# BAYES-TREX: a Bayesian Sampling Approach to Model Transparency by Example

**Serena Booth[*], Yilun Zhou[*], Ankit Shah, Julie Shah**
[*]Equal Contribution
CSAIL, Massachusetts Institute of Technology
{serenabooth, yilun, ajshah, julie_a_shah}@csail.mit.edu

## Abstract

Post-hoc explanation methods are gaining popularity for interpreting, understanding, and debugging neural networks. Most analyses using such methods explain decisions in response to inputs drawn from the test set. However, the test set may have few examples that trigger some model behaviors, such as high-confidence failures or ambiguous classifications. To address these challenges, we introduce a flexible model inspection framework: BAYES-TREX. Given a data distribution, BAYES-TREX finds in-distribution examples which trigger a specified prediction confidence. We demonstrate several use cases of BAYES-TREX, including revealing highly confident (mis)classifications, visualizing class boundaries via ambiguous examples, understanding novel-class extrapolation behavior, and exposing neural network overconfidence. We use BAYES-TREX to study classifiers trained on CLEVR, MNIST, and Fashion-MNIST, and we show that this framework enables more flexible holistic model analysis than just inspecting the test set. Code and supplemental material are available at https://github.com/serenabooth/Bayes-TrEx.

## 1 Introduction

Debugging, interpreting, and understanding neural networks can be challenging (Doshi-Velez and Kim 2017; Lipton 2018; Odena et al. 2019). Existing interpretability methods include visualizing filters (Zeiler and Fergus 2014), saliency maps (Simonyan, Vedaldi, and Zisserman 2013), input perturbations (Ribeiro, Singh, and Guestrin 2016; Lundberg and Lee 2017), prototype anchoring (Li et al. 2018; Chen et al. 2019), tracing with influence functions (Koh and Liang 2017), and concept quantification (Ghorbani, Wexler, and Kim 2019). While some methods analyze intermediary network components such as convolutional layers (Bau et al. 2017; Olah, Mordvintsev, and Schubert 2017), most methods instead explain decisions based on specific inputs. These inputs are typically selected from the test set, which may lack examples that lead to highly confident misclassifications or ambiguous predictions. Thus, it may be challenging to extract meaningful insights and attain a holistic understanding of model behaviors by using only test set inputs. Finding and analyzing inputs that invoke the gamut of model behaviours would improve *model transparency by example*.

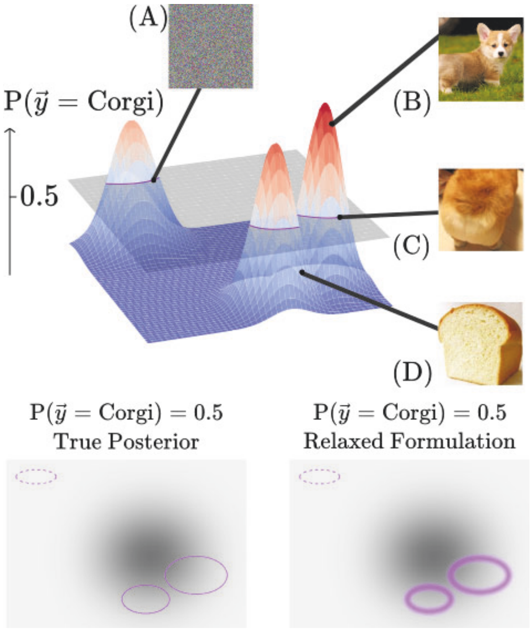
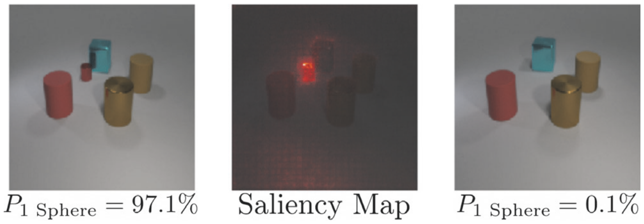

(A)
$P(\vec{y} = \text{Corgi})$
0.5
(B)
(C)
(D)
$P(\vec{y} = \text{Corgi}) = 0.5$
True Posterior
$P(\vec{y} = \text{Corgi}) = 0.5$
Relaxed Formulation

Figure 1: Given a Corgi/Bread classifier, we generate *prediction level sets*, or sets of examples which trigger a target prediction confidence (e.g., $\mathbf{p}_{\text{Corgi}} = \mathbf{p}_{\text{Bread}} = 0.5$). Perturbing an arbitrary image to trigger the target confidence is one way of finding such examples, as shown in (A). However, such examples give little insight into the typical model behavior because they are unrealistic and unlikely. For more insight, BAYES-TREX explicitly considers a data distribution (gray shading on the bottom plots) and finds in-distribution examples in a particular level set (e.g., likely Corgi (B), likely Bread (D), or ambiguous between Corgi and Bread (C)). Bottom left: the classifier level set of $\mathbf{p}_{\text{Corgi}} = \mathbf{p}_{\text{Bread}} = 0.5$ overlaid on the data distribution. Example (A) is unlikely to be sampled by BAYES-TREX due to near-zero density under the distribution, while example (C) is likely to be sampled. Bottom right: Sampling directly from the true posterior is infeasible, so we relax the formulation by "widening" the level set. By using different data distributions and confidences, BAYES-TREX can uncover examples that invoke various model behaviors to improve model transparency.



$P_{1\ \text{Sphere}} = 97.1\%$ Saliency Map $P_{1\ \text{Sphere}} = 0.1\%$

Figure 2: Bayes-TrEx finds a CLEVR scene which is incorrectly classified as containing a sphere. The generated example (left) is composed of only cylinders and cubes, but the classifier is $97.1\%$ confident this scene contains one sphere. The SmoothGrad (Smilkov et al. 2017) saliency map highlights the small red cylinder as the object that is confused for a sphere. When we remove it, the classifier's confidence that the scene contains one sphere drops to $0.1\%$.

To create new examples beyond the scope of the test set, Bayes-TrEx takes a data distribution—either manually defined or learned with generative models—and finds in-distribution examples that trigger various model behaviors. Bayes-TrEx finds examples with target prediction confidences $\mathbf{p}$ by applying Markov-Chain Monte-Carlo (MCMC) methods on the posterior of a hierarchical Bayesian model. For example, Fig. 1 shows a Corgi/Bread classifier. For different $\mathbf{p}$-level set targets (e.g., $\mathbf{p}_{\text{Corgi}} = \mathbf{p}_{\text{Bread}} = 0.5$), Bayes-TrEx can find examples where the model is highly confident in the Corgi class, in the Bread class, or ambiguous between the two. We use Bayes-TrEx to analyze classifiers trained on CLEVR (Johnson et al. 2017) with a manually defined data distribution, as well as MNIST (LeCun and Cortes 2010) and Fashion-MNIST (Xiao, Rasul, and Vollgraf 2017) with data distributions learned by variational autoencoders (VAEs) (Kingma and Welling 2013) or generative adversarial networks (GANs) (Goodfellow et al. 2014).

Bayes-TrEx can aid model transparency by example across several contexts. Each context requires a different data distribution and a specified prediction confidence target. For example, Bayes-TrEx can generate *ambiguous* examples to visualize class boundaries; *high-confidence misclassification* examples to understand failure modes; *novel class* examples to study model extrapolation behaviors; and *high-confidence* examples to reveal model overconfidence (e.g., in domain-adaptation). In all of these use cases, the discovered examples can be further assessed with existing local explanation techniques such as saliency maps (Fig. 2).

The main current alternative to Bayes-TrEx is to inspect a model by using test set examples. As a baseline comparison, we search for highly confident misclassifications and ambiguous examples in the (Fashion-)MNIST and CLEVR test sets. We find few such test set examples meet these constraints, and the majority of these can be attributed to mislabeling in the dataset collection pipeline rather than misclassification by the model. In contrast, Bayes-TrEx consistently finds more highly confident misclassified and ambiguous examples, which enables more flexible and comprehensive model inspection and understanding.

# 2 Related Work

## 2.1 Model Transparency

Broadly, transparency is achieved when a user can develop a correct understanding and expectation of model behavior. One common technique for developing transparency is the test set confusion matrix: this matrix expresses the classifier's tendency of mistaking one class for another. Other transparency methods try to "open" black-box models—for example, by visualizing convolutional filters through optimization (Erhan et al. 2009; Olah, Mordvintsev, and Schubert 2017) or image patches (Bau et al. 2017). Like Bayes-TrEx, other transparency methods communicate model behaviors through examples—for example, with counterfactuals (Antorán et al. 2020; Kenny and Keane 2020) or with student-teacher learning examples (Pruthi et al. 2020).

Some transparency methods aim to explain a model's response to an individual input. For example, saliency maps compute a heat map over the input that represents the importance of each pixel (Simonyan, Vedaldi, and Zisserman 2013; Zeiler and Fergus 2014). Importantly, these input-based methods require a two-stage pipeline: finding interesting inputs → explaining the model responses (e.g., with saliency maps). Current efforts are focused on the second stage with inputs simply retrieved from the test set. To the best of our knowledge, Bayes-TrEx is the first work dedicated to the first stage of finding interesting inputs. The examples uncovered by Bayes-TrEx can be used with any input-based method for further analysis (Fig. 2 and App. K).

## 2.2 Model Testing

TensorFuzz (Odena et al. 2019) is a fuzzing test framework for neural networks which finds inputs that achieve a wide coverage of user-specified constraints. TensorFuzz is similar to Bayes-TrEx in that both methods aim to find examples that elicit certain model behaviors. While TensorFuzz is designed to find *rare* inputs that trigger edge cases such as numerical errors, Bayes-TrEx finds common, in-distribution examples. As such, Bayes-TrEx is more suitable to help humans develop a correct mental model of the classifier. Scenic (Fremont et al. 2019) is a domain-specific language for model testing by generating failure-inducing examples. While Bayes-TrEx is in part inspired by Scenic, its formulation is more flexible.

## 2.3 Natural Adversarial Examples

One Bayes-TrEx use case is uncovering high-confidence classification failures in the data distribution. This idea is related to, but different from, natural adversarial attacks (Zhao, Dua, and Singh 2018). Most adversarial attacks inject crafted high-frequency information to mislead a trained model (Szegedy et al. 2014; Goodfellow, Shlens, and Szegedy 2014; Nguyen, Yosinski, and Clune 2015), but such artifacts are non-existent in natural images. Zhao et al. (2018) instead proposed a method to find *natural* adversarial examples by performing the perturbation in the latent space of a GAN. While this method finds an example which looks like a specific input, Bayes-TrEx finds high-confidence misclassifications in the entire data distribution.

### 2.4 Confidence in Neural Networks

BAYES-TREX can also be used to detect overconfidence in neural networks. An overconfident neural network (Guo et al. 2017) makes many mistakes with disproportionately high confidence. While many approaches aim to address this network overconfidence problem (Blundell et al. 2015; Gal and Ghahramani 2016; Lee et al. 2018; Thulasidasan et al. 2019), BAYES-TREX is complementary to these efforts. Rather than altering the confidence of a neural network, it instead infers examples of a particular confidence. If the model is overconfident, it may return few, if any, samples with ambiguous predictions. At the same time, it may find many misclassifications with high confidence. In our experiments (Sec. 4.8), we discover that the popular adversarial discriminative domain adaptation (ADDA) technique produces a more overconfident model than the baseline.

## 3 Methodology

Given a classifier $f : X \to \Delta_K$ which maps a data point to the probability simplex of $K$ classes, the goal is to find an input $\mathbf{x} \in X$ in a given data distribution $p(\mathbf{x})$ such that $f(\mathbf{x}) = \mathbf{p}$ for some prediction confidence $\mathbf{p} \in \Delta_K$. We consider the problem of sampling from the posterior

$$p(\mathbf{x}|f(\mathbf{x}) = \mathbf{p}) \propto p(\mathbf{x})\, p(f(\mathbf{x}) = \mathbf{p}|\mathbf{x}). \tag{1}$$

A common approach to posterior sampling is to use Markov Chain Monte-Carlo (MCMC) methods (Brooks et al. 2011). However, when the measure of the level set $\{\mathbf{x} : f(\mathbf{x}) = \mathbf{p}\}$ is small or even zero, sampling directly from this posterior using MCMC is infeasible: the posterior being zero everywhere outside of the level set makes it unlikely for a random-walk Metropolis sampler to land on $\mathbf{x}$ with non-zero posterior (Hastings 1970), and the gradient being zero everywhere outside of the level set means that a Hamiltonian Monte Carlo sampler does not have the necessary gradient guidance toward the level set (Neal et al. 2011).

To enable efficient sampling, we relax the formulation by "widening" the level set and accepting $\mathbf{x}$ when $f(\mathbf{x})$ is close to the target $\mathbf{p}$ (Fig. 1). Specifically, we introduce a random vector $\mathbf{u} = [u_1, \ldots, u_K]^T$, distributed as

$$u_i|\mathbf{x} \sim \mathcal{N}\left(f(\mathbf{x})_i, \sigma^2\right), \tag{2}$$

where $\sigma$ is a hyper-parameter.

Instead of directly sampling from Eqn. 1, we can now sample from the new posterior:

$$p(\mathbf{x}|\mathbf{u} = \mathbf{u}^*) \propto p(\mathbf{x})p(\mathbf{u} = \mathbf{u}^*|\mathbf{x}), \tag{3}$$
$$\mathbf{u}^* = \mathbf{p}. \tag{4}$$

The hyper-parameter $\sigma$ controls the peakiness of the relaxed posterior. A smaller $\alpha$ makes it closer to the true posterior and makes the distribution peakier and harder to sample, while a larger $\alpha$ makes it closer to the data distribution $p(\mathbf{x})$ and easier to sample. As $\sigma$ goes to 0, it approaches the true posterior. Formally,

$$\lim_{\sigma\to 0} p(\mathbf{x}|\mathbf{u} = \mathbf{u}^*) = p(\mathbf{x}|f(\mathbf{x}) = \mathbf{p}). \tag{5}$$

While the formulation in Eqn. 2 is applicable to arbitrary confidence $\mathbf{p}$, the dimension of $\mathbf{u}$ is equal to the number of classes, which poses scalability issues for large numbers of classes. However, for a wide range of interesting use cases of BAYES-TREX, we can use the following reductions:

1. Highly confident in class $i$: $\mathbf{p}_i = 1, \mathbf{p}_{\neg i} = 0$. We have
$$u|\mathbf{x} \sim \mathcal{N}\left(f(\mathbf{x})_i, \sigma^2\right), \qquad u^* = 1. \tag{6}$$

2. Ambiguous between class $i$ and $j$: $\mathbf{p}_i = \mathbf{p}_j = 0.5$, $\mathbf{p}_{\neg i,j} = 0$. We have
$$u_1|\mathbf{x} \sim \mathcal{N}\left(|f(\mathbf{x})_i - f(\mathbf{x})_j|, \sigma_1^2\right), \tag{7}$$
$$u_2|\mathbf{x} \sim \mathcal{N}(\min(f(\mathbf{x})_i, f(\mathbf{x})_j) - \max_{k\neq i,j} f(\mathbf{x})_k, \sigma_2^2), \tag{8}$$
$$u_1^* = 0, u_2^* = 0.5. \tag{9}$$
$\sigma_1$ and $\sigma_2$ are hyperparameters.

In addition, most high dimensional data distributions, such as those for images, are implicitly defined by a transformation $g : Z \to X$ from a latent distribution $p(\mathbf{z})$. Consequently, given

$$\mathbf{x} = g(\mathbf{z}), \tag{10}$$
$$\mathbf{u}|\mathbf{z} \sim \mathcal{N}(f(\mathbf{x}), \sigma^2), \tag{11}$$
$$p(\mathbf{z}|\mathbf{u} = \mathbf{u}^*) \propto p(\mathbf{z})p(\mathbf{u} = \mathbf{u}^*|\mathbf{z}), \tag{12}$$

BAYES-TREX samples $\mathbf{z}$ according to Eqn. 12 and reconstruct the example $\mathbf{x} = g(\mathbf{z})$ for model inspection.

## 4 Experiments

### 4.1 Overview

A key strength of BAYES-TREX is the ability to evaluate a classifier on any data distribution $\mathbb{P}_D$, independent of its training distribution $\mathbb{P}_C$. We demonstrate the versatility of BAYES-TREX on four relationships between $\mathbb{P}_D$ and $\mathbb{P}_C$ (Fig. 3). With $\mathbb{P}_C = \mathbb{P}_D$ (Fig. 3(a)), Sec. 4.3 and 4.4 present examples that trigger high and ambiguous model confidence and Sec. 4.5 presents examples that interpolate between two classes. In Sec. 4.6, we consider $\mathbb{P}_D$ with narrower support than $\mathbb{P}_C$ (Fig. 3(b)), where the support of $\mathbb{P}_D$ excludes examples from a particular class. In this case, high-confidence examples—as judged by the classifier—correspond to high-confidence misclassifications. In Sec. 4.7 and 4.8, we analyze the classifier $C$ for novel class extrapolation and domain adaptation behaviors with overlapping or disjoint supports of $\mathbb{P}_C$ and $\mathbb{P}_D$ (Fig. 3(c, d)). Representative results are in the main text; further results are in the appendix.

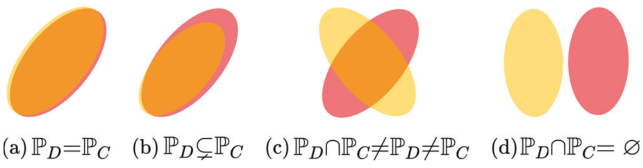

(a) $\mathbb{P}_D$=$\mathbb{P}_C$ (b) $\mathbb{P}_D\subsetneq\mathbb{P}_C$ (c) $\mathbb{P}_D\cap\mathbb{P}_C\neq\mathbb{P}_D\neq\mathbb{P}_C$ (d) $\mathbb{P}_D\cap\mathbb{P}_C= \varnothing$

Figure 3: Different relations between the classifier training distribution ($\mathbb{P}_C$, red) and BAYES-TREX data distribution ($\mathbb{P}_D$, yellow). (a) $\mathbb{P}_C$ and $\mathbb{P}_D$ are equal. (b) The support of $\mathbb{P}_D$ is a subset of that of $\mathbb{P}_C$. (c) $\mathbb{P}_D$ and $\mathbb{P}_C$ have overlapping supports. (d) Supports of $\mathbb{P}_C$ and $\mathbb{P}_D$ are disjoint.

| Model | Dataset | FID |
|---|---|---|
| VAE | MNIST | 72.33 |
| | Fashion-MNIST | 87.89 |
| GAN | MNIST | 11.83 |
| | Fashion-MNIST | 29.44 |

Table 1: Fréchet Inception Distance (FID) for VAE and GAN models trained on the entire dataset. A lower value indicates higher quality. Appx. B presents the statistics for all models.

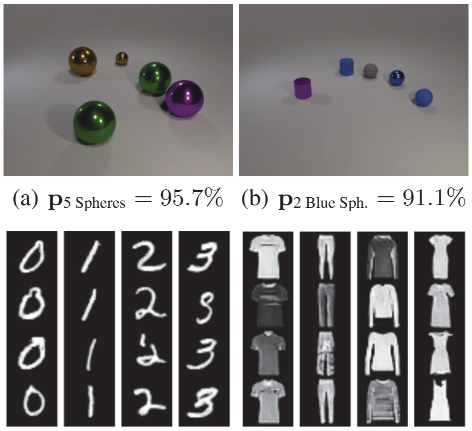
## 4.2 Datasets and Inference Details

We evaluate BAYES-TREX on rendered images (CLEVR) and organic datasets (MNIST and Fashion-MNIST). For all CLEVR experiments, we use the pre-trained classifier distributed by the original authors[1]. The transition kernel uses a Gaussian proposal for the continuous variables (e.g., $x$-position) and categorical proposal for the discrete variables (e.g., color), both centered around and peaked at the current value. For (Fashion-)MNIST experiments, architectures and training details are described in Appx. A. For domain adaptation analysis, we train ADDA and baseline models using the code provided by the authors[2].

CLEVR images are rendered from scene graphs, on which we define the latent distribution $p(\mathbf{z})$. Since the (Fashion-)MNIST groundtruth data distribution is unknown, we estimate it using a VAE or GAN with unit Gaussian $p(\mathbf{z})$. These learned data distribution representations have known limitations, which may affect sample quality (Arora and Zhang 2017). Table 1 lists the Fréchet Inception Distance (FID) (Heusel et al. 2017) for two VAE and GAN models, with the full table in Appx. B. The FID scores show the GANs generate more representative samples than the VAEs.

We consider two MCMC samplers: random-walk Metropolis (RWM) and Hamiltonian Monte Carlo (HMC). We use the former in CLEVR where the rendering function is non-differentiable, and the latter for (Fashion-)MNIST. For HMC, we use the No-U-Turn sampler (Hoffman and Gelman 2014; Neal et al. 2011) implemented in the probabilistic programming language Pyro (Bingham et al. 2018). We choose $\sigma = 0.05$ for all experiments. Alternatively, $\sigma$ can be annealed to gradually reduce the relaxation.

Selecting appropriate stopping criteria for MCMC methods is an open problem. State-of-the-art approaches require a gold standard inference algorithm (Cusumano-Towner and Mansinghka 2017) or specific posterior distribution properties, such as log-concavity (Gorham and Mackey 2015). As neither of these requirements are met for our domains, we select stopping criteria based on heuristic performance and cost of compute (Appx. I). CLEVR requires GPU-intensive rendering, so we stop after 500 samples. (Fashion-)MNIST samples are cheaper to generate, so we stop after 2,000 samples. Empirically, we find each sampling step takes 3.75 seconds for CLEVR, 1.18s for MNIST, and 1.96s for Fashion-MNIST, all on a single NVIDIA GeForce 1080 GPU.



(a) $\mathbf{p}_{5\text{ Spheres}} = 95.7\%$ (b) $\mathbf{p}_{2\text{ Blue Sph.}} = 91.1\%$

(c) MNIST (d) Fashion-MNIST

Figure 4: High-confidence samples, which pass the smoke test for CLEVR, MNIST, and Fashion-MNIST T-shirt, trousers, pullover, and dress. More examples in Appx. C.

## 4.3 High Confidence

As an initial smoke test, we evaluate BAYES-TREX by finding highly confident examples. (Fashion-)MNIST data distributions are learned by GAN. Fig. 4 depicts samples on the three datasets. Additional examples are in Appx. C.

## 4.4 Ambiguous Confidence

Next, we find ambiguous (Fashion-)MNIST examples for which the classifier has similar prediction confidence between two classes, using data distributions learned by a VAE. Fig. 5 shows ambiguous examples from each pair of classes (e.g. 0v1, 0v2, ..., 8v9). Note the examples presented are ambiguous from the classifier's perspective, though some may be readily classified by a human. Not all pairs result in successful sampling: for example, we were unable to find an ambiguous example with equal prediction confidence between the visually dissimilar classes 0 and 7. These ambiguous examples are useful for visualizing and understanding class boundaries; Appx. D presents a supporting class boundary latent space visualization. *Blended* ambiguous examples have previously been shown to be useful for data augmentation (Tokozume, Ushiku, and Harada 2018). While these generated ambiguous examples may be similarly useful, we leave this exploration to future work.

BAYES-TREX can also find examples which are ambiguous across more than two classes; Fig. 6 presents samples that are equally ambiguous across all 10 MNIST classes. All these images appear to be very blurry and not very realistic. This is intuitive: even for a human, it would be hard to write a digit in such a way that it is equally unrecognizable across all 10 classes. Details about the sampling formulation and visualizations are presented in Appx. D.

In general, for ambiguous examples, we observed only rare successes with data distributions learned by a GAN, which generates sharper and more visually realistic images than a VAE. There are two potential explanations:

1. GAN-distributions prevent efficient MCMC sampling.

[1] https://github.com/facebookresearch/clevr-iep

[2] https://github.com/erictzeng/adda

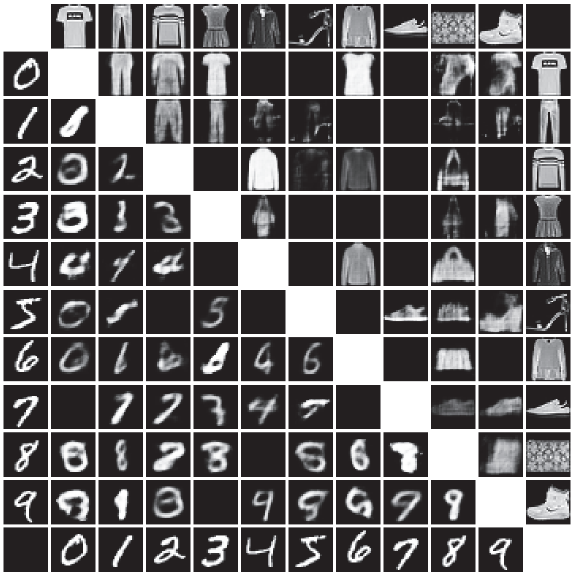


Figure 5: Each entry of the matrix is an ambiguous MNIST or Fashion-MNIST example for the classes on its row and column. Blacked-out cells indicate sampling failures. Examples on the outermost edges of the matrix are representative examples from each class (e.g., 0-9 for MNIST).

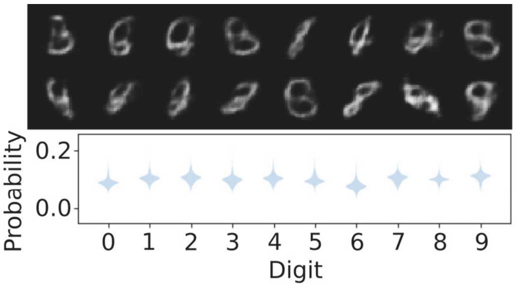


Figure 6: Samples of uniformly ambiguous predictions.

2. The classifier rarely makes ambiguous predictions on sharp and realistic images.

To experimentally evaluate the second explanation, we train a classifier to be consistently ambiguous between class $i$ and $i+1$ for an image of digit $i$ (wrapping around at $10 = 0$) using the following KL-divergence loss:

$$l(y, f(\mathbf{x})) = \mathbb{KL}(\mathbf{p}_y, f(\mathbf{x})), \tag{13}$$

$$\mathbf{p}_{y,i} = \begin{cases} 0.5 & i = y \text{ or } i = (y+1) \bmod 10, \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

Using this classifier, we sample ambiguous examples for 0v1, 1v2, ..., 9v0. Sampling succeeds for all ten pairs, even when using the same GAN model that rarely succeeded in the prior experiment. Fig. 7 presents the 0v1 samples and predicted confidence by this modified classifier, and the remaining pairs are visualized in Appx. E. Given this sampling success, we conclude that the second explanation is correct.

BAYES-TREX is also unable to generate ambiguous examples for CLEVR with the manually defined data distribution. Given that the pre-trained classifier only achieves ≈60% accuracy, the result suggests that the model is likely overconfident. Indeed, this has previously been observed in similar settings (Kim, Ricci, and Serre 2018).

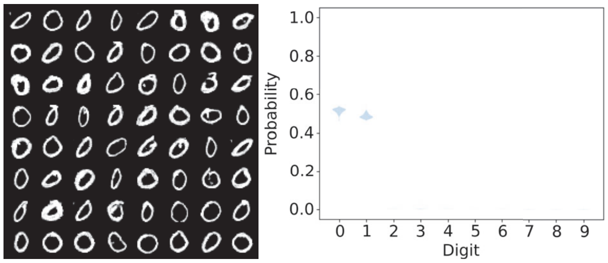


Figure 7: 0v1 ambiguous samples and confidence plot with the GAN distribution and always ambiguous classifier. This shows successful sampling and supports hypothesis 2.

## 4.5 Confidence Interpolation

BAYES-TREX can find examples that interpolate between classes. In Fig. 8, we show MNIST samples which interpolate from $(P_8 = 1.0, P_9 = 0.0)$ to $(P_8 = 0.0, P_9 = 1.0)$ and Fashion-MNIST samples from $(P_{\text{T-shirt}} = 1.0, P_{\text{Trousers}} = 0.0)$ to $(P_{\text{T-shirt}} = 0.0, P_{\text{Trousers}} = 1.0)$ over intervals of 0.1, with a VAE-learned data distribution.

The interpolation between two very different classes reveal insights into the model behavior. For example, the interpolation from 8 to 9 generally shrinks the bottom circle toward a stroke, which is the key difference between digits 8 and 9. For Fashion-MNIST, the presence of two legs is important for trousers classification, even appearing in samples with $(\mathbf{p}_{\text{T-shirt}} = 0.9, \mathbf{p}_{\text{Trousers}} = 0.1)$ (second column). By contrast, a wider top and the appearance of sleeves are important properties for T-shirt classification. These two trends result in most of the interpolated samples having a short sleeve on the top and two distinct legs on the bottom.

## 4.6 High-Confidence Failures

With neural networks being increasingly used for high-stakes decision making, high-confidence failures are one area of concern, as these failures may go unnoticed. BAYES-TREX can find such failures. Specifically, if the data distribution (Fig. 3(b)) does *not* include a particular class, then the resulting high-confidence examples correspond to high-confidence *misclassifications* for that class. For example, in Fig. 9(a), the CLEVR classifier is highly confident that there is one cube though there is no cube in the image. In App. K, the saliency map for Fig. 9(a) reveals that classifier mistakes the front shiny red cylinder for a cube. Removing this cylinder causes the confidence to drop to 29.0%. In addition, such high-confidence failures can also be used for data augmentation to increase network reliability (Fremont et al. 2019).

For (Fashion-)MNIST, a GAN is trained on all data sans a single class, resulting in the learned data distribution excluding the given class. Figs. 9(c) and 9(d) depict high-confidence misclassifications for digits 0-4 in MNIST and sandal, shirt, sneaker, bag, and ankle boot in Fashion-

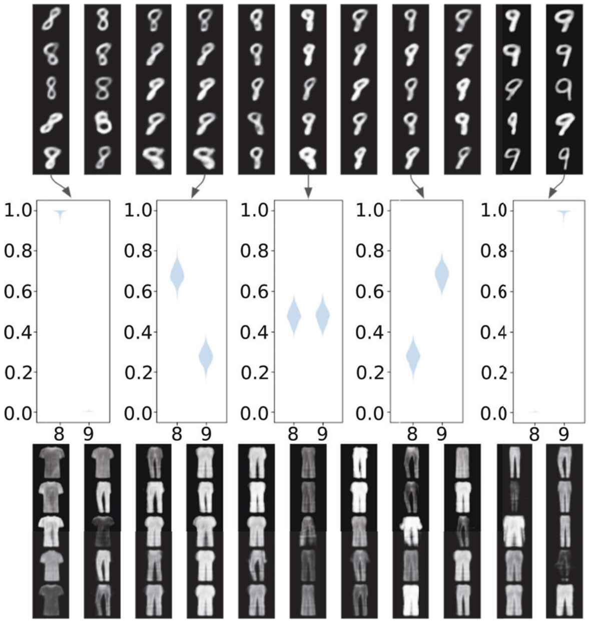


Figure 8: Confidence interpolation between digit 8 and 9 for MNIST and between T-shirt and trousers for Fashion-MNIST. Each of the 11 columns show samples of confidence ranging from [$\mathbf{p}_{\text{class a}} = 1.0$, $\mathbf{p}_{\text{class b}} = 0.0$] (left) to [$\mathbf{p}_{\text{class a}} = 0.0$, $\mathbf{p}_{\text{class b}} = 1.0$] (right), with an interval of 0.1. Some confidence plots for MNIST are shown in the middle.

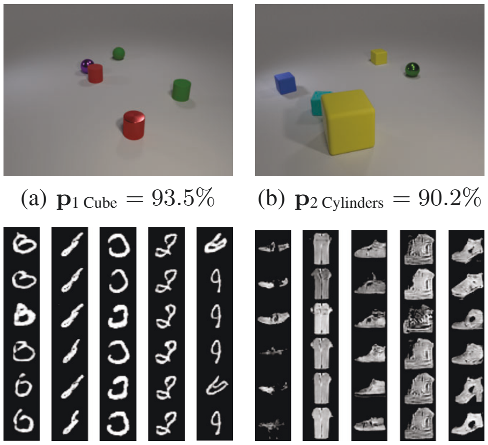
MNIST, respectively. By evaluating these examples, we can assess how well human-aligned a classifier is. For example, for MNIST, some thin 8s are classified as 1s and particular styles of 6s and 9s are classified as 4s. These results seem intuitive, as a human might make these same mistakes. Likewise, for Fashion-MNIST, most failures come from semantically similar classes, e.g. sneaker ⟷ ankle boot. Less intuitively, however, chunky shoes are likely to be classified as bags. Additional visualizations are presented in Appx. F.

### 4.7 Novel Class Extrapolation

It is important to understand the novel class extrapolation behavior of a model before deployment. For example, during training an autonomous vehicle might learn to safely operate around pedestrians, cyclists, and cars. But can we predict how the vehicle will behave when it encounters a novel class, like a tandem bicycle? BAYES-TREX can be used to understand such behaviors by sampling high-confidence examples with a data distribution that contains novel classes, while excluding the true target classes (Fig. 3(c, d)).

For CLEVR, we add a novel cone object to the data distribution and remove the existing cube from it. We sample images that the classifier is confident to include cubes, shown in Fig. 10 (a, b). A saliency map analysis in Appx. K confirms that the classifier indeed mistakes these cones for cubes. In Appx. G, we assess CLEVR's novel class extrapolation for cylinders and spheres, and similarly show the model readily confuses cones for these classes as well.

For MNIST and Fashion-MNIST, we train the respective



(a) $\mathbf{p}_{1\text{ Cube}} = 93.5\%$ (b) $\mathbf{p}_{2\text{ Cylinders}} = 90.2\%$

(c) MNIST (d) Fashion-MNIST

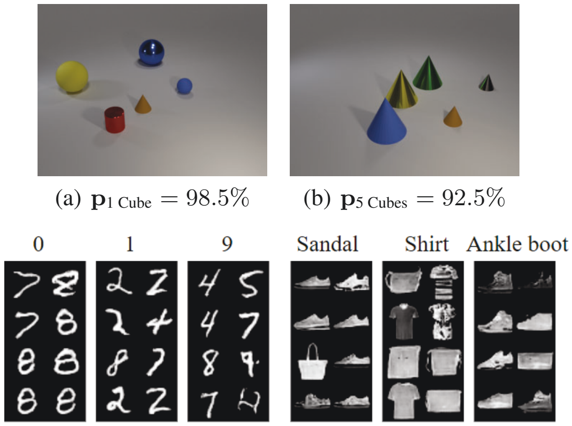
Figure 9: High confidence classification failures. (a): CLEVR, 1 Cube. Note that no cube is present in the sample. (b): CLEVR, 2 Cylinders—again, containing no cylinders. (c) MNIST failures for digits 0-4. 0s are composed of 6s; 1s of 8s; 2s of 0s, and so on. (d) Fashion-MNIST failures for sandal, shirt, sneaker, bag, and ankle boot. Additional examples are presented in Appx. F.

classifiers on digits 0, 1, 3, 6, 9 and pullover, dress, sandal, shirt and ankle boot classes. We train GANs using only the excluded classes (e.g., digits 2, 4, 5, 7, 8 for MNIST). Using these GANs, we find examples where the classifier has high prediction confidence, as shown in Fig. 10 (c, d). For MNIST, there are few reasonable extrapolation behaviors, most likely due to the visual distinctiveness between digits. By comparison, some Fashion-MNIST extrapolations are expected, such as confusing the unseen sneaker class for sandals and ankle boots. However, the classifier also confidently mistakes various styles of bags as sandals, shirts, and ankle boots. App. G contains additional visualizations.

### 4.8 Domain Adaptation

Finally, we use BAYES-TREX to analyze domain adaptation behaviors. We reproduce the SVHN (Netzer et al. 2011) → MNIST experiment studied by Tzeng, et al. (2017). We train two classifiers, a baseline classifier on labeled SVHN data only, and the ADDA classifier on labeled SVHN data and unlabeled MNIST data. Indeed, domain adaptation improves classification accuracy: 61% for the baseline classifier on MNIST vs. 71% for the ADDA classifier.

But is this the whole story? To study model performance in the high-confidence range, we use BAYES-TREX to generate high-confidence examples for both classifiers with the MNIST data distribution learned by GAN, as shown Fig. 11. It appears the ADDA model makes *more* mistakes in these images—for example, in the 2nd column in Fig. 11(b), all images where the classifier is highly confident to be 1 are actually 0s. To further study this, we hand-label 10 images per class and compute the classifier accuracy on them. Table 3 shows the accuracy per digit class, as well as the over-



(a) $\mathbf{p}_{\text{1 Cube}} = 98.5\%$ (b) $\mathbf{p}_{\text{5 Cubes}} = 92.5\%$

(c) MNIST (d) Fashion-MNIST

Figure 10: Novel class extrapolation examples. (a, b): For CLEVR, the novel cone objects are mistaken for cubes. (c, d): For (Fashion-)MNIST, we train classifiers on subsets of the data (digits 0, 1, 3, 6, 9 and pullover, dress, sandal, shirt, and ankle boot), and train GANs with the excluded data. Samples for which the classifier is highly confident ($\approx 99\%$) in several target classes are shown (e.g., targets 0, 1, and 9 for MNIST). Additional examples are presented in Appx. G.

all accuracy. This analysis confirms the baseline model is more accurate than the ADDA model on these samples, suggesting that ADDA is more overconfident than the baseline. While this result does not contradict the higher overall accuracy of ADDA, it does caution against deploying such domain adaptation models without further inspection and confidence calibration assessment.

## 4.9 Quantitative Evaluation

We quantitatively evaluate the quality of BAYES-TREX samples by assessing whether the classifier's prediction confidence matches the specified target on the generated examples. Table 2 presents the mean and standard deviation of the confidence on a selection of representative settings, and Appx. I lists the full set of such evaluations. The prediction confidences are tightly concentrated around the targets, demonstrating sampler success.

## 4.10 Test-Set Comparison

Standard model evaluations are typically performed on the test set. While inspecting test set examples is not an apples-to-apples comparison for all BAYES-TREX use cases (e.g. domain adaptation), we study the comparable ones.

**Ambiguous Confidence** We find ambiguous examples in the (Fashion-)MNIST datasets where the classifier has confidence in $[40\%, 60\%]$ for two classes. Out of 10,000 test examples on each dataset, we find only 12 MNIST examples across 10 class pairings, and 162 Fashion-MNIST examples across 12 pairings, as shown in Fig. 12. By comparison, BAYES-TREX found ambiguous examples for 38 MNIST pairings and 28 Fashion-MNIST pairings (cf. Fig. 5).

| Test | Data | Target | Prediction Confidence |
|---|---|---|---|
| A | M | $\mathbf{p}_4 = 1$ | $1.00 \pm .01$ |
| | F | $\mathbf{p}_{\text{Coat}} = 1$ | $0.98 \pm .02$ |
| | C | $\mathbf{p}_{\text{2 Blue Sph.}} = 1$ | $0.89 \pm .25$ |
| B | M | $\mathbf{p}_1 = \mathbf{p}_7 = 0.5$ | $0.49, 0.49 \pm .02, .03$ |
| | F | $\mathbf{p}_0 = \mathbf{p}_3 = 0.5$ | $0.48, 0.48 \pm .02, .02$ |
| C | M | $\mathbf{p}_8, \mathbf{p}_9 = 0.6, 0.4$ | $0.58, 0.37 \pm .04, .04$ |
| | F | $\mathbf{p}_0, \mathbf{p}_1 = 0.2, 0.8$ | $0.17, 0.79 \pm .04, .04$ |
| D | M | $\mathbf{p}_8 = 1$ | $0.98 \pm .02$ |
| | F | $\mathbf{p}_{\text{Bag}} = 1$ | $0.97 \pm .03$ |
| | C | $\mathbf{p}_{\text{1 Cube}} = 1$ | $0.93 \pm .06$ |
| E | M | $\mathbf{p}_6 = 1$ | $1.00 \pm .01$ |
| | F | $\mathbf{p}_{\text{Sandal}} = 1$ | $1.00 \pm .01$ |
| | C | $\mathbf{p}_{\text{1 Cylinder}} = 1$ | $0.96 \pm .03$ |
| F | M | $\mathbf{p}_5 = 1$ | $1.00 \pm .01$ |

Table 2: Mean and standard deviation of the sample prediction confidences. Tests are A: high confidence, B: ambiguous, C: interpolation, D: misclassifications, E: novel classes, and F: domain adaptation. Data are M: MNIST, F: Fashion, C: CLEVR. Fashion-MNIST classes 0-9 correspond to T-shirt, trousers, pullover, dress, coat, sandal, shirt, sneaker, bag and ankle boot. See Appx. I for full statistics.

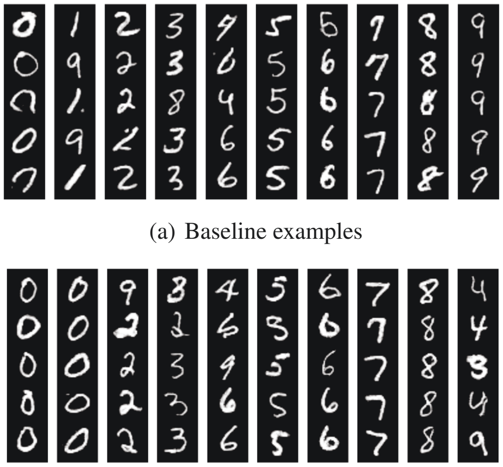
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | All |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Base | 1 | .6 | 1 | .7 | .5 | .9 | .9 | .7 | 1 | .7 | .8 |
| DA | .9 | 0 | .8 | .9 | .2 | 1 | .8 | 1 | 1 | .6 | .72 |

Table 3: Per-digit and overall accuracy among high-confidence MNIST samples for the baseline and domain adaptation (DA) models. While DA has higher overall accuracy (0.71 vs. 0.61), it performs *worse* on high-confidence samples (0.72 vs. 0.80). This suggests overconfidence.

**High-Confidence Failures** We collect and inspect highly confident test set misclassifications (confidence $\geq 85\%$). For CLEVR, out of $15,000$ test images, the baseline discovers between 0 and 15 examples for each target. Notably, there are no 2-cylinder misclassifications in the test set, but BAYES-TREX successful generated some (Fig. 9(b)).

From the 10,000 test examples in (Fashion-)MNIST, 84 MNIST images and 802 Fashion-MNIST images were confidently misclassified. Upon closer inspection, however, we find that the a large fraction of the failures are actually due to *mislabeling*, rather than misclassification. We manually relabel all 84 MNIST misclassifications and ten Fashion-MNIST misclassifications per class, except for the trousers class which only has 3 misclassifiations. We find that the 60 out of 84 MNIST images 42 out of 93 Fashion-MNIST images are mislabeled, rather than misclassified.

Table 4 gives detailed statistics of the number of genuinely misclassified examples. Given the scene graph data representation, all CLEVR misclassifications are genuine. Table 5 visualizes some misclassified vs. mislabeled images, with additional classes in Appx. J. Identifying mislabeled examples may be useful for correcting the dataset, but is not for our task of model understanding.



(a) Baseline examples

(b) ADDA examples

Figure 11: High confidence examples for baseline and ADDA models, classes 0 to 9, showing more misclassifications for the ADDA model. More examples in Appx. H.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CLEVR | Cls. | 1 Sph. | 1 Cube | 1 Cyl. | 2 Cyl. | | | | | | |
| 28/28 | # | 5 | 8 | 15 | 0 | | | | | | |
| MNIST | Cls. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 24/84 | # | 3 | 3 | 0 | 5 | 3 | 1 | 3 | 4 | 0 | 2 |
| Fashion | Cls. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 51/93 | # | 2 | 0 | 9 | 4 | 9 | 1 | 3 | 2 | 1 | 10 |

Table 4: Number of *genuine* high-confidence misclassifications from test sets. Counts for CLEVR and MNIST are for the entire test set; counts for Fashion-MNIST are for ten random high-confidence misclassifications per class, except for trousers which only has 3 total misclassifications.

**Novel Class Extrapolation** In Sec. 4.7 analysis, we find that the model mistakes some bags for ankle boots. Interestingly, this propensity is not evident from test set evaluations: the test set confusion matrix in Appx. J shows that no bags are misclassified as ankle boots. This provides further evidence of the value of holistic evaluations with BAYES-TREX, beyond standard test set evaluations.

## 5 Discussion

BAYES-TREX is a Bayesian inference approach for generating examples that trigger specified target predictions and so provide insight into model behaviors. These examples can be further analyzed with downstream interpretability methods (Fig. 2 and Appx. K). To make BAYES-TREX easier for model designers to use, future work should develop methods to cluster and visualize trends in the generated examples, as well as to estimate coverage of the level set.

For organic data, the underlying data distributions can be learned with VAEs or GANs. These have known limitations in sample diversity (Arora and Zhang 2017) and are computationally expensive to train, especially for high resolution images. In principle, BAYES-TREX is agnostic to the distribution learner form and can benefit from future research in this area. In practice, BAYES-TREX is currently limited to low dimensional latent spaces, as applying MCMC sampling to high dimensional latent spaces is an open problem.

Finally, while we analyzed only classification models with BAYES-TREX, it also has the potential for analyzing structured prediction models such as machine translation or robotic control. For these domains, dependency among outputs would need to be explicitly taken into account. We plan to extend BAYES-TREX to these areas in the future.

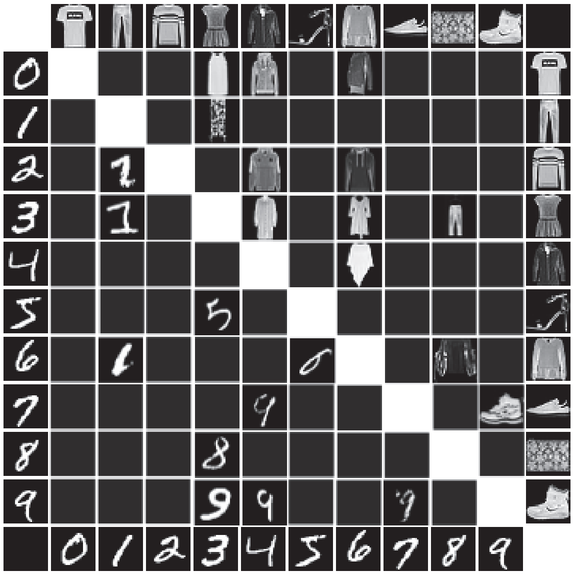


Figure 12: Ambiguous examples from the (Fashion-)MNIST test sets. Compared to those found by BAYES-TREX in Fig. 5, test set examples have much poorer coverage.

| Class | Cause | Images |
|---|---|---|
| 0 | Misclass. | |
| | Mislabeled | |
| 1 | Misclass. | |
| | Mislabeled | |
| 2 | Misclass. | ∅ |
| | Mislabeled | |
| Trouser | Misclass. | ∅ |
| | Mislabeled | |
| Bag | Misclass. | |
| | Mislabeled | |

Table 5: High confidence misclassifications from the test set. The majority are due to incorrect ground truth labels, not classifier failures. Full table of all classes in Appx. J.

## Acknowledgements

The authors would like to thank: Alex Lew, Marco Cusumano-Towner, and Tan Zhi-Xuan for their insights on how to formulate this inference problem and use probabilistic programming effectively; Christian Muise and Hendrik Strobelt for helpful early discussions; and James Tompkin and the anonymous reviewers for comments on the draft. SB is supported by an NSF GRFP.

## Ethics Statement

BAYES-TREX has potential to allow humans to build more accurate mental models of how neural networks make decisions. Further, BAYES-TREX can be useful for debugging, interpreting, and understanding networks—all of which can help us build *better*, less biased, increasingly human-aligned models. However, BAYES-TREX is subject to the same caveats as typical software testing approaches: the absence of exposed bad samples does not mean the system is free from defects. One concern is how system designers and users will interact with BAYES-TREX in practice. If BAYES-TREX does not reveal degenerate examples, these stakeholders might develop inordinate trust (Lee and See 2004) in their models.

Additionally, one BAYES-TREX use case is to generate examples for use with downstream local explanation methods. As a community, we know many of these methods can be challenging to understand (Olah, Mordvintsev, and Schubert 2017; Nguyen, Yosinski, and Clune 2019), misleading (Adebayo et al. 2018; Kindermans et al. 2019; Rudin 2019), or susceptible to adversarial attacks (Slack et al. 2020). In human-human interaction, even nonsensical explanations can increase compliance (Langer, Blank, and Chanowitz 1978). As we build post-hoc explanation techniques, we must evaluate whether the produced explanations help humans moderate trust and act appropriately—for example, by overriding the model's decisions.

## References

Adebayo, J.; Gilmer, J.; Muelly, M.; Goodfellow, I.; Hardt, M.; and Kim, B. 2018. Sanity checks for saliency maps. In *NeurIPS*, 9505–9515.

Antorán, J.; Bhatt, U.; Adel, T.; Weller, A.; and Hernández-Lobato, J. M. 2020. Getting a clue: A method for explaining uncertainty estimates. *arXiv preprint arXiv:2006.06848* .

Arora, S.; and Zhang, Y. 2017. Do GANs actually learn the distribution? An empirical study. *arXiv:1706.08224* .

Bau, D.; Zhou, B.; Khosla, A.; Oliva, A.; and Torralba, A. 2017. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6541–6549.

Bingham, E.; Chen, J. P.; Jankowiak, M.; Obermeyer, F.; Pradhan, N.; Karaletsos, T.; Singh, R.; Szerlip, P.; Horsfall, P.; and Goodman, N. D. 2018. Pyro: Deep Universal Probabilistic Programming. *JMLR* .

Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; and Wierstra, D. 2015. Weight uncertainty in neural networks. In *ICML*, 1613–1622.

Brooks, S.; Gelman, A.; Jones, G.; and Meng, X.-L. 2011. *Handbook of markov chain monte carlo*. CRC press.

Chen, C.; Li, O.; Tao, D.; Barnett, A.; Rudin, C.; and Su, J. K. 2019. This looks like that: deep learning for interpretable image recognition. In *NeurIPS*, 8928–8939.

Cusumano-Towner, M.; and Mansinghka, V. K. 2017. AIDE: An algorithm for measuring the accuracy of probabilistic inference algorithms. In *NeurIPS*.

Doshi-Velez, F.; and Kim, B. 2017. Towards a rigorous science of interpretable machine learning. *arXiv* .

Erhan, D.; Bengio, Y.; Courville, A.; and Vincent, P. 2009. Visualizing higher-layer features of a deep network. *University of Montreal* 1341(3): 1.

Fremont, D. J.; Dreossi, T.; Ghosh, S.; Yue, X.; Sangiovanni-Vincentelli, A. L.; and Seshia, S. A. 2019. Scenic: a language for scenario specification and scene generation. In *PLDI*.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 1050–1059.

Ghorbani, A.; Wexler, J.; and Kim, B. 2019. Automating interpretability: Discovering and testing visual concepts learned by neural networks. *arXiv:1902.03129* .

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NeurIPS*, 2672–2680.

Goodfellow, I.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv:1412.6572* .

Gorham, J.; and Mackey, L. 2015. Measuring sample quality with Stein's method. In *NeurIPS*, 226–234.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On calibration of modern neural networks. In *ICML*.

Hastings, W. K. 1970. Monte Carlo sampling methods using Markov chains and their applications. In *Bibliometrika*, 97–109. Oxford University Press.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 6626–6637.

Hoffman, M. D.; and Gelman, A. 2014. The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *JMLR* 15(1): 1593–1623.

Johnson, J.; Hariharan, B.; van der Maaten, L.; Fei-Fei, L.; Zitnick, C. L.; and Girshick, R. 2017. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *CVPR*.

Kenny, E. M.; and Keane, M. T. 2020. On generating plausible counterfactual and semi-factual explanations for deep learning. *arXiv preprint arXiv:2009.06399* .

Kim, J.; Ricci, M.; and Serre, T. 2018. Not-So-CLEVR: learning same–different relations strains feedforward neural networks. *Interface focus* 8(4): 20180011.

Kindermans, P.-J.; Hooker, S.; Adebayo, J.; Alber, M.; Schütt, K. T.; Dähne, S.; Erhan, D.; and Kim, B. 2019. The (un) reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, 267–280. Springer.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv:1312.6114* .

Koh, P. W.; and Liang, P. 2017. Understanding black-box predictions via influence functions. In *ICML*.

Langer, E. J.; Blank, A.; and Chanowitz, B. 1978. The mindlessness of ostensibly thoughtful action: The role of "placebic" information in interpersonal interaction. *Journal of personality and social psychology* 36(6): 635.

LeCun, Y.; and Cortes, C. 2010. MNIST handwritten digit database. Accessed 2021-03-08. URL http://yann.lecun.com/exdb/mnist/.

Lee, J. D.; and See, K. A. 2004. Trust in automation: Designing for appropriate reliance. *Human factors* 46(1): 50–80.

Lee, K.; Lee, H.; Lee, K.; and Shin, J. 2018. Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples. In *International Conference on Learning Representations*.

Li, O.; Liu, H.; Chen, C.; and Rudin, C. 2018. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Lipton, Z. C. 2018. The mythos of model interpretability. *Queue* 16(3): 31–57.

Lundberg, S. M.; and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In *NeurIPS*, 4765–4774.

Maaten, L. v. d.; and Hinton, G. 2008. Visualizing data using t-SNE. *JMLR* 9(Nov): 2579–2605.

Neal, R. M.; et al. 2011. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo* 2(11): 2.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Nguyen, A.; Yosinski, J.; and Clune, J. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 427–436.

Nguyen, A.; Yosinski, J.; and Clune, J. 2019. Understanding neural networks via feature visualization: A survey. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer.

Odena, A.; Olsson, C.; Andersen, D.; and Goodfellow, I. 2019. TensorFuzz: Debugging Neural Networks with Coverage-Guided Fuzzing. In *ICML*, 4901–4911. Long Beach, California, USA.

Olah, C.; Mordvintsev, A.; and Schubert, L. 2017. Feature Visualization. *Distill* doi:10.23915/distill.00007. Https://distill.pub/2017/feature-visualization.

Pruthi, D.; Dhingra, B.; Soares, L. B.; Collins, M.; Lipton, Z. C.; Neubig, G.; and Cohen, W. W. 2020. Evaluating Explanations: How much do explanations from the teacher aid students? *arXiv preprint arXiv:2012.00893* .

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. "Why should I trust you?" Explaining the predictions of any classifier. In *KDD*, 1135–1144.

Rudin, C. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1(5): 206–215.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2013. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *CoRR* abs/1312.6034.

Slack, D.; Hilgard, S.; Jia, E.; Singh, S.; and Lakkaraju, H. 2020. How can we fool LIME and SHAP? Adversarial Attacks on Post hoc Explanation Methods. *AAAI Conference on Artificial Intelligence, Ethics, and Society (AIES)* .

Smilkov, D.; Thorat, N.; Kim, B.; Viégas, F.; and Wattenberg, M. 2017. Smoothgrad: removing noise by adding noise. *arXiv:1706.03825* .

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR*.

Thulasidasan, S.; Chennupati, G.; Bilmes, J. A.; Bhattacharya, T.; and Michalak, S. 2019. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *NeurIPS*, 13888–13899.

Tokozume, Y.; Ushiku, Y.; and Harada, T. 2018. Between-class learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5486–5494.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *CVPR*.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv:1708.07747* .

Zeiler, M. D.; and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, 818–833. Springer.

Zhao, Z.; Dua, D.; and Singh, S. 2018. Generating Natural Adversarial Examples. In *ICLR*.