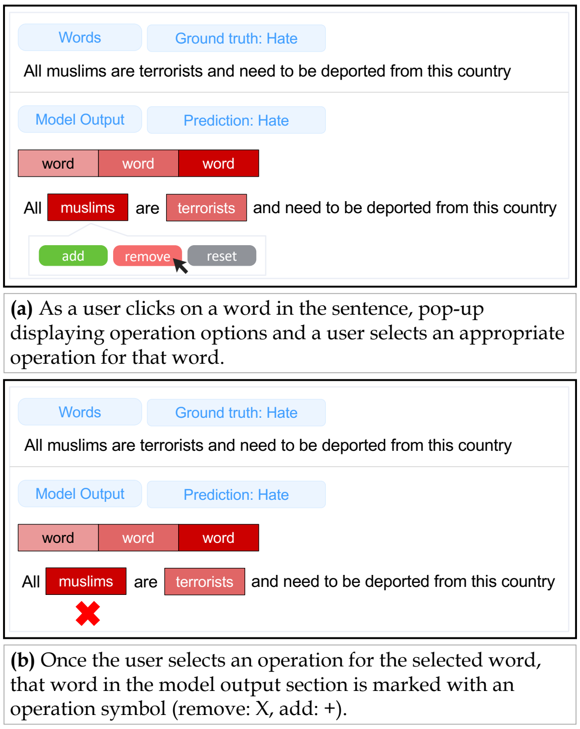
Our explanation generation module outputs rationales from $\mathcal{M}$. For each instance $\mathbf{x} \in \mathcal{D}_T$, $\mathcal{M}$ generates rationales $\phi(\mathbf{x}) = [\phi(\mathbf{w}_1), \phi(\mathbf{w}_2), \ldots, \phi(\mathbf{w}_n)]$ where $\mathbf{w}_i$ denotes the i-th token in the sentence. Each importance score $\phi(\mathbf{w}_i)$ has a score with regard to all the classes. Our module is exploiting $\phi^p(\mathbf{w}_i)$ which the importance score is attributed to model predicted label p. Here, we exploit heuristic methods that assign importance scores ϕ based on gradient changes in $\mathcal{M}$ (Shrikumar et al., 2017; Sundararajan et al., 2017).

3.2 Explanation Visualization

Our framework supports visualizing the generated rationale in two different forms, *instance* and *task* explanations. Instance explanations display word importance scores for model predictions for each train instance, while task explanations aggregate and rank words according to their importance to the predicted label. In this section, we first present a UI for visualizing and capturing human feedback for instance explanations and then a UI for task explanations.

Instance Explanation Figure 3 illustrates how our framework visualizes instance explanations and captures human feedback on them. First, the trained model makes a prediction and generates explanations for one of the train instances that the model correctly predicts: “All muslims are terrorists and need to be deported from this country”. The reason why we present only the instances that the model correctly predicts is that we are asking users to provide feedback for the ground truth label and comparing it with $\phi^p(\mathbf{w}_i)$ which the importance score is attributed to the model predicted label p. If p is not equal to the ground truth label, the human feedback would act as a source of incorrect prediction.

Next, the user is presented with the sentence and its ground truth label on the upper deck (Words Section), and the sentence with highlighted rationales and its predicted label on the lower deck (Model Output Section). Then, the user can choose to select words to decrease or increase its importance toward the ground truth label (Figure 3 (a)). If the user clicks the word (*muslims*) that the model is focusing on to predict *hate*, a small pop-up displaying buttons for operation options (*i.e.*, *add*, *remove* and *reset*) appear. Once the user selects a desired operation (*remove*) for the selected word (*muslims*) that is not a right reason for *hate*, that word in the model output section is marked with operation symbol (’X’ for *remove*, ’+’ for *add* – Figure 3 (b)). The user may cancel their decision to operation for the word by clicking *reset* in the pop-up.



All muslims are terrorists and need to be deported from this country

All muslims are terrorists and need to be deported from this country

(a) As a user clicks on a word in the sentence, pop-up displaying operation options and a user selects an appropriate operation for that word.

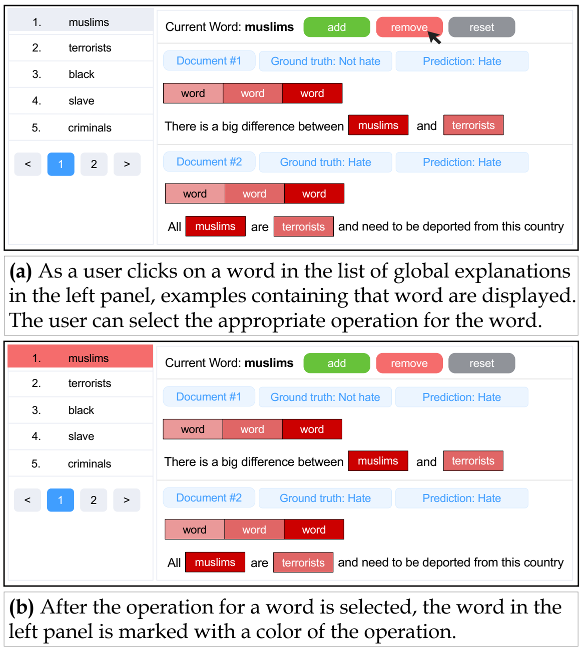
All muslims are terrorists and need to be deported from this country

All muslims are terrorists and need to be deported from this country

(b) Once the user selects an operation for the selected word, that word in the model output section is marked with an operation symbol (remove: X, add: +).

Figure 3: The workflow to provide human feedback on **instance explanations**. Humans provide explanations (*remove “muslims”*) for the ground truth label (*hate*).

Task Explanation Figure 4 illustrates how our framework visualizes task explanations and captures human feedback on them. First, task explanations are presented in list format on the left panel in descending order of its importance (Figure 4 (a)). Here, the importance is a score averaged by the word importance score of all examples containing that word. As user clicks on a word in the list, all the examples containing that word are displayed. The user can then choose to two different operations (*remove* and *add*). If user clicks *remove* for the word (*muslims*) that should not be conditioned on any label (both *hate* and *not hate*), the model will consider it as an unimportant word in all cases. Here, we don’t need to consider whether the prediction is correct or not since the word is not important for all the cases (Figure 4 (a)). If user clicks *add* for the word that should be useful for the correct prediction, the model will consider it as an important word for the ground truth label. Here, we consider it as an important word only for the



There is a big difference between muslims and terrorists

All muslims are terrorists and need to be deported from this country

(a) As a user clicks on a word in the list of global explanations in the left panel, examples containing that word are displayed. The user can select the appropriate operation for the word.

There is a big difference between muslims and terrorists

All muslims are terrorists and need to be deported from this country

(b) After the operation for a word is selected, the word in the left panel is marked with a color of the operation.

Figure 4: The workflow to provide human feedback on **task explanations**. Humans remove the word (*muslims*) that should not be conditioned on any labels (i.e., *hate*, *not hate*).

correct prediction. After the operation for a word is selected, the word in the left panel is marked with a color of that operation (red for *remove* and green for *add*).

3.3 Explanation-based Model Debugging

Our explanation-based model debugging module is based on explanation regularization (ER) which regularizes model to produce rationales that align to human rationales (Zaidan and Eisner, 2008; Ross et al., 2017; Liu and Avci, 2019a; Ghaeini et al., 2019; Kennedy et al., 2020a; Rieger et al., 2020; Lin et al., 2020; Huang et al., 2021; Joshi et al., 2022). Existing works require the human to annotate rationales for each training instance or apply task-level human priors (*e.g.*, task-specific lexicons) across all training instances before training. Despite its effectiveness, the regularized model may not be fully free of hidden biased patterns. To catch all the hidden biased patterns, our framework asks the human to provide binary feedback (*i.e.*, click to add or remove) given the current model explanations and use them to regularize the model. Here we ask the human to provide feedback to the model in order to output the "**correct prediction**".

For the instance explanation, as shown in Figure 5, the trained model $\mathcal{M}$ generates rationales $\phi^p(\mathbf{x}) = [\phi^p(w_1), \phi^p(w_2), \ldots, \phi^p(w_n)]$, where $\phi^p(w_i)$ denotes the importance score of i-th token in the sentence $\mathbf{x}$ towards model predicted label p. As the user selects a word (*muslims*) that is spuriously correlated with the correct prediction p (*hate*), the regularized score $t^p(w_i)$ where i is a user-selected word index (w_2 = *muslims*) becomes 0 (See Figure 5). For the task explanation, we aggregate words based on its score averaged by the word importance score of examples containing that word, and present them in a descending order. When the user clicks a word w (*muslims*) to decrease its importance, then regularized score $t^p(w)$ where w is user-selected word (w = *muslims*) for all the examples become 0 (See Figure 6).

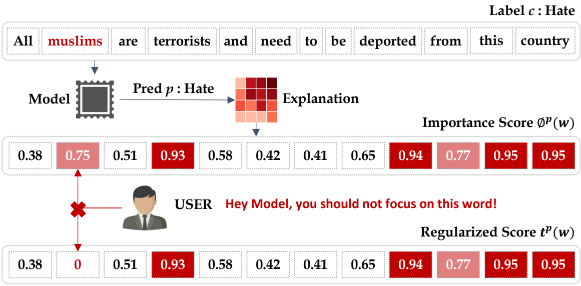


Figure 5: **Instance Explanation-based Model Debugging.** Trained model generates explanations in a form of word importance score $\phi^p(w)$ towards prediction label $\mathbf{p}$. User selects words to *add* or *remove* based on $\phi^p(w)$. The regularization score $t^p(w)$ for the selected words to be removed are 0 while selected words to add are 1.

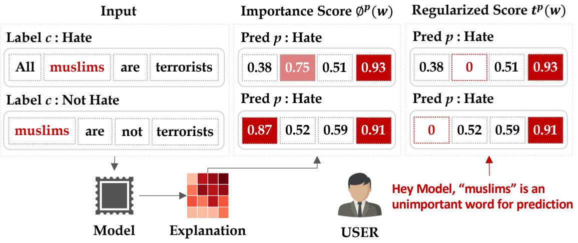


Figure 6: **Task Explanation-based Model Debugging.** Trained model generates explanations in a form of word importance score $\phi^p(w)$ towards prediction label $\mathbf{p}$. As user selects a word to ignore for prediction, the regularization score $t^p(w)$ for the selected word in all the examples that contain that word becomes 0.

After the click process, the user can start the debugging process based on the examples labeled so far. Here, the learning objective for re-training the model $\mathcal{M}$ is $\mathcal{L} = \mathcal{L}_{\text{task}} + \mathcal{L}_{\text{ER}}$, where $\mathcal{L}_{\text{task}}$ is a cross-entropy loss for traditional sequence classification tasks and $\mathcal{L}_{\text{ER}}$ is an explanation regularization loss which minimizes the distance between $\phi^p(w)$ and $t^p(w_i)$ (Joshi et al., 2022). In this framework, we support two different regularization loss:

Mean Squared Error (MSE) (Liu and Avci, 2019b; Kennedy et al., 2020b; Ross et al., 2017), Mean Absolute Error (MAE) (Rieger et al., 2020).

4 Implementation Details

To start **XMD**, users should input the trained model following Hugging Face model structure (Wolf et al., 2020). After users input the train data and the model, our framework uses Captum (Kokhlikyan et al., 2020) to generate explanation. For visualizing the explanation and capturing the human feedback, we implement UI using Vue.js [2]. Here, we re-use UI components from LEAN-LIFE, an explanation-based annotation framework (Lee et al., 2020), for capturing human feedback. To train the model with ER, we use PyTorch (Paszke et al., 2019) and Huggingface (Wolf et al., 2020).

5 Experiments

We conduct extensive experiments investigating how our debugging process affects the performance on in-distributed (ID) and out-of-distribution (OOD) data, and the model explanation. Here, we present experimental results on sentiment analysis for the instance explanation and hate speech detection for the task explanation. For base model, we use BigBird-Base (Zaheer et al., 2020).

Tasks and Datasets For sentiment analysis, we exploit SST (Socher et al., 2013) as the ID dataset, and Yelp (restaurant reviews) (Zhang et al., 2015), Amazon (product reviews) (McAuley and Leskovec, 2013) and Movies (movie reviews) (Zaidan and Eisner, 2008; DeYoung et al., 2019) as OOD datasets. To simulate human feedback for the instance explanation, we leverage ground truth rationales for SST (Carton et al., 2020) as human feedback. For hate speech detection, we use STF (de Gibert et al., 2018) as the ID dataset, and HatEval (Barbieri et al., 2020), Gab Hate Corpus (GHC) (Kennedy et al., 2018) and Latent Hatred (ElSherief et al., 2021) for OOD datasets. To simulate human feedback for the task explanations, we leverage group identifiers (*e.g.*, *black*, *muslims*) (Kennedy et al., 2020b) as words that need to be discarded for determining whether the instance is hate or not.

Regularize	ER Loss	Sentiment Analysis			
		In-distribution	Out-of-Distribution		
		SST	Amazon	Yelp	Movies
None	None	93.4	89.1	89.0	82.0
Correct	MSE	**94.7**	88.4	91.8	**94.5**
	MAE	94.0	**92.3**	**94.4**	94.0

Table 1: **Instance Explanation** ID/OOD Performance (Accuracy). Best models are bold and second best ones are underlined within each metric.

Regularize	ER Loss	Hate Speech Analysis			
		In-distribution	Out-of-Distribution		
		STF	HatEval	GHC	Latent
None	None	89.5	88.2	64.5	67.2
Correct	MSE	89.2	**90.1**	62.3	67.9
	MAE	89.1	**90.1**	59.3	64.9
Incorrect	MSE	88.9	86.3	**67.9**	**70.3**
	MAE	89.3	88.8	64.2	67.6
ALL	MSE	**90.0**	88.4	63.8	67.0
	MAE	89.7	86.9	66.5	70.2

Table 2: **Task Explanation** ID/OOD Performance (Accuracy). Best models are bold and second best ones are underlined within each metric.

ID/OOD Performance Table 1 shows the performance on ID and OOD when regularize on correct predictions using its instance explanation. We see that our framework helps model to not only do much better on ID data, but also generalize well to OOD data. For task explanation, we present performance by regularizing on correct and incorrect prediction and all the instances regardless of prediction. Table 2 presents the performance with *remove* operations for task explanations (*i.e.*, group identifiers) for incorrect predictions, correct predictions, and for all instances, respectively. We observe that our framework helps model not to focus on the words that should not be conditioned on any label and lead to performance enhancement on both ID and OOD data.

Efficiency To quantify the advantage that **XMD** provides, we compare the time taken to annotate instances using **XMD** versus traditional labelling for instance explanations. While **XMD** requires humans to interact with a trained model and decrease or increase importance scores of words, traditional labelling is not model-in-the-loop in nature, and requires users to directly annotate binary importance scores to words in the instance (DeYoung et al., 2019; Carton et al., 2020). We ask two graduate students to annotate 50 instances, using the traditional and the **XMD** labelling methods. For both of these labelling settings, we ensure that there is no overlap between the instances, so as to avoid familiarity and record

[2] https://vuejs.org/

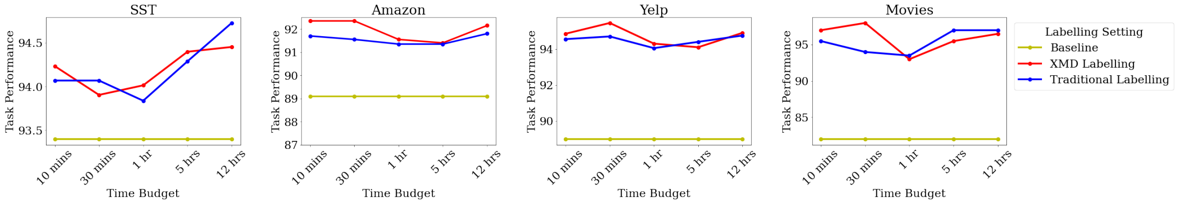


Figure 7: **Time Efficiency:** This simulation assumes two annotators annotating instances in parallel with a strict time budget, using the traditional labelling (∼ 110s/instance) or **XMD** labelling (∼ 60s/instance) methods.

the time taken to annotate each instances. Upon aggregating across all instances and both annotators, it is found that one instance takes ∼ 60 seconds and ∼ 110 seconds to be annotated using the framework and the traditional labelling method respectively. Using this time estimate, we simulate the time-efficiency of these two labelling methods with varying amounts of time budgets for annotations. Figure 7 presents our results for this experiment. We note that although both labelling methods outperforms the baseline of no explanation annotation, using **XMD** is particularly helpful when the time budget given is limited (< 1 hour), especially in the OOD setting (Amazon, Yelp, Movies datasets).

6 Related Work

Spurious Bias Mitigation Recent studies have explored mitigating spurious biases in NLP models. One of the research lines is a dataset debiasing such as adversarial filtering (Zellers et al., 2018; Le Bras et al., 2020) or data augmentation using adversarial data (Jia and Liang, 2017) and counterfactual data (Kaushik et al., 2020). However, creating such datapoints are challenging since they require an exhaustive understanding of the preconceived notions that may cause such spurious biases and the collecting cost is expensive. Another line of research is robust learning techniques such as instance reweighting (Schuster et al., 2019), confidence regularization (Utama et al., 2020), and model ensembling (He et al., 2019; Mahabadi and Henderson, 2019; Clark et al., 2019).

Explanation-Based Model Debugging Many works have explored explanation-based debugging of NLP models, mainly differing in how model behavior is explained, how HITL feedback is provided, and how the model is updated (Lertvittayakumjorn and Toni, 2021; Hartmann and Sonntag, 2022; Balkir et al., 2022). Model behavior can be explained using instance (Idahl et al., 2021; Koh and Liang, 2017; Ribeiro et al., 2016) or task (Lertvittayakumjorn et al., 2020; Ribeiro et al., 2018) explanations, typically via feature importance scores. HITL feedback can be provided by modifying the explanation's feature importance scores (Kulesza et al., 2009, 2015; Zylberajch et al., 2021) or deciding the relevance of high-scoring features (Lu et al., 2022; Kulesza et al., 2010; Ribeiro et al., 2016; Teso and Kersting, 2019). The model can be updated by directly adjusting the model parameters (Kulesza et al., 2009, 2015; Smith-Renner et al., 2020), improving the training data (Koh and Liang, 2017; Ribeiro et al., 2016; Teso and Kersting, 2019), or influencing the training process (Yao et al., 2021; Cho et al., 2019; Stumpf et al., 2009). In particular, explanation regularization (ER) influences the training process so that the model's explanations align with human explanations (Joshi et al., 2022; Ross et al., 2017; Kennedy et al., 2020a; Rieger et al., 2020; Liu and Avci, 2019a).

Our **XMD** system is agnostic to the choice of explanation method or HITL feedback type, while updating the model via ER. Compared to prior works, **XMD** gives users more control over the interactive model debugging process. Given either global or local explanations, users can flexibly provide various forms of feedback via an intuitive, web-based UI. After receiving user feedback, **XMD** automatically updates the model in real time. The debugged model can then be downloaded and imported into real-world applications via Hugging Face (Wolf et al., 2020).

7 Conclusion

In this paper, we propose an open-source and web-based explanation-based NLP Model Debugging framework **XMD** that allows user to provide various forms of feedback on model explanation. This debugging process guides the model to make predictions with the correct reason and lead to significant improvement on model generalizability. We hope that **XMD** will make it easier for researchers and practitioners to catch spurious correlations in the model and debug them efficiently.

References

Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. 2020. Debugging tests for model explanations. *arXiv preprint arXiv:2011.05429*.

Esma Balkir, Svetlana Kiritchenko, Isar Nejadgholi, and Kathleen C Fraser. 2022. Challenges in applying explainability methods to improve the fairness of nlp models. *arXiv preprint arXiv:2206.03945*.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Samuel Carton, Anirudh Rathore, and Chenhao Tan. 2020. Evaluating and characterizing human rationales. *arXiv preprint arXiv:2010.04736*.

Minseok Cho, Gyeongbok Lee, and Seung-won Hwang. 2019. Explanatory and actionable debugging for machine learning: A tableqa demonstration. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1333–1336.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. 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 4069–4082, Hong Kong, China. Association for Computational Linguistics.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

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

Mengnan Du, Varun Manjunatha, Rajiv Jain, Ruchi Deshpande, Franck Dernoncourt, Jiuxiang Gu, Tong Sun, and Xia Hu. 2021. Towards interpreting and mitigating shortcut learning behavior of NLU models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 915–929, Online. Association for Computational Linguistics.

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Reza Ghaeini, Xiaoli Z Fern, Hamed Shahbazi, and Prasad Tadepalli. 2019. Saliency learning: Teaching the model where to pay attention. *arXiv preprint arXiv:1902.08649*.

Mareike Hartmann and Daniel Sonntag. 2022. A survey on improving nlp models with human explanations. *arXiv preprint arXiv:2204.08892*.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3326, Dublin, Ireland. Association for Computational Linguistics.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 132–142, Hong Kong, China. Association for Computational Linguistics.

Quzhe Huang, Shengqi Zhu, Yansong Feng, and Dongyan Zhao. 2021. Exploring distantly-labeled rationales in neural network models. *arXiv preprint arXiv:2106.01809*.

Maximilian Idahl, Lijun Lyu, Ujwal Gadiraju, and Avishek Anand. 2021. Towards benchmarking the utility of explanations for model debugging. *arXiv preprint arXiv:2105.04505*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Brihi Joshi, Aaron Chan, Ziyi Liu, Shaoliang Nie, Maziar Sanjabi, Hamed Firooz, and Xiang Ren. 2022. Er-test: Evaluating explanation regularization methods for nlp models. *arXiv preprint arXiv:2205.12542*.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Brendan Kennedy, Mohammad Atari, Aida M Davani, Leigh Yeh, Ali Omrani, Yehsong Kim, Kris Coombs, Shreya Havaldar, Gwenyth Portillo-Wightman, Elaine Gonzalez, and et al. 2018. Introducing the gab hate corpus: Defining and applying hate-based rhetoric to social media posts at scale.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020a. Contextualizing hate speech classifiers with post-hoc explanation. *arXiv preprint arXiv:2005.02439*.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020b. Contextualizing hate speech classifiers with post-hoc explanation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5435–5442, Online. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

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

Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. 2015. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th international conference on intelligent user interfaces*, pages 126–137.

Todd Kulesza, Simone Stumpf, Margaret Burnett, Weng-Keen Wong, Yann Riche, Travis Moore, Ian Oberst, Amber Shinsel, and Kevin McIntosh. 2010. Explanatory debugging: Supporting end-user debugging of machine-learned programs. In *2010 IEEE Symposium on Visual Languages and Human-Centric Computing*, pages 41–48. IEEE.

Todd Kulesza, Weng-Keen Wong, Simone Stumpf, Stephen Perona, Rachel White, Margaret M Burnett, Ian Oberst, and Amy J Ko. 2009. Fixing the program my computer learned: Barriers for end users, challenges for the machine. In *Proceedings of the 14th international conference on Intelligent user interfaces*, pages 187–196.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *International Conference on Machine Learning*, pages 1078–1088. PMLR.

Dong-Ho Lee, Rahul Khanna, Bill Yuchen Lin, Seyeon Lee, Qinyuan Ye, Elizabeth Boschee, Leonardo Neves, and Xiang Ren. 2020. LEAN-LIFE: A label-efficient annotation framework towards learning from explanation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 372–379, Online. Association for Computational Linguistics.

Piyawat Lertvittayakumjorn, Lucia Specia, and Francesca Toni. 2020. FIND: Human-in-the-Loop Debugging Deep Text Classifiers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 332–348, Online. Association for Computational Linguistics.

Piyawat Lertvittayakumjorn and Francesca Toni. 2021. Explanation-based human debugging of nlp models: A survey. *Transactions of the Association for Computational Linguistics*, 9:1508–1528.

Bill Yuchen Lin, Dong-Ho Lee, Ming Shen, Ryan Moreno, Xiao Huang, Prashant Shiralkar, and Xiang Ren. 2020. Triggerner: Learning with entity triggers as explanations for named entity recognition. *arXiv preprint arXiv:2004.07493*.

Frederick Liu and Besim Avci. 2019a. Incorporating priors with feature attribution on text classification. *arXiv preprint arXiv:1906.08286*.

Frederick Liu and Besim Avci. 2019b. Incorporating priors with feature attribution on text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6274–6283, Florence, Italy. Association for Computational Linguistics.

Jinghui Lu, Linyi Yang, Brian Mac Namee, and Yue Zhang. 2022. A rationale-centric framework for human-in-the-loop machine learning. *arXiv preprint arXiv:2203.12918*.

Rabeeh Karimi Mahabadi and James Henderson. 2019. Simple but effective techniques to reduce biases.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14867–14875.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging nlp models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Laura Rieger, Chandan Singh, William Murdoch, and Bin Yu. 2020. Interpretations are useful: penalizing explanations to align neural networks with prior knowledge. In *International conference on machine learning*, pages 8116–8126. PMLR.

Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. 2020. An investigation of why overparameterization exacerbates spurious correlations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8346–8356. PMLR.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. 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 3419–3425, Hong Kong, China. Association for Computational Linguistics.

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

Alison Smith-Renner, Ron Fan, Melissa Birchfield, Tongshuang Wu, Jordan Boyd-Graber, Daniel S Weld, and Leah Findlater. 2020. No explainability without accountability: An empirical study of explanations and feedback in interactive ml. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13.

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

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Simone Stumpf, Vidya Rajaram, Lida Li, Weng-Keen Wong, Margaret Burnett, Thomas Dietterich, Erin Sullivan, and Jonathan Herlocker. 2009. Interacting meaningfully with machine learning systems: Three experiments. *International journal of human-computer studies*, 67(8):639–662.

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

Stefano Teso and Kristian Kersting. 2019. Explanatory interactive machine learning. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 239–245.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Mind the trade-off: Debiasing nlu models without degrading the in-distribution performance. *arXiv preprint arXiv:2005.00315*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Huihan Yao, Ying Chen, Qinyuan Ye, Xisen Jin, and Xiang Ren. 2021. Refining language models with compositional explanations. *Advances in Neural Information Processing Systems*, 34:8954–8967.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. In *NeurIPS*.

Omar Zaidan and Jason Eisner. 2008. Modeling annotators: A generative approach to learning from annotator rationales. In *Proceedings of the 2008 conference on Empirical methods in natural language processing*, pages 31–40.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. *arXiv:1509.01626 [cs]*.

Hugo Zylberajch, Piyawat Lertvittayakumjorn, and Francesca Toni. 2021. HILDIF: Interactive debugging of NLI models using influence functions. In *Proceedings of the First Workshop on Interactive Learning for Natural Language Processing*, pages 1–6, Online. Association for Computational Linguistics.