Do Machine Learning Models Learn Statistical Rules Inferred from Data?

Aaditya Naik [1] **Yinjun Wu** [1] **Mayur Naik** [1] **Eric Wong** [1]

Abstract

Machine learning models can make critical errors that are easily hidden within vast amounts of data. Such errors often run counter to rules based on human intuition. However, rules based on human knowledge are challenging to scale or to even formalize. We thereby seek to infer statistical rules from the data and quantify the extent to which a model has learned them. We propose a framework SQRL that integrates logic-based methods with statistical inference to derive these rules from a model's training data without supervision. We further show how to adapt models at test time to reduce rule violations and produce more coherent predictions. SQRL generates up to 300K rules over datasets from vision, tabular, and language settings. We uncover up to 158K violations of those rules by state-of-the-art models for classification, object detection, and data imputation. Test-time adaptation reduces these violations by up to 68.7% with relative performance improvement up to 32%. SQRL is available at `https://github.com/DebugML/sqrl`.

1. Introduction

Machine learning models can make a variety of errors due to factors such as noisy data, poor model generalizability, and domain shift. Understanding a model's performance with respect to such errors typically involves the use of quantitative metrics such as accuracy or F1-score.

While these metrics assess a model's performance in an overall sense, they lack the ability to distinguish fundamental errors from mundane ones. Indeed, a model with lower accuracy could even make more coherent predictions than a model with higher accuracy. Making datasets and models even bigger to improve upon these metrics further exacerbates this problem. The net result is that the most critical errors are easily hidden within vast amounts of data.

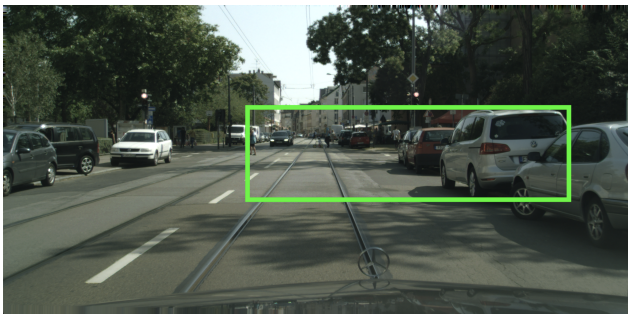


Figure 1: The bounding box predicted as a car illustrates a basic error by EfficientPS, a top performing model for panoptic segmentation in the Cityscapes challenge.

As an example, consider Figure 1, where the green bounding box spanning across the frame is predicted to be a car by EfficientPS (Mohan & Valada, 2021), a top-performing model for panoptic segmentation on the Cityscapes dataset (Cordts et al., 2016). This prediction is an instance of a fundamental error since it obviously defies reasonable notions of the shapes of cars. Such an error may even cause the autonomous vehicle's controller to halt the vehicle abruptly with harmful consequences.

These errors often run counter to rules based on human intuition (e.g., intuition about the width of a typical car with respect to its height). We can therefore cast the problem of estimating such errors in terms of finding rule violations. Such rules can not only help in estimating these errors but also in preventing them by improving the model's predictions with respect to the rules.

A central challenge concerns how to identify such rules at scale. On one hand, the rules must be statistically valid with a well-defined meaning to avoid reporting uninteresting or false violations (i.e., false positives). On the other hand, the rules must go beyond basic concepts and express complex phenomena in the data to avoid missing interesting violations (i.e., false negatives). Even human experts may fail to come up with or agree upon such a set of rules, including identifying relevant statistics, their bounds, and logical predicates over them.

Our key insight is to synthesize a set of statistical quantile

[1]Department of Computer and Information Science, University of Pennsylvania, PA, USA. Correspondence to: Aaditya Naik <asnaik@seas.upenn.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

rules by statistically deriving rules that the training data conforms with. Our approach not only automates the effort to specify a large set of rules but also to determine the goodness of a rule. We have also implemented a general framework, Statistical Quantile Rule Learning (SQRL), which integrates logic-based methods with statistical inference to synthesize such rules from a given dataset without supervision. We also propose to adapt models at test time to reduce violations of the synthesized rules.

We evaluate our approach by applying it to datasets and models from five different domains: tabular classification on a cardiovascular disease dataset (Ulianova), image classification on ImageNet (Deng et al., 2009), object detection on the Cityscapes (Cordts et al., 2016) and KITTI (Geiger et al., 2013) datasets, time-series data imputation over the Physionet dataset (Silva et al., 2012), and sentiment analysis over the Financial PhraseBank dataset (Malo et al., 2014). Our framework generates between 35 to around 300K rules over these datasets and finds between 578 to around 158K violations of those rules on the model predictions. To correct those rule violations, we adapt the models at test time, which reduces the rule violations by up to 68.7% and achieves a relative performance improvement by up to 32%.

We summarize the contributions of our work:

1. We propose statistical quantile rules that estimate basic errors for machine learning datasets and models.
2. We develop a framework SQRL to synthesize valid and expressive rules from data without additional supervision.
3. We propose to improve the models by reducing violations of the rules using test-time adaptation.
4. We evaluate our approach using the rules as metrics over models trained for five different applications.

2. Our Approach

Using rules to characterize basic errors in machine learning is a recurring strategy for improving the trustworthiness and fidelity of models. For example, knowledge graphs and logical constraints have been used for integrating domain knowledge into machine learning pipelines. While these approaches have worked in small and controlled settings, they face several challenges in being applicable to large data settings. Knowledge graphs require substantial human effort to construct even for small datasets, while hard logical constraints can be too simple for noisy datasets often filled with exceptions.

We thereby seek an alternative way that is suited for modern machine learning datasets. To answer this question, we first identify the key desiderata of a representation of such rules:

1. **Validity:** they must be valid for most of the data, but allow for exceptions or noise,
2. **Expressivity:** they must be capable of expressing or capturing complex phenomena and relations, and
3. **Scalability:** they must be generatable without requiring human supervision on individual rules.

We illustrate these criteria using the example from Figure 1. Validity requires that akin to human intuition, any rule that enables the detection of this particular mistake must be true for most other images as well, so as to not raise false alarms. In our example, it is reasonable that the shape of the bounding box is unlikely to represent a car. Expressivity is a competing requirement that states that the representation must be expressive enough to avoid only the most basic or vacuous rules that miss these errors. For example, if using the shape of the car alone is inadequate, it should be possible to use a combination of attributes, such as different ranges of shapes depending on the position of the car. Finally, scalability acknowledges that manually crafting such rules for rich datasets is infeasible, as it involves a combinatorial explosion of features to consider along with statistically valid bounds on them.

2.1. Statistical Quantile Rules

We propose quantile-based statistical inference as a general framework for representing and generating such rules. Specifically, let X be a random variable for a data point, and let $\phi(X)$ be some statistic of X. Then, if

$$\mathbb{P}\left(a \leq \phi(X)\right) = 1 - \delta, \tag{1}$$

for some δ and a, we say that $a \leq \phi(X)$ is a $1-\delta$ quantile rule. In other words, this means that $1-\delta$ of all the data has a statistic $\phi(X)$ that falls above a.

For example, suppose X is a random variable representing bounding boxes of cars from the Cityscapes challenge in Figure 1. Possible statistics $\phi(X)$ for a car are its width, height, or aspect ratio. To obtain the quantile rule for the aspect ratio statistic, we can calculate the aspect ratio of every car in the training data, and find the $1-\delta$ quantile to get the threshold a.

Quantile rules satisfy the above desiderata. First, they are valid for $1-\delta$ of the data, with the δ fraction allowing for exceptions. Second, a quantile rule can be computed for *any* kind of statistic, which can be arbitrarily complex. While the example demonstrates a quantile rule for basic shape properties such as height and width, one can consider more complex statistics such as lighting, textures, or even predictions from other machine learning models. Most importantly, a quantile rule does not need a human to check if a given rule is good or bad. The rule is, by construction, a valid $1-\delta$ quantile for a statistic on the training data.[1] A

[1] One can also easily check if the quantile rule generalizes by

"good" quantile rule provides a tight threshold a, whereas a "bad" quantile rule results in a very conservative threshold a. Although overly conservative thresholds may not be very useful (and may be vacuous), all quantile rules circumvent the need for a human to check for correctness as they are correct by construction.

2.2. Classes of Quantile Rules

The quantile rule framework can capture a wide range of knowledge by varying the statistic ϕ. To highlight the expressivity of the framework, we discuss several general classes of quantile rules that can be viewed as variants of the original quantile rule from (1) that we use in this paper.

Two-sided quantile rules. A simple generalization of the quantile rule is to use a double-sided quantile to get both an upper and a lower bound for a given statistic. Specifically, if

$$\mathbb{P}\left(a \leq \phi(X) \leq b\right) = 1 - \delta, \tag{2}$$

then $\phi(X) \in [a, b]$ is a $1-\delta$ quantile rule. Typically, we can compute a and b to be the lower and upper $\frac{1-\delta}{2}$ quantiles respectively. For example, we can compute not just a lower bound on the aspect ratio, but a complete interval of common aspect ratios.

Mini-batch quantile rules. A generalization of the quantile rule is to consider statistics of not just one but a mini-batch of random variables. Specifically, suppose $\mathbf{X} = (X_1, \ldots, X_m)$ is a minibatch of size m. Then, if

$$\mathbb{P}\left(a \leq \phi(\mathbf{X})\right) = 1 - \delta, \tag{3}$$

then $a \leq \phi(\mathbf{X})$ is a minibatch quantile rule, where ϕ can be any minibatch statistic such as mean or standard deviation.[2]

Logic quantile rules. The statistic ϕ can be much more than a mathematical formula—it can also involve discrete, logical expressions. Specifically, let ψ be a Boolean logical formula, and let $\psi(\mathbf{X})$ be a formula evaluated on every example of a minibatch $\mathbf{X}$. Then, if

$$\mathbb{P}\left(a \leq \phi(\psi(\mathbf{X}))\right) = 1 - \delta, \tag{4}$$

then $a \leq \phi(\psi(\mathbf{X}))$ is a logic quantile rule. For example, ψ could be the logical formula $\texttt{smoke}(x) \Rightarrow \texttt{has\_cardio\_disease}(x)$, and ϕ could be any mini-batch summary statistic such as the F1 score of this rule.

Neural quantile rules. To highlight the expressivity of the statistic, we can consider quantile rules that use statistics of another model, i.e., a neural network. Specifically, let f be a neural model that extracts some features (such as an object detector or an attribute extractor). Then, if

$$\mathbb{P}\left(a \leq \phi(f(X))\right) = 1 - \delta, \tag{5}$$

then $a \leq \phi(f(X))$ is a neural quantile rule. For example, f could be an object detector that extracts various objects such as cars, signs, and pedestrians, while ϕ can be any statistic of the resulting objects such as counts or sizes.

Lastly, we can combine concepts from the above classes of rules to produce even more expressive ones. We illustrate different combinations in the rest of the paper.

3. The SQRL Framework

The space of statistical quantile rules possible for a dataset is unbounded. A large dataset with rich features can induce an enormous set of such rules even with a small bound on the rule size. It is thus infeasible to rely on human supervision to manually craft and check each rule one by one.

We, therefore, develop an end-to-end framework, Statistical Quantile Rule Learning (SQRL), to generate rules at scale. The workflow of SQRL is depicted in Figure 2. We first describe how it generates rules from data and then present how the generated rules can be used to evaluate and improve a model's violations of those rules.

3.1. Generating Statistical Quantile Rules

Our formalism of statistical quantile rules is designed to be applicable to a rich variety of machine learning datasets and models. As such, it is too general for any specific scenario, and we desire a mechanism to enable a domain expert to guide the generation of rules. For instance, the choice of generating logic versus neural quantile rules depends on whether to use a statistic over a logical formula or a neural model's output. Likewise, the choice of sample-based versus mini-batch quantile rules depends on whether a statistic of individual samples or mini-batches is more appropriate.

The power of statistical quantile rules can be attributed to two distinct sources of unboundedness: the structure of the rules and the statistical bounds. Consider the following concrete rule over a sample $(\mathbf{x}, \mathbf{y})$:

$$\mathbf{y} = \mathsf{person} \Rightarrow (0.35 \leq \mathsf{aspect\_ratio}(\mathbf{x}) \leq 0.57),$$

where $\mathbf{x}$ is an image with a single bounding box derived by a neural model, $\mathsf{aspect\_ratio}$ is the aspect ratio of the box, and $\mathbf{y}$ is the object label. This rule illustrates a conditional extension of a two-sided neural quantile rule. We use this form of the rule when the learned bounds apply to a subset of samples that satisfy a condition, e.g., $\mathbf{y} = \mathsf{person}$.

This rule is one of an unbounded number of possible rules in terms of the structure (e.g., choice of neural models, features, and logic predicates) and the statistical bounds

checking the rule on a validation set.

[2]The $1-\delta$ quantile for a minibatch statistic over randomly resampled minibatches can be interpreted as a classic $1-\delta$ confidence interval.

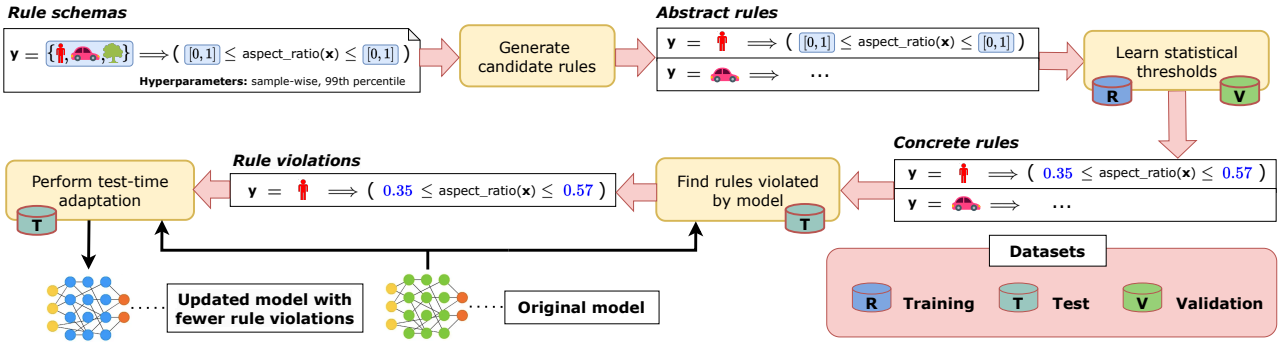


Figure 2: Workflow of our SQRL framework for generating rules and evaluating and adapting models for rule violations.

(e.g., the interval of the aspect ratio).

SQRL allows the user to succinctly specify a *rule schema* S to guide rule generation. The rule schema serves as a prior over the generated rules, analogous to meta-rules in Inductive Logic Programming (Cropper & Muggleton, 2015), mode declarations in Answer Set Programming (Law et al., 2020), and rule templates in logic program synthesis (Raghothaman et al., 2020). Then, the problem we wish to solve can be stated more formally as follows:

Given a rule schema S and a dataset D, output a set of concrete rules R that is:

1. *consistent* with S: no generated rule can include constructs not specified within S,
2. *exhaustive* over S: every single rule consistent with S must be generated, and
3. *valid* with respect to D: each generated rule must hold over $1 - \delta$ of D.

SQRL solves this problem in two phases, using a combination of logic-based methods and statistical inference. In the first phase, it uses logic-based methods to enumerate a set of *abstract rules* with respect to the user-provided rule schema S. In the second phase, it uses statistical inference to compute bounds for each abstract rule to yield a concrete rule that is valid according to D.

The input rule schema S is a set of rule templates of the form $T[\Theta]$ where T specifies the structure of the rule, including any hyperparameters (e.g., mini-batch size and the percentile threshold), and Θ specifies placeholders for the statistical bounds. Rules of different classes defined in Section 2.2 require different forms of schemas. Our running example uses the following schema for two-sided quantile rules:

$$Y \in \{\mathsf{car},\ \mathsf{person},\ \mathsf{rider},\ \ldots\},\ \phi \in \{\mathsf{aspect\_ratio},\ \ldots\},$$
$$\mathbf{y} = Y \Rightarrow \theta_{\mathsf{lb}} \leq \phi(\mathbf{x}) \leq \theta_{\mathsf{ub}},\ 1 - \delta = 0.98,$$

where Y is the set of object labels, ϕ is the set of statistics, and $\Theta = [\theta_{\mathsf{lb}}, \theta_{\mathsf{ub}}]$ are placeholders for the lower and upper bounds on the statistic for a given label for the percentile threshold specified by $1 - \delta$.

The above schema results in a set of several abstract rules:

$$\mathbf{y} = \mathsf{car} \Rightarrow \theta_{\mathsf{lb,car}} \leq \mathsf{aspect\_ratio}(\mathbf{x}) \leq \theta_{\mathsf{ub,car}},$$
$$\mathbf{y} = \mathsf{person} \Rightarrow \theta_{\mathsf{lb,person}} \leq \mathsf{aspect\_ratio}(\mathbf{x}) \leq \theta_{\mathsf{ub,person}},$$
$$\mathbf{y} = \mathsf{rider} \Rightarrow \theta_{\mathsf{lb,rider}} \leq \mathsf{aspect\_ratio}(\mathbf{x}) \leq \theta_{\mathsf{ub,rider}}.$$

The algorithm for generating these abstract rules takes $O(mn^k)$ time, where m is the number of labels in Y, n is the number of statistics in ϕ, and k is the largest number of statistics to be used in a single rule.

Concrete rules are then instantiated from each abstract rule by learning the 98th percentile bounds over the given dataset. An example of such a rule generated over the KITTI dataset (Geiger et al., 2013) is:

$$\mathbf{y} = \mathsf{car} \Rightarrow 0.07 \leq \mathsf{aspect\_ratio}(\mathbf{x}) \leq 2.77.$$

The method to obtain such bounds for each abstract rule is outlined in Algorithm 1. It calculates each abstract rule's statistics over multiple randomly drawn samples (or mini-batches) from the training set, over which the statistical bounds are obtained. Since the bounds may not hold over unseen datasets, we only keep the rules for which the statistical bounds are consistent between the training set and a held-out validation set according to the Jaccard index, which is formulated in Appendix A. For each rule, consider n to be the size of each minibatch. To derive the quantile bounds for that minibatch, we must first sort the statistics, which takes $O(n\log(n))$ time. We learn the bounds themselves using the `numpy.percentile` function. While its complexity is not stated, assuming it is $O(n)$, deriving the bounds for each minibatch takes $O(n\log(n))$ time.

3.2. Using the Statistical Quantile Rules

There are many potential applications of the rules generated by SQRL. In this paper, we focus on two important uses: evaluating and improving models with respect to statistical

Algorithm 1 Computing statistics for abstract rules

Input: Training set R, validation set V, a set of abstract rules $\mathcal{R}_{\text{abs}}$, mini-batch size B, a quantile threshold $1-\delta$, a validation threshold ϵ
Output: A set of candidate rules: $\mathcal{R}_{\text{cand}}$

Randomly sample N_1 mini-batches of size B from R
Randomly sample N_2 mini-batches of size B from V
Initialize the set of output rules as empty: $R_{\text{cand}} = \{\}$
for each rule $r \in R_{\text{abs}}$ **do**
    collect statistics over each of the above N_1 training mini-batches for r;
    compute $1-\delta$ percentile bounds, ϕ_{train}, over those samples;
    collect statistics over each of the above N_2 validation mini-batches for r;
    compute $1-\delta$ percentile bounds, ϕ_{valid}, over those samples;
    if $|\text{Jaccard}(\phi_{\text{train}}, \phi_{\text{valid}})| > 1-\epsilon$ **then**
        add r to $\mathcal{R}_{\text{cand}}$
    end if
end for

quantile rule violations.

To evaluate rule violations, we establish a metric that calculates the number of violations of each rule in the model's predictions on a test set. A rule is violated if the value of $\phi(X)$ lies outside the learned bounds, where X is either a single sample or a minibatch depending on the rule.

We can also attempt to improve the number of violations of a given model on the statistical quantile rules by incorporating an auxiliary semantic loss that captures rule violations. However, it can be impractical to re-train or fine-tune a large model using such a loss, since our framework typically generates a large set of rules. We therefore instead adapt the model's performance at test time.

Consider a statistical quantile rule r of the form $\theta_{\mathsf{lb}} \leq \phi(X)$. We define a loss depending on model parameters W for its output X over a sample (or a mini-batch of samples) as so:

$$l_W(r, X) = \begin{cases} 0 & \text{if } X \text{ satisfies } r \\ \min\{\theta_{\mathsf{lb}} - \phi(X), 1\} & \text{otherwise} \end{cases} \quad (6)$$

Essentially, the loss is a zero or non-zero value assigned to those samples satisfying and violating r respectively. Intuitively, the non-zero loss is the distance between a and the result $\phi(X)$ of evaluating the rule r over the sample X. To avoid potentially large loss values, the loss is clipped at 1. The loss functions for other kinds of rules are similarly defined in Appendix B. Given a set of concrete rules $\mathcal{R}$, we average Equation (6) for all of them as the objective function for test-time adaptation:

$$\mathcal{L}_W(\mathcal{R}, \mathbf{X}) = \frac{1}{|\mathcal{R}|} \sum\nolimits_{r \in \mathcal{R}} l_W(r, \mathbf{X}). \quad (7)$$

The above loss may not be differentiable for some statistics ϕ, such as the F1 score. In such cases, we use their conjugates, such as the conjugate F1 score (Bénédict et al., 2021).

Algorithm 2 sketches the overall test-time adaptation algorithm for reducing rule violations. It evaluates Equation (7) in each iteration and performs back-propagation on all parameters from the batch-normalization layers. The process is repeated for N iterations in total.

Algorithm 2 Rule-based test-time adaptation algorithm

Input: Test dataset T, a set of statistical quantile rules $\mathcal{R}$, and a pretrained model M, number of iterations N

Initialization collect the parameters from all the batch normalization layers of M as a set W and only fine-tune W, keeping all the other parameters in M fixed
while number of iterations is smaller than N **do**
    Randomly sample a mini-batch $\mathbf{X}$ from T
    Evaluate loss $\mathcal{L}_W(\mathcal{R}, \mathbf{X})$ using Equation (7)
    if $\mathcal{L}_W(\mathcal{R}, \mathbf{X}) > 0$ **then**
        Perform back-propagation on M and update W
    end if
end while

4. Evaluation

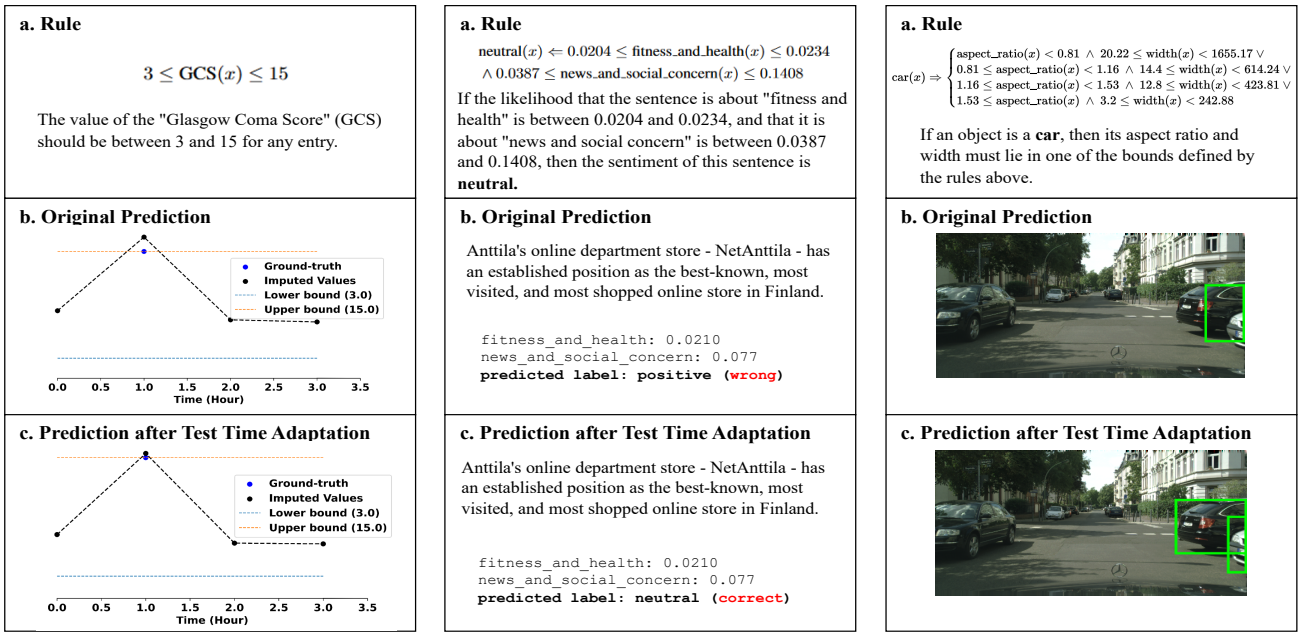
We evaluate SQRL on datasets and models for five different tasks. We present metrics and qualitative aspects of statistical quantile rules generated by the framework from various datasets. We measure how well existing models trained on those datasets perform with respect to the generated rules. We also evaluate the effectiveness of adapting the models at test time to reduce rule violations. We include some additional discussions on the uses and concerns of the generated rules in Appendix E.

Benchmark Tasks and Models. We consider five different tasks: tabular classification, image classification, object detection, time-series imputation, and finally semantic analysis. We describe the datasets below and further detail them in Appendix D.

For tabular classification, we consider the data from the Cardiovascular Disease dataset (Ulianova) over which we train the FT-Transformer model (Gorishniy et al., 2021).

For image classification, we use a ResNet-34 model trained over the original ImageNet dataset (Deng et al., 2009) but use ImageNet-X (Idrissi et al., 2022) for generating the rules and evaluating violations. The ResNet-34 model is trained to classify between the 17 metaclasses defined by ImageNet-X.

For object detection, we consider the object detector component of the EfficientPS model (Mohan & Valada, 2021). Specifically, we leverage a version of the EfficientPS model which is pretrained over the KITTI self-driving dataset



(a) Time series imputation. (b) Sentiment analysis. (c) Object detection.

Figure 3: Qualitative results on time series imputation, sentiment analysis, and object detection tasks (those for the tabular and image classification tasks are in Appendix C.2). For each task, we show (a) a rule generated by SQRL, (b) a model's prediction violating the rule, and (c) the model's prediction satisfying the rule after test-time adaptation. Each of these rules satisfies our three desiderata: they are valid since by construction they hold for 98% of the data; they are expressive enough to find faults in models; and they are generated at scale without human supervision on individual rules.

(Geiger et al., 2013)[3]. We evaluate the model over the validation splits of the KITTI, Cityscapes (Cordts et al., 2016), and $\text{Cityscapes}_{\text{rain}}$ (Tremblay et al., 2020), a version of Cityscapes augmented with heavy rains.

For time series imputation, we trained one state-of-the-art time series imputation model, SAITS (Du et al., 2023) on the Physionet Challenge 2012 dataset (Silva et al., 2012) (Physionet for short). After using the pre-processing scripts from (Tashiro et al., 2021), the resulting time series samples contain around 80% missing entries. The goal of this task is to impute those missing values.

For semantic analysis, we use the pretrained FinBERT model (Araci, 2019) on the Financial PhraseBank dataset (Malo et al., 2014) (PhraseBank for short). The goal of this task is to predict the sentiment (negative, neutral, or positive) of each statement about financial news in this dataset.

4.1. Evaluating Models using Statistical Quantile Rules

To evaluate how well machine learning models capture statistical quantile rules, we synthesize a suite of rules over their corresponding training data and measure the total number of violations of these rules. We describe the setup for each task below and summarize the results in Table 1.

Tabular Classification. The type of rules considered varies across different tasks and is specified via the rule schema. For the tabular classification task, we consider a two-sided form of *logic quantile rules* with the statistic ϕ as the F1 score of the rule over a minibatch. As discussed in Section 2.2, these rules are expressed as boolean formulae over the features from the respective datasets.

We use the columns in the cardiovascular dataset as the features over which the formulae can be generated. While columns such as 'smoke' are boolean-valued, others such as 'age' are not. We bucket the values of such columns into 8 buckets to produce 8 boolean-valued features for values belonging in each bucket. This results in a total of 52 features over which the rules are generated. We generate 300K rules, of which 400 are selected (200 rules per class) for evaluation in the manner described in Algorithm 1. We show an example of a rule and its violating minibatch in Figure 4a in Appendix C.2.

Image Classification. Similar to the previous task, we consider two-sided *logic quantile rules* with the F1 score as the summarization statistic for each minibatch. The features are obtained from the boolean-valued annotations in ImageNet-X, such as background or pose. This yields 16 features over

[3]The pretrained model is downloaded from `https://github.com/DeepSceneSeg/EfficientPS`

which rules are generated. We show an example of a rule and its violation in Figure 4b in Appendix C.2.

Since we evaluate rule violations over the ImageNet validation set, we split the training set of the ImageNet-X dataset into a *rule-training* set and a *rule-validation* set using an 80%/20% split. For each meta-class, we also select 20 rules (i.e., 340 rules in total out of around 73K) in accordance with Algorithm 1.

Object Detection. Due to the nature of the predictions in this task, instead of logic quantile rules, we consider a two-sided form of *neural quantile rules* for a total of 6 statistics related to the bounding box predictions made by the model. These statistics include the aspect ratio, width, height, area, the x-coordinate of the center, and the y-coordinate of the bottom of the bounding box. Each statistic is defined with respect to individual bounding boxes. Statistics are also considered in pairs. For each class, we consider all pairs of statistics (s_1, s_2). For each pair, we group objects of that class by s_1 into four buckets. Within each bucket, we learn the 98 percentile bounds for statistic s_2. This results in generating rules as seen in Figure 3c(a), where s_1 is the aspect ratio and s_2 is the width of each bounding box. A violation of this rule is shown in Figure 3c(b). We learn 252 rules using these 6 statistics for 7 classes that the object detector is trained to predict. In all cases, the 98 percentile bounds for the rules are learned over the KITTI training dataset.

Time Series Imputation. The imputation model outputs estimates of the missing entries. Therefore, for this task, we consider the set of statistics to be the features in the dataset. For each feature, we consider a two-sided form of *neural quantile rules* over the imputed values. As mentioned in Appendix D, there are 35 features in the Physionet dataset resulting in 35 generated logic rules. For each rule, the 98 percentile bounds of the statistics are learned over the non-missing values of the training split of the Physionet dataset. Figure 3a(a) presents an example of a generated rule. This rule shows the bounds learned for the "Glasgow Coma Score" feature, which also matches the universally accepted range of this feature in the medical domain (Teasdale & Jennett, 1974). We show an example of an imputed value violating this rule in Figure 3a(b) where it exceeds the upper bound.

Semantic Analysis. Similar to the previous two tasks, two-sided *neural quantile rules* are considered over each sentence in the PhraseBank dataset. For each sentence, we use pretrained models to extract features that make up the statistics for rules to be generated. We use the pretrained DistilRoBERTa model (Hartmann, 2022) to calculate the likelihood of each of the 7 basic emotions, and the Roberta model pretrained on 11,267 tweets (Antypas et al., 2022) to calculate the likelihood of each of the 19 Twitter topics for each sentence. This results in 26 features generated for each sentence. Similar to the object detection task, for each class we generate rules over each feature, as well as over each pair of features, producing 7.8K rules in all. We then select 158 rules in accordance with Algorithm 1. Figures 3b(a) and 3b(b) show an example of a rule and its violation respectively.

Results. Table 1 shows the total and per sample number of rule violations. For the time series imputation task, the number of rule violations is large (up to around 158K) despite the smaller number of rules (35). This indicates that the state-of-the-art time series imputation models are far from perfect. Apart from this, the image and tabular classification tasks also violate a larger number of rules per sample. Note that the rules for the tabular and image classification tasks are primarily a measure of the consistency of a batch of predictions with respect to the training data. Since we have 200 rules per class for tabular classification to measure this consistency as opposed to 20 for image classification, misclassifications can contribute to a larger number of rule violations in the former task.

On the other hand, there are fewer violations per sample in the object detection and sentiment analysis tasks. Only a minority of the generated rules are violated by the predictions of the object detector. We show the violation results on the Cityscapes-rainy dataset in Table 1, and results on KITTI and Cityscapes in Appendix C.2.

4.2. Adapting Models using Statistical Quantile Rules

We next consider reducing the number of test-time violations of the rules shown in Table 1. We do so by using the rule-driven test-time adaptation technique presented in Section 3. We compare SQRL with state-of-the-art test-time adaptation techniques as baselines.

Baselines. Most test-time adaptation methods, surveyed in Section 5, are orthogonal to our work. We therefore only compare against the following most relevant baselines:

- **Batch normalization**: Batch normalization (BN) (Ioffe & Szegedy, 2015) computes statistics of the batch normalization layers during test time rather than reusing the learned statistics during the training phase.
- **Entropy minimization-based methods**: Tent (Wang et al., 2020) minimizes the entropy of the model predictions, but only focuses on the classification problem. We, therefore, follow the analysis from (Goyal et al.) to minimize the negative L2 norm of the model output for the regression problem in the object detection task.
- **Pseudo-label-based methods**: Robust Pseudo-Label (RPL) (Rusak et al., 2021) and Conjugate PL (CPL) (Goyal et al.) generate pseudo labels as supervisions for test-time adaptations.

Task	Tabular Classification	Image Classification	Object Detection	Time Series Imputation	Sentiment Analysis
Model	FT-Transformer	ResNet-34	EfficientPS	SAITS	FinBERT
Rule Class	Logic	Logic	Neural	Neural	Logic
Sample Size	4096	256	1	1	128
Dataset **R**	$\text{Cardiovascular}_{\text{train}}$	$\text{ImageNet}_{\text{rule-train}}$	$\text{KITTI}_{\text{train}}$	$\text{Physionet}_{\text{train}}$	$\text{PhraseBank}_{\text{train}}$
Dataset **V**	$\text{Cardiovascular}_{\text{valid}}$	$\text{ImageNet}_{\text{rule-valid}}$	-	$\text{Physionet}_{\text{valid}}$	$\text{PhraseBank}_{\text{valid}}$
Dataset **T**	$\text{Cardiovascular}_{\text{test}}$	$\text{ImageNet}_{\text{valid}}$	$\text{Cityscapes}_{\text{rainy,valid}}$	$\text{Physionet}_{\text{test}}$	$\text{PhraseBank}_{\text{test}}$
# Total Rules	292,129	73,032	252	35	7878
# Selected Rules	400	340	252	35	158
# Total Violations	26083±56	7716±318	8,108	157772±18697.80	578
# Rules Violated per Sample	389.29±0.83	42.62±1.76	16.21	197.46±23.40	0.59

Table 1: Results of generating rules and evaluating models against them. Rules are generated over each training dataset using the specified sample size. For sample sizes larger than one, we randomly sample a batch from the dataset. The validation datasets are used to sample a subset of generated rules for evaluation. We count the number of selected rules violated by the model on each test sample. The last two rows show the sum and average of these counts over all test samples respectively.

Task	Tabular Classification		Image Classification		Object Detection		Time series Imputation		Sentiment Analysis	
Dataset	Cardiovascular		ImageNet-X		Cityscapes-rainy		Physionet		PhraseBank	
Metric	AUC	% Reduced	Accuracy	% Reduced	mAP	% Reduced	MSE ($\times 10^{-3}$)	% Reduced	Accuracy	% Reduced
No adapt	80.10	-	80.47	-	25.49	-	7.7	-	84.02	-
BN	80.10	0.0%	80.47	2.9±3.4%	25.49	-0.0±0.0%	7.7	0.0±0.0%	84.02	0.0±0.0%
Tent	80.10	-0.1±0.0%	**80.51**	0.7±5.3%	25.52	-1.4±0.8%	8.0	-0.2±0.9%	**86.08**	14.7±0.0%
RPL	80.09	-0.1±0.0%	80.48	-0.8±6.5%	25.70	0.2±0.3%	7.4	18.7±16.2%	84.85	7.3±0.2%
CPL	80.10	-0.1±0.0%	80.44	-0.1±6.0%	24.95	-0.1±0.1%	5.6	12.7±6.5%	85.77	**15.7±0.1%**
MEMO	80.10	1.8±0.0%	80.47	11.6±4.1%	-	-	-	-	**86.08**	15.0±0.2%
Ours	**80.12**	**38.1±0.0%**	**80.51**	**22.2±4.5%**	**27.01**	**31.4±0.2%**	**5.2**	**68.7±8.1%**	**86.08**	**15.7±0.1%**

Table 2: Results of reducing rule violations using different test-time adaptation methods for our three tasks.

- **Data augmentation-based methods**: MEMO (Zhang et al., 2021) generates multiple augmented versions of a single test sample and minimizes the entropy of the model output across these samples. Since it primarily deals with classification models, generalizing it to object detection models and time series imputation models requires non-trivial efforts. We, therefore, ignore the comparison between MEMO and SQRL in the object detection and time series imputation tasks.

For the baseline methods, we follow the default setups that perform test-time adaptation for a few epochs since overfitting can occur with more epochs. We use up to 60 epochs for SQRL. We follow the default setups of test-time adaptation by only fine-tuning the statistics of the batch normalization layers rather than the entire model. We also perform an ablation study in Appendix C.1 to analyze the effect of fine-tuning the entire model with SQRL.

Results. The evaluation results are shown in Table 2. We report two metrics for all methods: the model performance and the percentage of rule violations reduced ("% Reduced") with respect to the total violations reported in Table 1 after adaptation. For the model performance metrics, we report the AUC score for tabular classification (since it is a binary classification task), the prediction accuracy for image classification, the Mean Average Precision (mAP) for object detection, the Mean Square Error (MSE) for time series imputation and the accuracy for sentiment analysis. SQRL is able to significantly reduce the number of rule violations at test-time (by up to 68.7% in the Physionet dataset), which can also lead to up to 32% relative performance improvement in the model in the Physionet dataset (reducing the MSE from 7.7×10^{-3} to 5.2×10^{-3}) and a slight performance improvement in other datasets. We show examples of time series imputation, sentiment analysis, and object detection in Figure 3, wherein each subfigure, the incorrect predictions violating the rule shown in (a) are depicted in (b) which are corrected in (c) after test-time adaptation. We include more examples from the tabular and image classification tasks in Figure 4 in Appendix C.2. For example, as

Figure 3a(c) shows, SQRL could pull the imputed values at the 1_{st} hour (the black dot) closer to the ground truth (the blue dot) after test-time adaptation. In contrast, the baselines negligibly affect the violations and produce models which perform slightly worse than SQRL.

5. Related Work

Test-time adaptation. Test-time adaptation aims to adapt models to new data distributions in the presence of distribution shift, which typically assumes no access to the source data or the ground-truth labels of data on the target distribution. These approaches involve minimizing the entropy of the model output (Wang et al., 2020; Goyal et al.), producing pseudo-labels to perform supervised training (Goyal et al.; Rusak et al., 2021), or self-supervision (Zhang et al.; Chen et al., 2022) at test-time; enforcing model output to be consistent across different augmentations of one test sample (Zhang et al., 2021); constructing generative models for producing labeled test samples (Li et al., 2020); etc. To our knowledge, none of the existing test-time adaptation methods take into account enforcing the model output to align with statistical quantile rules. In addition, most of these techniques leverage strategies orthogonal to ours, such as meta-learning (Xiao et al., 2021) or generative models (Li et al., 2020). Furthermore, most of them are limited to classification tasks while our method is applicable to generic tasks, including both classification and regression tasks.

Injecting rules into ML models. There have been many efforts to effectively inject rules or domain knowledge into neural nets. One widely adopted strategy is to add a term encoding violations of logic rules to regularize the objective function. For instance, (Hu et al., 2016) designed a teacher-student framework for jointly learning from labeled samples and logic rules, (Seo et al., 2021) jointly learns embeddings of logic rules and training data, and (Ganchev et al., 2010) regularizes the model posteriors with constraints on data. Some other solutions in this area include penalizing bias rules through adversarial learning (Zhang et al., 2018) and solving a constrained optimization problem during training by regarding logic rules as constraints (Fioretto et al., 2021; Narasimhan, 2018). However, all of these approaches demand manually specified rules as input, whereas our methods can automatically learn symbolic rules and their statistical bounds. Moreover, the state-of-the-art solutions to integrate rules into ML models primarily focus on enhancing supervised learning whereas our method is able to deal with a more challenging setting, i.e., test-time adaptation. As mentioned above, there is no supervision from data during test-time adaptation and thus it is impossible to regularize model training with rules.

Rule learning. The problem of synthesizing first-order logic rules from data has been extensively studied in the literature on inductive logic programming (ILP). Techniques such as ILASP (Law et al., 2020) and Prosynth (Raghothaman et al., 2020) leverage pure symbolic reasoning to search logic rules that can produce expected answers in a given database. Works such as NeuralLP (Yang et al., 2017) and NLIL (Yang & Song, 2019) attempt to leverage neural networks to guide the search for feasible logic rules. NLIL can also learn one simple statistic, the confidence score, for each synthesized rule. However, it is less expressive than the arbitrary statistics captured by our framework. Besides, confidence scores are calculated over the entire dataset rather than over mini-batches of data. We therefore cannot check their consistency between different portions of the data.

Weakly supervised learning with logic rules. Similar to test-time adaptation, weakly supervised learning is also applicable when no ground-truth labels exist. One weak supervision strategy is to employ logic rules to annotate unlabeled samples, which is then followed by regular supervised learning (Ratner et al., 2016; 2017). However, to our knowledge, state-of-the-art rule-based weakly supervised learning approaches can only handle classification tasks. In contrast, our solutions are applicable in very general machine learning settings, including both classification and regression tasks. In addition, although some works such as (Varma & Ré, 2018) can automatically derive logic rules as weak supervision signals, they are limited to reasoning about symbolic rules. Hence, they are not applicable to our setting where statistics are necessary.

6. Conclusion and Future Work

We formalized statistical quantile rules as a means of characterizing basic errors inconsistent with training data and defined the problem of extracting such rules at scale. We proposed SQRL, a general framework to generate a large number of such rules for a given dataset and evaluate violations of these rules by a model. Through our extensive empirical studies, we found that machine learning models do not always learn statistical rules inferred from data but can be adapted to correct these rule violations by leveraging rule-based test-time adaptation. In the future, we intend to evaluate statistical quantile rules more widely over modern models. We also intend to explore more uses of these rules, like more effective ways to train models to reduce rule violations and using them for unsupervised learning.

Acknowledgements

We thank the reviewers for their feedback. This research was supported in part by NSF grants #1836936 and #2107429.

References

Antypas, D., Ushio, A., Camacho-Collados, J., Neves, L., Silva, V., and Barbieri, F. Twitter Topic Classification. In *Proceedings of the 29th International Conference on Computational Linguistics*, Gyeongju, Republic of Korea, October 2022. International Committee on Computational Linguistics.

Araci, D. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*, 2019.

Bénédict, G., Koops, V., Odijk, D., and de Rijke, M. sigmoidf1: A smooth f1 score surrogate loss for multilabel classification. *arXiv preprint arXiv:2108.10566*, 2021.

Chen, D., Wang, D., Darrell, T., and Ebrahimi, S. Contrastive test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 295–305, 2022.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223, 2016.

Cropper, A. and Muggleton, S. Logical minimisation of meta-rules within meta-interpretive learning. In *Inductive Logic Programming*, 2015.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Du, W., Côté, D., and Liu, Y. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023.

Fioretto, F., Van Hentenryck, P., Mak, T. W., Tran, C., Baldo, F., and Lombardi, M. Lagrangian duality for constrained deep learning. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part V*, pp. 118–135. Springer, 2021.

Ganchev, K., Graça, J., Gillenwater, J., and Taskar, B. Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 11:2001–2049, 2010.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

Gorishniy, Y., Rubachev, I., Khrulkov, V., and Babenko, A. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34: 18932–18943, 2021.

Goyal, S., Sun, M., Raghunathan, A., and Kolter, J. Z. Test time adaptation via conjugate pseudo-labels. In *Advances in Neural Information Processing Systems*.

Hartmann, J. Emotion english distilroberta-base. `https://huggingface.co/j-hartmann/emotion-english-distilroberta-base/`, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hu, Z., Ma, X., Liu, Z., Hovy, E., and Xing, E. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2410–2420, 2016.

Idrissi, B. Y., Bouchacourt, D., Balestriero, R., Evtimov, I., Hazirbas, C., Ballas, N., Vincent, P., Drozdzal, M., Lopez-Paz, D., and Ibrahim, M. Imagenet-x: Understanding model mistakes with factor of variation annotations. *arXiv preprint arXiv:2211.01866*, 2022.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Law, M., Russo, A., and Broda, K. The ilasp system for inductive learning of answer set programs. *arXiv preprint arXiv:2005.00904*, 2020.

Li, R., Jiao, Q., Cao, W., Wong, H.-S., and Wu, S. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9641–9650, 2020.

Malo, P., Sinha, A., Korhonen, P., Wallenius, J., and Takala, P. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796, 2014.

Mohan, R. and Valada, A. Efficientps: Efficient panoptic segmentation. *International Journal of Computer Vision*, 129(5):1551–1579, 2021.

Narasimhan, H. Learning with complex loss functions and constraints. In *International Conference on Artificial Intelligence and Statistics*, pp. 1646–1654. PMLR, 2018.

Raghothaman, M., Mendelson, J., Zhao, D., Naik, M., and Scholz, B. Provenance-guided synthesis of datalog programs. In *Proceedings of the ACM Symposium on Principles of Programming Languages (POPL)*, 2020.

Ratner, A., Bach, S. H., Ehrenberg, H., Fries, J., Wu, S., and Ré, C. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, pp. 269. NIH Public Access, 2017.

Ratner, A. J., De Sa, C. M., Wu, S., Selsam, D., and Ré, C. Data programming: Creating large training sets, quickly. *Advances in neural information processing systems*, 29, 2016.

Rusak, E., Schneider, S., Pachitariu, G., Eck, L., Gehler, P. V., Bringmann, O., Brendel, W., and Bethge, M. If your data distribution shifts, use self-learning. *Transactions of Machine Learning Research*, 2021.

Seo, S., Arik, S., Yoon, J., Zhang, X., Sohn, K., and Pfister, T. Controlling neural networks with rule representations. *Advances in Neural Information Processing Systems*, 34: 11196–11207, 2021.

Silva, I., Moody, G., Scott, D. J., Celi, L. A., and Mark, R. G. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pp. 245–248. IEEE, 2012.

Tashiro, Y., Song, J., Song, Y., and Ermon, S. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.

Teasdale, G. and Jennett, B. Assessment of coma and impaired consciousness: a practical scale. *The Lancet*, 304 (7872):81–84, 1974.

Tremblay, M., Halder, S. S., de Charette, R., and Lalonde, J.-F. Rain rendering for evaluating and improving robustness to bad weather. *International Journal of Computer Vision*, 2020.

Ulianova, S. Cardiovascular Disease dataset. `https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset`.

Varma, P. and Ré, C. Snuba: Automating weak supervision to label training data. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 12, pp. 223. NIH Public Access, 2018.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2020.

Xiao, Z., Zhen, X., Shao, L., and Snoek, C. G. Learning to generalize across domains on single test samples. In *International Conference on Learning Representations*, 2021.

Yang, F., Yang, Z., and Cohen, W. W. Differentiable learning of logical rules for knowledge base reasoning. *Advances in neural information processing systems*, 30, 2017.

Yang, Y. and Song, L. Learn to explain efficiently via neural logic inductive learning. *arXiv preprint arXiv:1910.02481*, 2019.

Zhang, B. H., Lemoine, B., and Mitchell, M. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 335–340, 2018.

Zhang, M. M., Levine, S., and Finn, C. Memo: Test time robustness via adaptation and augmentation. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

Zhang, Z., Chen, W., Cheng, H., Li, Z., Li, S., Lin, L., and Li, G. Divide and contrast: Source-free domain adaptation via adaptive contrastive learning. In *Advances in Neural Information Processing Systems*.

A. Additional Details of Jaccard Index

As introduced in Section 3.1, the Jaccard index is used for selecting consistent statistical rules between the training and validation sets. To illustrate how to compute the Jaccard index, we revisit the running example in Section 3.1 which defines the following schema:

$$\begin{aligned} &Y \in \{\mathsf{car},\ \mathsf{person},\ \mathsf{rider},\ \ldots\}, \\ &\phi \in \{\mathsf{aspect\_ratio},\ \ldots\}, \quad \gamma = 0.98, \\ &\mathbf{y} = Y \Rightarrow \theta_{\mathsf{lb}} \leq \phi(\mathbf{x}) \leq \theta_{\mathsf{ub}}, \end{aligned}$$

which can be evaluated on both the training set and validation set, leading to the following instantiations of the two-sided quantile rules on the training set and validation set respectively:

$$\begin{aligned} &\mathbf{y} = Y \Rightarrow \theta_{\mathsf{lb,train}} \leq \phi_{\mathsf{train}}(\mathbf{x}) \leq \theta_{\mathsf{ub,train}}, \text{ for training set,} \\ &\mathbf{y} = Y \Rightarrow \theta_{\mathsf{lb,valid}} \leq \phi_{\mathsf{valid}}(\mathbf{x}) \leq \theta_{\mathsf{ub,valid}}, \text{ for validation set.} \end{aligned}$$

The Jacaard index between the above two rules is thus formulated as follows:

$$\begin{aligned} \text{Jaccard}(\phi_{\mathsf{train}}, \phi_{\mathsf{valid}}) &= \frac{[\theta_{\mathsf{lb,train}}, \theta_{\mathsf{ub,train}}] \bigcap [\theta_{\mathsf{lb,valid}}, \theta_{\mathsf{ub,valid}}]}{[\theta_{\mathsf{lb,train}}, \theta_{\mathsf{ub,train}}] \bigcup [\theta_{\mathsf{lb,valid}}, \theta_{\mathsf{ub,valid}}]} \\ &= \frac{\min\{\theta_{\mathsf{ub,train}}, \theta_{\mathsf{ub,valid}}\} - \max\{\theta_{\mathsf{lb,train}}, \theta_{\mathsf{lb,valid}}\}}{\max\{\theta_{\mathsf{ub,train}}, \theta_{\mathsf{ub,valid}}\} - \min\{\theta_{\mathsf{lb,train}}, \theta_{\mathsf{lb,valid}}\}}. \end{aligned}$$

For one-sided quantile rules, i.e., either $\theta_{\mathsf{lb}} = -\infty$ or $\theta_{\mathsf{ub}} = \infty$, then we take the lower bound or upper bound of the statistics $\phi(\mathbf{x})$, which is calculated from the entire dataset to replace θ_{lb} or θ_{ub}.

B. Test-time Adaptation Loss Functions

Recall the intuition behind the loss function mentioned in Equation 6: The loss $l_W(r, X)$ for rule r of a model with parameters W and its output X is zero if X satisfies r, and non-zero if it doesn't. The non-zero value assigned depends on the class of the quantile rule. Equation 6 shows the loss for the basic form of quantile rules. We now define the loss functions used for test-time adaptation for some extensions of quantile rules used in this paper.

1. **Two-sided Quantile Rules.** We extend the definition from (6) for two-sided quantile rules. Assume a two-sided quantile rule $r := \theta_{\mathsf{lb}} \leq \phi(X) \leq \theta_{\mathsf{ub}}$. We define the loss for such a rule as so:

$$l(r; X) = \begin{cases} 0 & X \text{satisfies } r \\ \min\{(\theta_{\mathsf{lb}} - \phi(X))(\theta_{\mathsf{ub}} - \phi(X)), 1\} & \text{otherwise} \end{cases} \tag{8}$$

   Note that the non-zero loss is a quadratic function of $\phi(X)$ with the upper and lower bounds being the roots, but with the output of $\phi(X)$ clipped at 1. This ensures a positive non-zero loss for a rule violation when the value of $\phi(X)$ is either less than θ_{lb} or more than θ_{ub}.

2. **Logic Quantile Rules.** Assume a two-sided logic quantile rule $r := \theta_{\mathsf{lb}} \leq \phi(\psi(X)) \leq \theta_{\mathsf{ub}}$, where X is the output of a model with parameters W over a minibatch of samples. The loss for such a rule is dependent on the statistic ϕ since it is a statistic that aggregates the result of $\psi(X)$. For our experiments, we use the F1 score as this statistic. However, as discussed in Section 3.2, the F1 score is non-differentiable, so we instead use a surrogate F1 loss defined in (Bénédict et al., 2021). Let this loss function be termed L_{F1}. We then define the loss for logic quantile rules as so:

$$l(r; X) = \begin{cases} 0 & X \text{satisfies } r \\ \min\{(\theta_{\mathsf{lb}} - L_{\mathsf{F1}}(\psi(X)))(\theta_{\mathsf{ub}} - L_{\mathsf{F1}}(\psi(X))), 1\} & \text{otherwise} \end{cases} \tag{9}$$

	Tabular classification		Image classification		object detection (Cityscapes-rainy)	
	AUC score	% Reduced	Accuracy	% Reduced	mAP	% Reduced
Our method (fine-tuning whole model)	71.17	30.61±3.5%	80.40	23.28±1.40	26.87	31.3±0.5%
Our method	80.12	38.1±0.0%	80.51	22.17±4.5%	27.01	31.4±0.2%

Table 3: Results for fine-tuning the whole models VS fine-tuning batch normalization layers

Task	**Object Detection**	
Model	EfficientPS	EfficientPS
Rule Class	Neural	Neural
Sample Size	1	1
Dataset **R**	$\text{KITTI}_{\text{train}}$	$\text{KITTI}_{\text{train}}$
Dataset **V**	-	-
Dataset **T**	$\text{KITTI}_{\text{valid}}$	$\text{Cityscapes}_{\text{valid}}$
# Total Rules	252	252
# Selected Rules	252	252
# Total Violations	1,561	7,673
# Rules Violated per Sample	7.80	15.34

Table 4: Results of generating rules and evaluating models against them on KITTI and Cityscapes dataset respectively.

C. Supplementary Experiments

C.1. Ablation study

Note that in the experiments, during the rule-based test time adaptation phase, we only fine-tune the batch normalization layers of the models. The comparison between fine-tuning the whole model and fine-tuning batch-normalization layers has been studied for many state-of-the-art test time adaptation methods, such as (Wang et al., 2020). We therefore also conduct this experiment as our ablation study for the Tabular classification task, Image classification task, and object detection task on the Cityscapes-rainy dataset.

The results are presented in Table 3. This table clearly shows that it is not ideal for fine-tuning the whole model during the rule-based test time adaptation phase since it can significantly drop the model performance by up to 9%. Therefore, same as prior test time adaptation methods, it is reasonable to fine-tune batch normalization layers rather than the entire model for rule-based test time adaptation.

C.2. Additional quantitative results

In Table 4, we present the results of generating and evaluating rules on the KITTI and Cityscapes datasets respectively. By comparing the results of the Cityscapes-rainy dataset against the above results, we can observe that the violations increase over datasets of a different distribution like Cityscapes. This thus implies that the consistency of the model's predictions to the training data worsens over datasets that may be out of distribution. In Table 5, we show the results after performing test-time adaptations over the KITTI and Cityscapes datasets, which still suggests that our methods can reduce more rule violations than other methods without hurting the model performance.

C.3. Additional qualitative results

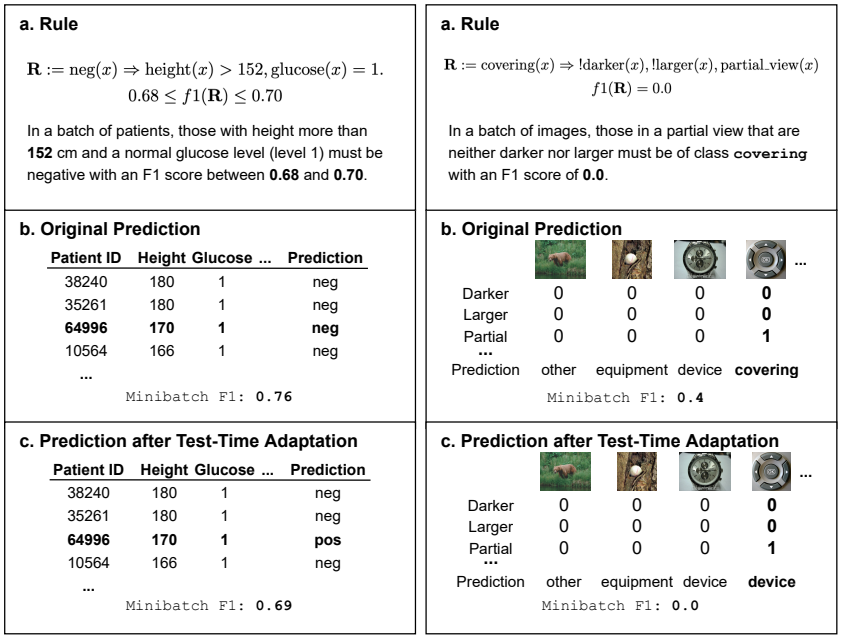
In Figure 4, we show the qualitative results for Tabular Classification and Image Classification tasks.

D. Benchmark Tasks and Models

We consider three different tasks: tabular classification, image classification, and object detection.

Task	Object Detection			
Dataset	KITTI		Cityscapes	
Metric	mAP	% Reduced	mAP	% Reduced
No adapt	44.23	-	56.15	-
BN	44.23	0.0±0.0%	56.15	0.0±0.0%
Tent	44.84	0.2±0.2%	56.24	0.0±0.0%
RPL	44.26	0.4±0.1%	56.67	-0.7±0.2%
CPL	44.62	-3.7±0.0%	57.13	0.2±0.1%
MEMO	-	-	-	-
Ours	44.27	0.1±0.2%	56.91	6.1±1.2%

Table 5: Results of reducing rule violations using different test-time adaptation methods for our three tasks.



(a) Tabular Classification task. (b) Image Classification task.

Figure 4: Additional qualitative results on Tabular Classification and Image Classification task. For each of the three tasks, we show (a) a rule generated by SQRL, (b) a model's prediction that violates the rule, and (c) the model's prediction satisfying the rule on the sample after test-time adaptation. Each of these rules satisfies our three desiderata: they are valid since by construction they hold for 98% of the data; they are expressive enough to find faults in models; and they are generated at scale without human supervision on individual rules.

Tabular Classification. This task concerns classification over tabular data from the Cardiovascular Disease dataset. The dataset has information about 70k patients in the form of 11 attributes (6 numeric attributes and 5 discrete attributes) collected at medical examinations and one binary label to indicate whether the patient has cardiovascular disease. We split the dataset by 65%/15%/20% for training, validation, and testing. We train FT-Transformer (Gorishniy et al., 2021), a state-of-the-art tabular classification model, on the training set. This model is then adapted to the test dataset during the test-time adaptation phase. For learning and evaluating the rules, we use 67 training, 22 validation, and 6 testing samples, each of size 4096.

Image Classification. We define this task using ImageNet-X (Idrissi et al., 2022), a variant of the ImageNet dataset (Deng et al., 2009). It consists of a subset of 12k training samples and all the validation samples from the ImageNet dataset. Each image in the ImageNet-X dataset is also associated with 16 binary human annotations to indicate the characteristics of the object in the image, such as the pose of the object or the lighting of the entire image, over which we can define logic rules. We use a ResNet-34 model (He et al., 2016) trained on the training data from the original ImageNet dataset to predict the 17 metaclasses defined by (Idrissi et al., 2022). We perform test-time adaptation over the validation samples of this dataset. For learning and evaluating the rules, we use 181 training, 79 validation, and 9 testing samples, each of size 256.

Object Detection. This task concerns object detection in self-driving applications over the Cityscapes (Cordts et al., 2016) and KITTI datasets (Geiger et al., 2013). We first consider different scenarios. First, we adapt the EfficientPS model (Mohan & Valada, 2021) pre-trained on training data from the KITTI dataset (Geiger et al., 2013)[4] to the validation samples of that dataset. Additionally, we evaluate our method in distribution shift settings in which the pre-trained EfficientPS model is adapted to two versions of Cityscapes datasets (Cordts et al., 2016), one including the normal scenes and the other augmented with heavy rains (Tremblay et al., 2020). Note that EfficientPS is a panoptic segmentation model which handles multiple tasks such as object detection and semantic segmentation. We only evaluate the object detection component of this model. We use individual samples for training and testing. To learn the rules, we use 855 training samples. We use 200 test samples for testing on the KITTI benchmark, and 500 each for testing on Cityscapes and $\text{Cityscapes}_{\text{rainy}}$.

Time series Imputation. By reusing the pre-process scripts from (Tashiro et al., 2021), we obtain 4000 time-series samples, each of which contains 35 hourly-collected features during a 48-hour stay of one patient at an ICU. We randomly partition those time-series samples into training, validation, and test set with 70-10-20 splits. To evaluate the imputation performance, we input 40% of the non-missing observations in the test split to the imputation model, which then outputs the imputation values for the remaining 60% of non-missing observations. We thus compare the imputed values against the 60% ground-truth non-missing observations to evaluate the model performance.

Sentiment Analysis. The goal of this task is to predict the sentiment of each sentence in the PhraseBank dataset (Malo et al., 2014), which is composed of 3487 sentences in the training set, 387 sentences in the validation set, and 969 sentences in the test set. Note that different from the object detection task where the statistics are collected from the object detection model itself, the statistics used for the sentiment analysis task are obtained by using other pretrained models. We therefore consider neural quantile rules of the following form to guarantee a differentiable loss:

$$
\begin{aligned}
&Y \in \{\mathsf{Positive},\ \mathsf{Negative},\ \mathsf{Neutral}\},\ \phi_1, \phi_2 \in \{\mathsf{fitness\_and\_health}, \mathsf{news\_and\_social\_concern},\ \ldots\},\\
&\theta_{\mathsf{lb}} \leq \phi_1(\mathbf{x}) \leq \theta_{\mathsf{ub}} \Rightarrow \mathbf{y} = Y,\ 1-\delta = 0.98,\ \text{OR}\\
&\theta_{\mathsf{lb},1} \leq \phi_1(\mathbf{x}) \leq \theta_{\mathsf{ub},1}, \theta_{\mathsf{lb},2} \leq \phi_2(\mathbf{x}) \leq \theta_{\mathsf{ub},2} \Rightarrow \mathbf{y} = Y,\ 1-\delta = 0.98,
\end{aligned}
$$

in which, fitness_and_health, news_and_social_concern are features extracted from pretrained models.

E. Discussions

E.1. Capturing statistical correlations.

By constructing the statistical quantile rules, while generating them in a scalable manner, some of the generated rules may capture spurious correlations. However, since these rules are valid over 98% of the data, only those spurious correlations that are also valid in 98% of the data will be captured. Many spurious correlations in practice do not persist in such a large fraction of the data (e.g., dogs can be correlated with being outdoors but fewer than 98% of dogs are photographed outdoors), so these correlations will not be picked up by our approach.

E.2. Redundancy of generated rules.

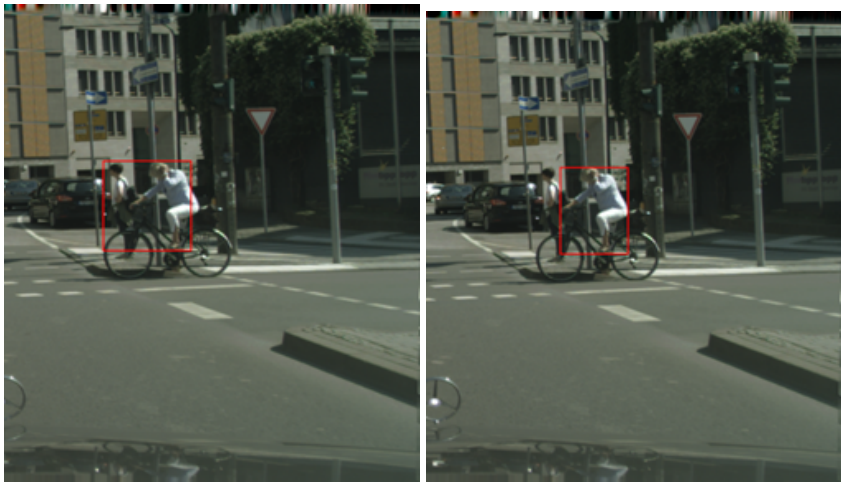
Depending on the schema, SQRL may generate redundant rules. However, this is only the case for logic quantile rules and does not occur in schemas where logic rules are not used, like in object detection, semantic analysis, and imputation tasks. The validated rules from the tabular classification and image classification tasks have up to 2% of the generated rules as redundant. Furthermore, redundant rules don't negatively affect the test time adaptation and only slightly increase the computational costs.

E.3. Other uses of statistical quantile rules.

Providing contexts for mispredictions. Since statistical quantile rules can be used to evaluate models, they can also be used to identify model mispredictions at test time where the ground truth labels are not available. We train a linear classifier using violations of rules as features for a particular sample, and whether or not that sample was mispredicted by a model as the label. A direct interpretation of the weights of the linear classifier indicates the rules that are most correlated with model

[4]The pre-trained model is downloaded from `https://github.com/DeepSceneSeg/EfficientPS`

Figure 5: One example of model mispredictions violating the rule in Equation (10)



(a) Predicted bounding box (b) Ground truth bounding box

Figure 6: One example of the predicted bounding box (see Figure (6a)) is regarded as a correct one according to the IoU score. But this predicted bounding box is still very visually different from the ground truth bounding box (see Figure (6b)) and violates the rule in Equation (11)

errors. For instance, in the object-detection task, 92% of the predictions violating the following rule are errors:

$$\begin{aligned}\mathsf{person}(\mathbf{x}) \Rightarrow & (\mathsf{size}(\mathbf{x}) < 4172 \wedge 302.6 \leq \mathsf{pos}_\mathsf{y}(\mathbf{x}) < 514.1) \vee \\ & (4172 \leq \mathsf{size}(\mathbf{x}) < 13680.5 \wedge 348.76 \leq \mathsf{pos}_\mathsf{y}(\mathbf{x}) < 513.24) \vee \\ & (13680.5 \leq \mathsf{size}(\mathbf{x}) < 47795.75 \wedge 393.44 \leq \mathsf{pos}_\mathsf{y}(\mathbf{x}) < 541.90) \vee \\ & (\mathsf{size}(\mathbf{x}) \geq 47795.75 \wedge 407.45 \leq \mathsf{pos}_\mathsf{y}(\mathbf{x}) < 764.09).\end{aligned} \tag{10}$$

One example of violating the above rule is shown in Figure 5. In this figure, the bounding box predicted to be a person violates the above rule since for the position of the bounding box, the size is too large. This thus explains the reason for this prediction being wrong.

Rules also provide contexts for mispredictions beyond what traditional metrics provide. They can indicate the severity of errors, the frequency of certain kinds of errors over others, and similar rule violations can indicate similar kinds of errors.

Detecting errors that traditional metrics cannot identify. Even for predictions that traditional metrics deem correct, it is possible that the metric is too coarse to sufficiently evaluate that prediction. For example, the mAP metric checks the overlap (IoU score) between a predicted bounding box and a ground truth box. If it finds a ground truth box that has an overlap of more than a threshold (typically 0.5), then it checks the label of the predicted box and says the prediction is correct if the labels match. We show such an example here, where the predicted bounding box is shown in Figure 6a, while

the ground truth is shown in Figure 6b. Intuitively speaking, the predicted bounding box is incorrect since it covers almost two persons in the figure as opposed to the ground truth. According to the metrics used for the self-driving model, this is a correct prediction, though it violates the following quantile rule about pedestrians since nearly all pedestrians are narrower than the bounding box would imply:

$$\begin{aligned}\mathsf{person}(\mathbf{x}) \Rightarrow\ & (\mathsf{y}(\mathbf{x}) <= 433 \wedge 0.70 < \mathsf{aspect\_ratio}(\mathbf{x}) < 4.4) \\ & \vee\ (433 < \mathsf{y}(\mathbf{x}) <= 468 \wedge 0.85 < \mathsf{aspect\_ratio}(\mathbf{x}) < 4.81) \\ & \vee\ (468 < \mathsf{y}(\mathbf{x}) <= 513 \wedge 1.07 < \mathsf{aspect\_ratio}(\mathbf{x}) < 5.0) \\ & \vee\ (\mathsf{y}(\mathbf{x}) > 513 \wedge 1.20 < \mathsf{aspect\_ratio}(\mathbf{x}) < 5.61).\end{aligned} \tag{11}$$