Generating Interpretable Data-Based Explanations for Fairness Debugging using Gopher

Jiongli Zhu
University of California, San Diego
La Jolla, CA, USA
jiz143@ucsd.edu

Romila Pradhan
Purdue University
West Lafayette, IN, USA
rpradhan@purdue.edu

Boris Glavic
Illinois Institute of Technology
Chicago, IL, USA
bglavic@hawk.iit.edu

Babak Salimi
University of California, San Diego
La Jolla, CA, USA
bsalimi@ucsd.edu

ABSTRACT

Machine learning (ML) models, while increasingly being used to make life-altering decisions, are known to reinforce systemic bias and discrimination. Consequently, practitioners and model developers need tools to facilitate debugging for bias in ML models. We introduce Gopher, a system that generates compact, interpretable and causal explanations for ML model bias. Gopher identifies the top-k coherent subsets of the training data that are root causes for model bias by quantifying the extent to which removing or updating a subset can resolve the bias. We describe the architecture of Gopher and will walk the audience through real-world use cases to highlight how Gopher generates explanations that enable data scientists to understand how subsets of the training data contribute to the bias of a machine learning (ML) model. Gopher is available as open-source software; The code and the demonstration video are available at https://gopher-sys.github.io/.

CCS CONCEPTS

• **Information systems** → **Data cleaning**; **Data analytics**.

KEYWORDS

Data debugging, Explanations, Interpretability, Fairness

ACM Reference Format:
Jiongli Zhu, Romila Pradhan, Boris Glavic, and Babak Salimi. 2022. Generating Interpretable Data-Based Explanations for Fairness Debugging using Gopher. In *Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22), June 12–17, 2022, Philadelphia, PA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3514221.3520170

1 INTRODUCTION

Despite the success of machine learning (ML) in making life-changing decisions, there are concerns that ML models that are biased can be disproportionately harmful for certain segments of the society. Consequently, there is a need for generating human-understandable explanations for the behavior of ML algorithms to help analysts to debug and address the bias of a model. The field of *eXplainable Artificial Intelligence* (*XAI*) aims to address this issue. The primary focus of XAI has been on generating feature-based explanations that quantify the extent to which input features contribute to an ML model's predictions. This class of explanations includes methods based on feature importance quantification [5, 11, 19], surrogate models, causal and counterfactual methods [8, 11, 14], and differ on whether they address correlational, causal, counterfactual or contrastive patterns. Explanations produced by such systems identify which features of a test data point are correlated with a misprediction or bias. However, they do not explain *why* the model exhibits this bias. If we only consider data as a source of model bias, these methods fail to generate diagnostic explanations that let users trace mispredictions and bias back to the *training data*. For example, feature-based approaches cannot generate explanations of the form: *"The main source of gender bias for this classifier, which decides about loan applications, is its training data, which is biased against the credit scores of unmarried females who are house owners."*

In this demonstration, we present Gopher, a system that assists users in debugging the bias of an ML model. Given a fairness metric, Gopher identifies *coherent* subsets of training data that, when eliminated or updated, remove or reduce the bias. Explanations are represented compactly as *patterns*, e.g., the subset of the training data representing *"unmarried females who are house owners"* would be encoded as a pattern: `gender` = *Female* ∧ `property` = *House*. Such patterns have been used to find data slices in which the model performs poorly [4, 15] but not in the context of fairness. Gopher uses *causal responsibility* toward model bias as the metric to identify the subsets of the training data which significantly impact bias. The causal responsibility of a subset D' of the training data D is the amount of bias reduction that is achieved by removing or perturbing D' and retraining the model over the updated training dataset. Thus, causal responsibility measures the actual impact that D' has on the bias. Gopher uses patterns to compactly describe such subsets of the training data. However, finding the top-k explanations (patterns) with the highest causal responsibility is expensive, because it requires retraining the model for a large number (exponential in the schema size) of candidate explanations. Gopher implements a range of optimizations to be able to efficiently compute top-k explanations: (i) we utilize influence-function-based approximations for causal responsibility of subsets that do not require retraining of a model [3, 10]; and (ii) we prune the search space of patterns by searching through a lattice-based structure inspired by frequent itemset mining [2]. Explanations generated by Gopher help system developers to debug ML algorithms for data errors and bias in training data. See [13] for details of Gopher.

We make the following contributions in this demonstration:

- We present Gopher, a system that generates interpretable, training-data-based explanations for debugging ML model bias by identifying the top-k training data subsets that contribute

SIGMOD '22, June 12–17, 2022, Philadelphia, PA, USA.

ACM ISBN 978-1-4503-9249-5/22/06…$15.00
https://doi.org/10.1145/3514221.3520170

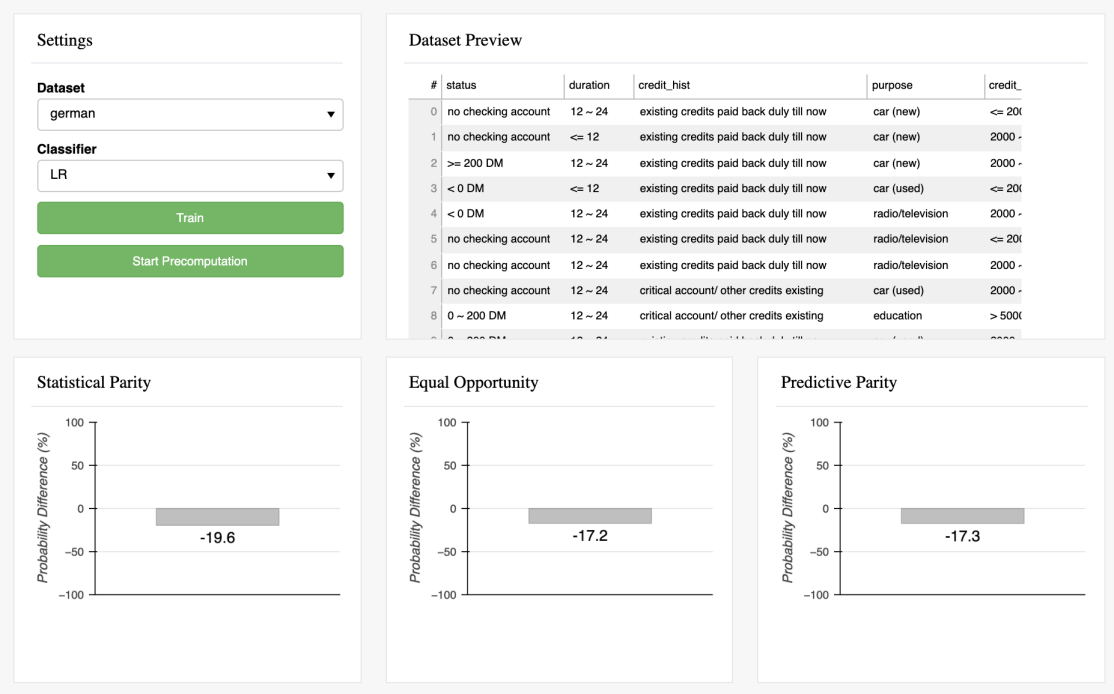

Figure 1: Main user interface of Gopher. The user selects from a list of datasets, and selects one of the supported classifiers. Gopher then shows the bias of the generated classifier according to several fairness metrics.

significantly to model bias. Furthermore, we can identify homogeneous updates to such training data subsets that would reduce bias (e.g., *changing the marital status of female house owners would reduce bias significantly*).

- The demonstration will enable the audience to experience firsthand how Gopher's explanations significantly reduce model bias, and are easily understood and interpreted.
- We will show removal-based and update-based explanations generated by Gopher over real-world datasets. While removal-based explanations identify subsets of the training data most responsible for bias, update-based explanations suggest changes to these subsets so as to reduce the bias.

While the importance of information pertaining the causes of bias for building fair ML algorithms is widely accepted in the algorithmic fairness literature, no current bias mitigation solution fits all situations [7, 9, 18]. Gopher is a novel step in this direction that centers on training data as the source of model bias and detects "issues" for further analysis.

2 SYSTEM OVERVIEW

This section describes the internals of Gopher. Given a training dataset, and a classifier trained on this dataset that exhibits bias in its predictions over some test dataset, Gopher generates the top-k *predicate patterns* that explain the bias of the classifier by the most. In this case, bias of the classifier is captured in terms of *fairness metrics* as described next.

Fairness metrics. Gopher supports generating explanations for three of the most widely used fairness metrics that capture the level of bias inherent in model predictions over different populations of the test data. Given a protected attribute (e.g., gender) that divides the data into *privileged* and *protected* groups of individuals (e.g., males and females), these fairness metrics indicate how biased the model's predictions are for the two groups. *Statistical parity* computes the difference of the probabilities predicted by the classifier on the protected group and the privileged group. *Equal opportunity* is computed as the difference of true positive rates between the protected and the privileged groups. *Predictive parity* is computed as the difference of predicted positive values on the protected group and the privileged group.

Format of explanations. Gopher generates explanations in terms of *patterns* where a pattern ϕ is a conjunction of predicates that represents a subset of training data. For example, the pattern $\phi = (\texttt{gender} = \text{'Female'}) \wedge (\texttt{age} < 45)$ describes data instances where gender is 'Female' and age is less than 45. Such patterns are compact and, hence, are easy to interpret.

To enable sorting patterns with regard to their effect on model bias, Gopher computes the causal responsibility of a pattern through an intervention on the training dataset, which is achieved by either removing or updating the data instances in the pattern.

Removal-based explanations. Gopher identifies patterns such that when the subset of the training data represented by a pattern is removed and the model is retrained on the modified training data, then this causes the greatest possible reduction in model bias. However, instead of removing the subset of data (satisfying a pattern) and retraining the model which is prohibitively expensive, Gopher accurately estimates the *influence* (effect of removing the subset) through second-order influence function approximations [3] (please see [13] for details of Gopher's algorithms). For

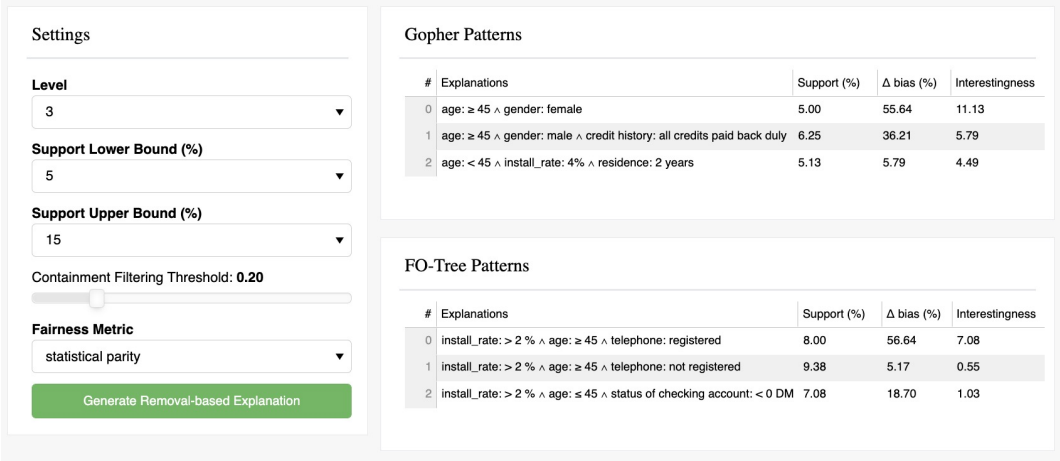

Figure 2: The removal-based explanations module. The user selects the level to indicate the maximum number of predicates in an explanation, and bounds for support of a pattern. Gopher shows the patterns most responsible for bias of the model selected in Figure 1, and reports their support, the reduction in model bias achieved by removing the pattern and its interestingness score. The user also sees how Gopher's explanations compare against a baseline, FO-tree.

efficient influence estimation, Gopher performs a number of precomputations (gradients and Hessian matrices of the loss function of the ML model), and avoids redundant computations in the later steps of explanation generation. Even with this optimization, computing the influence of subsets is prohibitively expensive because the space of patterns grows exponentially with an increase in the number of attribute. To reduce the huge search space of patterns, Gopher applies a lattice-based search (inspired by ideas in frequent itemset mining [2]) with pruning heuristics. First, Gopher only considers patterns whose support (i.e., fraction of the data satisfying the pattern) lies between two user-defined thresholds. The lower threshold prunes subsets that describe a small portion of the training data and, thus, are unlikely to identify systematic issues. The upper threshold ignores subsets that are considered *uninteresting* because the contribution per data instance is very low. The interestingness of a pattern is captured through an *interestingness* score– the higher the score, the more influential the pattern. The lattice structure is defined in *levels* that indicate the number of predicates in a pattern, and hence, its expressiveness. The higher the level, the larger the number of predicates. Level 0 includes patterns with single predicates (e.g., $\texttt{gender} = \textit{Female}$). Level 1 patterns are formed by merging level 0 patterns (e.g., the pattern $\texttt{gender} = \textit{Female} \wedge \texttt{property} = \textit{House}$ is obtained from patterns $\texttt{gender} = \textit{Female}$ and $\texttt{property} = \textit{House}$). Patterns in subsequent levels are formed by merging patterns sharing exactly one predicate at the previous level (e.g., patterns $\texttt{gender} = \textit{Female} \wedge \texttt{property} = \textit{House}$ and $\texttt{gender} = \textit{Female} \wedge \texttt{age} \leq 45$ are merged to form the pattern $\texttt{gender} = \textit{Female} \wedge \texttt{property} = \textit{House} \wedge \texttt{age} \leq 45$). These patterns are further pruned during merging: Gopher does not consider patterns with lower responsibility toward model bias than the patterns it is generated from. Finally, Gopher filters the search results based on a *containment score* that controls the data overlap between generated explanations, and ensures that the set of resulting explanations are diverse.

Update-based explanations. In the spirit of modifying the training data as little as possible, Gopher takes an influential subset of training data as input and, instead of removing it, finds a *homogeneous* modification to this subset that would reduce model bias. The main assumption Gopher makes in this process is that as a result of the modification, the updated model parameters can be obtained from the original model parameters by taking *one* step of gradient descent. Each data point in the subset thus updated is then projected back to the training data distribution by using *projected gradient descent.*

Figures 1 to 3 show the screenshots of Gopher's GUI. In the input module (Figure 1), the user can select a dataset and a classifier. Gopher shows the bias of the classifier trained on the chosen dataset according to several fairness metrics e.g., statistical parity, equal opportunity, and predictive parity. The user can then explore removal-based explanations (Figure 2) and updated-based explanations (Figure 3) for the chosen fairness metric as detailed in the following section.

3 DEMONSTRATION DETAILS

Dataset. We will demonstrate Gopher mainly on the German Credit dataset [6] that consists of information of 1,000 bank account holders with their personal and financial information. The prediction task is determining whether an individual can be classified as a good or bad credit risk. Additionally, we provide two other datasets for users in the demonstration: Adult Income (Adult) [6] and Stop, Question, and FriskData (SQF) [1].

Classifier. Three classic ML models with twice-differentiable loss functions will be available in the demonstration: logistic regression [12], support vector machines (SVM) [12], and a feed-forward neural network [16] with 1 layer and 10 nodes (which is sufficient to obtain high predictive accuracy).

Fairness Metrics. Gopher will compute and present three fairness metrics to quantify the bias of a model including statistical parity, equal opportunity and predictive parity [17].

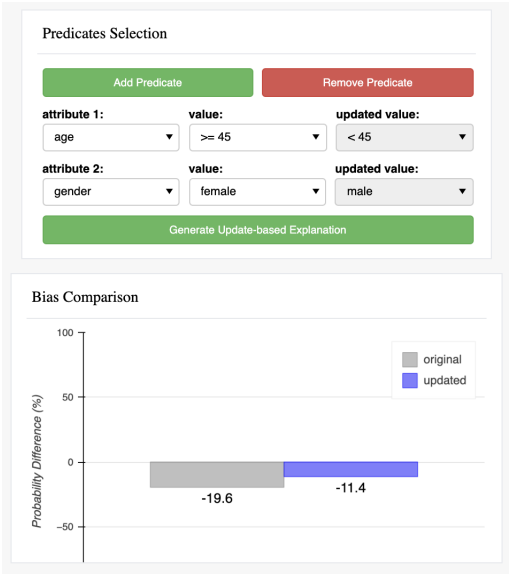

Figure 3: The update-based explanations module. For the pattern selected in the dropdown menu, Gopher indicates how it can be updated (i.e., changes to the predicates) to reduce model bias, and the corresponding reduction compared to the original bias of the model.

Our demonstration will start by selecting a biased dataset and a classifier. Gopher will then train the selected model on the selected dataset and output the corresponding predictive accuracy and fairness metrics. Users can choose one of the fairness metrics to generate explanations.

Removal-based Explanation Generation. Gopher enables users to explore tweaking multiple settings of explanation generation and customize them based on their needs. For example, in Figure 2, Gopher is set to generate level-3 explanations which means there are at most 3 predicates combined in an explanation. The user also selects the lower and upper bounds for the support of the patterns (the fraction of data instances included in the pattern) to be 5% and 15% respectively. Additionally, users can change the diversity of removal-based explanations generated by Gopher by adjusting the containment filter threshold so that the containment scores (fraction of overlapping data instances) between output patterns are ensured to stay less than the threshold. Based on the observed fairness metrics in Figure 1, the user chooses to generate explanations for statistical parity.

Under the same settings, users can compare explanations generated by Gopher using second-order influence functions to those generated by an *FO-tree*. *FO-tree* learns a decision tree regressor over the *first-order* influence approximations of individual training data instances. The best explanation in the *FO-tree* is a path from the root to the node having the highest influence. As seen in the comparative results in Figure 2, Gopher generates explanations that are more *interesting* than *FO-tree* because they achieve similar (or better) reduction in bias by removing fewer data points. Besides, *FO-tree* only returns disjoint subsets while Gopher offers some flexibility by adjusting for the diversity of explanations and generates patterns that might be overlapping in their data instances. Note that patterns generated by *FO-tree* are included in the search space of a special case of Gopher when overlapping is not allowed.

Update-based Explanation Generation. The explanations generated by Gopher consider removal of entire subsets of data instances as represented by the patterns. However, sometimes it may be more desirable to perturb the training dataset only minimally. Instead of removing the subset to mitigate bias, Gopher also supports finding an *update* to a pattern (ideally, an *influential* pattern) such that the dataset is minimally perturbed and model bias is reduced. Gopher lets users identify influential patterns through removal-based explanations (as in Figure Figure 2) and update them (as in Figure 3). Alternatively, Gopher also allows users to specify any pattern that they would like to update. For example, in Figure 3, user chooses to update the pattern: $\texttt{age} < 45 \wedge \texttt{gender} = \textit{Female}$ which was identified as an influential pattern for bias (statistical parity difference in this case) of the Logistic Regression classifier trained on the German dataset in the previous step (as seen in Figure 2). Gopher updates data instances in this pattern to have $\texttt{age} \geq 45$ and $\texttt{gender} = \textit{Male}$, which successfully mitigates bias from -19.6% to -11.4%.

REFERENCES

[1] Nypd stop, question and frisk data. https://www1.nyc.gov/site/nypd/stats/reports-analysis/stopfrisk.page. [Online; accessed 19-October-2021].
[2] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.
[3] Samyadeep Basu, Xuchen You, and Soheil Feizi. On second-order group influence functions for black-box predictions. In *ICML*, pages 715–724, 2020.
[4] Y. Chung, T. Kraska, N. Polyzotis, K. Tae, and S. Whang. Automated data slicing for model validation: A big data - ai integration approach. *IEEE Transactions on Knowledge & Data Engineering*, 32(12):2284–2296, 2020.
[5] Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pages 598–617. IEEE, 2016.
[6] Dheeru Dua and Casey Graff. Uci machine learning repository, 2017.
[7] Runshan Fu, Yan Huang, and P. Singh. Ai and algorithmic bias: Source, detection, mitigation and implications. *Social Science Research Network*, 2020.
[8] Sainyam Galhotra, Romila Pradhan, and Babak Salimi. Explaining black-box algorithms using probabilistic contrastive counterfactuals. In *Proceedings of the 2021 International Conference on Management of Data*, pages 577–590, 2021.
[9] Naman Goel, Alfonso Amayuelas, Amit Deshpande, and Amit Sharma. The importance of modeling data missingness in algorithmic fairness: A causal perspective. In *AAAI*, volume 35, pages 7564–7573, 2021.
[10] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, pages 1885–1894, 2017.
[11] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NIPS*, pages 4765–4774, 2017.
[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *JMLR*, 12:2825–2830, 2011.
[13] Romila Pradhan, Jiongli Zhu, Boris Glavic, and Babak Salimi. Interpretable data-based explanations for fairness debugging. In *SIGMOD*, 2022. preprint available at https://arxiv.org/pdf/2112.09745.
[14] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *SIGKDD*, pages 1135–1144, 2016.
[15] Svetlana Sagadeeva and Matthias Boehm. Sliceline: Fast, linear-algebra-based slice finding for ml model debugging. In *SIGMOD*, pages 2290–2299, 2021.
[16] https://docs.fast.ai/tabular.learner.htm. Fastai neural network.
[17] Sahil Verma and Julia Rubin. Fairness definitions explained. In *Proceedings of the International Workshop on Software Fairness*, pages 1–7, 2018.
[18] Jialu Wang, Yang Liu, and Caleb Levy. Fair classification with group-dependent label noise. In *ACM FAccT*, pages 526–536, 2021.
[19] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *KAIS*, 41(3):647–665, 2014.