A Critical Analysis of BayLIME

Tatsuo Okubo, Vanessa Moody, Ruth Kohen, Jon Huml [*]

December 2021

1 Problem Statement

Increases in computational power, floods of cheap sensors or data extraction methods, and sophisticated modeling has launched the era of machine and deep learning. When we find typical measures of model success inadequate, the answer has been clear: more parameters, more data, better engineering. This feedback loop has made highly overparameterized networks too high-performing to ignore, and such models now have relevance across nearly all disciplines, ranging from medicine to law, where the decisions can be a matter of life or death. We are then faced with a dilemma: drown in a sea of data and continue to make decisions based on heuristics and hunches **or** incorporate these models into our workflows.

Of course, these high-performing black boxes come at a price. A major obstacle to the widespread use of such models is their lack of transparency. Users of such models, such as doctors, should hesitate to adopt a model's conclusions if the basis of the decision is not clear, or else risk making judgements based on mere happenstance. This has given rise to the field of explainable AI (XAI), an attempt to provide human-interpretable descriptions of relevant features upon which a model makes a classification. Challenges of XAI include makings its explanations robust, exhaustive, and faithful to the actual workings of the model it is trying to explain. The paper we have chosen for this project [2] contributes to this field of XAI by expanding on LIME (Local Interpretable Model-Agnostic Explanation) [7] through the addition of a Bayesian framework.

2 Context and Scope

The classical field of statistics has made great contributions to scientific studies in the past century, not only in the hard sciences, but in public policy, law, or the practice of medicine. The workflow here, though not necessarily simple, is at least straightforward: modelers make a hypothesis, gather data, and reject or

[*]Interested in research

fail to reject said hypothesis based on the theoretical guarantees of the hypothesis testing method. Modern deep and machine learning, in contrast, presents new paradigm shifts: first, where the modeler may or may not have a specific hypothesis, which we term the exploratory regime. In the second paradigm shift, the space of hypotheses is too large to possibly explore, which we term the machine domain. As an example, platforms like Youtube and Facebook live in the machine domain. Content flow is too large for any number of humans to sort in a reasonable amount of lifetimes, and thus curation is left to deep and machine learning algorithms. Similarly, one of the miracles of the late 20th and early 21st century is the Human Genome project, where the feature space across all humans will be even more complex. Of course, we disregard the utility of such data at our own risk, yet we must also balance the influx of data with the inherent capacity of our modeling techniques. We will have to model the data if we wish to make use of it, which necessarily entails some degree of compression. We compress thousands of pixels to binary labels, or thousands or millions of numbers to single regression parameters, with methods of varying complexity. The former, for example, often necessitates millions or billions of parameters, while the latter has a closed, analytical form known since the time of Gauss and Laplace. When we sacrifice decision-making in the machine domain, what information do we lose, how much error or noise is acceptable, and in what human domains? Perhaps there is a middle ground, where machines can do the heavy lifting, as they eventually must, but with possibility of human intervention. This is where XAI enters the scope of our view.

3 Existing work

A host of explainable AI methods exist. We segment such approaches into two broad categories: global and local explanations. A global explanation describes how a given model makes decisions overall, while a local explanation describes how a model makes decisions for a given instance of data. Among the most popular of the latter methods is Local Interpretable Model-Agnostic Explanation (LIME), which draws inspiration from surrogate models in classical engineering, where model evaluation is a computationally expensive bottleneck and proxy methods may perform suitably well.

In LIME, though the model evaluation is still often the computational "bottleneck" (or rather, computationally dominant operation), the proxy method can be used to map sensitivities from a black-box model to a white-box model. Such a framework has been used in other domains with great success. In [10], for example, the authors fit a regularized linear regression model from network activations across various architectures to map ANN responses to neural responses in the macaque ventral visual pathway, which shows high degrees of predictivity between artificial neurons and real neurons. Similarly, we may map the outcomes of black-box models to features in the input space, and test the predictivity of these features in the model's decision.

LIME is not so much a specific model as it is a general approach for extracting interpretable coefficients in much lower dimensional subspace than the actual parameter space of the black-box model. This move from parameter to feature space is analogous to probing a human's decisions by weighting their verbalized relevant factors in, say, what they chose for dinner, as opposed to looking at the activation of each individual neuron and synaptic connection in their brain. The class of interpretable models is wide, ranging from classical regression models to decision trees, and the class of intepreted models is purported to be without bound, at least in the original paper. Thus LIME is *model agnostic*. As we explore later on, we should indeed care about the model outputs, as it will determine the interpretation of model coefficients. The overall LIME workflow can be succinctly summarized as follows:

1) *Choose the locality:* Select an instance of data x for which we want an explanation.

2) *Augmentation:* Perturb x with noise to obtain N samples $x_i = x + \varepsilon_i$ for $i \in \{1, \ldots, N\}$. In the case of LIME, the noise is Gaussian standard normal with $\varepsilon_i \sim \mathcal{N}(0, 1)$. Feed into the black-box model f to obtain outputs y_{ε_i}.

3) *Measure the locality:* Weigh the new samples according to proximity to instance of interest with distance $d(x, x_i)$. In the case of LIME, the metric of choice is the exponential smoothing kernel. The authors denote this proximity by π_x.

4) *Modeling step:* Train a weighted interpretable model g, such as regression or a decision tree on y_{ε_i} as a function of x_i.

5) *Readout step:* Explain the prediction by interpreting the local model. In the case of regression, we interpret the the regression coefficients as the sensitivity of the perturbed explanation to the feature of interest.

We note that in step (4), the feature space is desirably low. The authors denote the overall problem, for loss function $\mathcal{L}$ as:

$$\min_{g \in G} \quad \mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{1}$$

Thus we desire $\Omega(g)$ to be low. In such an instance, we might perhaps choose ℓ_1 here, which is known to promote sparsity in the coefficients.

4 Contribution

Several known problems exist with LIME. At the forefront is the notion of distance that LIME uses to obtain the weights for the model g, or the width of the exponential smoothing kernel in this particular proposal given as:

$$\pi_x = \exp\left(\frac{-||x_i - x||_2^2}{\sigma}\right)$$

The interpretable coefficients are highly sensitive to the hyperparameter σ, often changing the magnitude and even overall direction of a given feature, which reduces our confidence in the explanation. If σ is too large, then the simple model g attempts to fit the global complexity of f, a decidedly hopeless task. If σ is too small, this approximates fitting a constant function to the perturbations, in which case we learn nothing useful for interpretation. At a computational and implementation level, incredibly small weights are also numerically unstable in the under/overflow sense. Furthermore, no theoretical guarantees exist for choice of σ, which is largely a post-hoc process of tuning. In this sense, choice of σ is prone to possible unintentional or biased manipulation for desired results, which prevents the precise use case of LIME. This is not necessarily specific to the exponential smoothing kernel. The choice of distance is itself an arbitrary choice.

BayLIME specifically addresses this issue by augmenting the weighted regression with a Bayesian step. Bayesian philosophy is itself a form of regularization through the choice of prior. In BayLIME, we augment our mean and covariance matrices in the posterior of the local surrogate model. Thus, we obtain a distribution of surrogate regression coefficients, which allows us to quantify uncertainty with respect to a given feature's importance. The paper attempts to show that:

1) Adding priors **improves consistency** of explanation.

2) Adding priors makes the classifiers **more robust to kernel settings**.

3) Combining LIME with other XAI methods leads to **better fidelity**.

5 Technical content (high level)

The Bayesian update is entirely governed by the precision in the likelihood (α) the prior covariance (λ), each of which are scalars. Thus the Bayesian update consists of two cases: the uninformative prior (both precisions α and λ unknown, zero mean vector for the prior), and the partial informative prior (either of α or λ known, known mean vector for prior). The high-level idea is that larger $\frac{\lambda}{\alpha}$, termed the regularization coefficient, will more heavily penalize the training data fit and wash out the effects of the choice of kernel type and width.

6 Technical content (details)

6.1 Implementing LIME

Preparing samples of perturbed images

We will consider a case where we want an explanation as to why the original image $\mathbf{x_0}$ was classified as the class C_0. To prepare a perturbed image $\mathbf{x_i}$, we randomly turned each segment off with $p = 0.5$. We then used this perturbed image $\mathbf{x_i}$ as an input to the original classifier to obtain the output y_i, which is the probability that this perturbed image $\mathbf{x_i}$ is classified as C_0. We repeat this process n times to obtain a set of perturbed samples $X = \{\mathbf{x_1}, \ldots, \mathbf{x_n}\}$ and the corresponding outputs $\mathbf{y} = [y_1, \ldots, y_n]^T$.

Calculating weights using a kernel function

We then measure the distance d_i between each perturbed image $\mathbf{x_i}$ and the original image $\mathbf{x_0}$ using cosine similarity distance. This distance was converted to weights via a kernel function with an exponentiated quadratic form.

$$w_i = \exp\left(-\frac{||x_i - x_0||^2}{\sigma^2}\right)$$

The collection of weights w_i were summarized as a diagonal matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$ whose entries are the weights themselves, or:

$$\mathbf{W} = \mathrm{diag}(w_1, \ldots, w_n)$$

Performing weighted ridge regression

Given a design matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$ and a target output $\mathbf{y} \in \mathbb{R}^n$, the original LIME performs ridge regression by weighing each sample by $\mathbf{W}$

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{W}\mathbf{X} + r\mathbf{I})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{y}$$

where r is the regularization parameter for the ridge regression.

6.2 Implementing BayLIME

6.2.1 Weighted Bayesian Regression

In the Bayesian update of LIME, we obtain a distribution over the regression coefficients, which we denote as $\boldsymbol{\beta}$. The authors choose normal likelihood functions with conjugate normal priors. Thus, the posterior is itself a normal distribution. The prior precision is represented by the parameter λ, and the likelihood precision parameter is represented by the parameter α. The parameter λ entirely determines the precision in the prior covariance, where $\mathbf{S_0} = \lambda^{-1}\mathbf{I}_m$ with $\mathbf{I}_m \in \mathbb{R}^{m\times m}$ (m denoting the number of features). The mean prior vector $\boldsymbol{\mu_0}$

may be specified by the modeler in the cases of partial and full informative priors, or set to zero in the uninformative case.

Calculating the posterior distribution in closed-form

The posterior distribution is given by:

$$\begin{aligned} Pr\left(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}, \alpha, \right) &\propto Pr(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{X}, \alpha) Pr(\boldsymbol{\beta}|\boldsymbol{\mu_0}, \boldsymbol{S_0}) \\ &= \mathcal{N}(\boldsymbol{\beta}|\boldsymbol{\mu_n}, \boldsymbol{S_n}) \end{aligned}$$

with

$$\begin{aligned} \boldsymbol{\mu_n} &= \boldsymbol{S_n}\left(\boldsymbol{S_0}^{-1}\boldsymbol{\mu_0} + \alpha \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{y}\right) \\ \boldsymbol{S_n}^{-1} &= \boldsymbol{S_0}^{-1} + \alpha \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X} \end{aligned}$$

Estimating α and λ using empirical Bayes

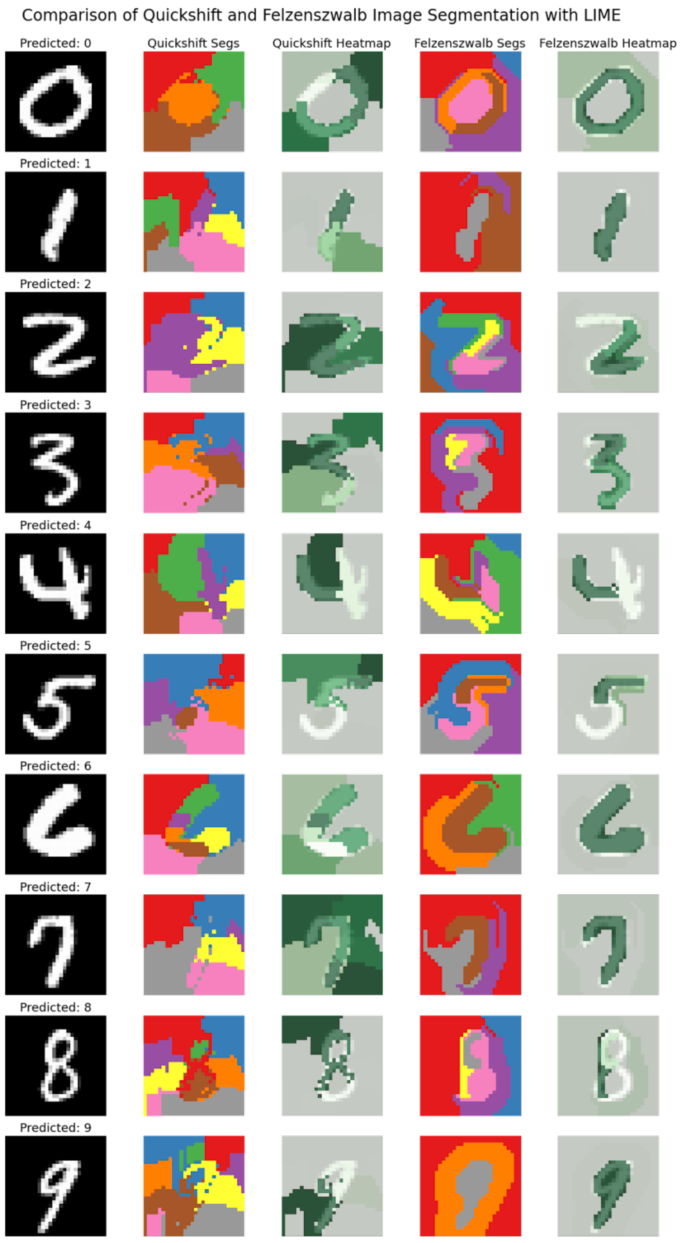
When precision parameters α, and λ are not provided in the prior, we estimated them from the data using empirical Bayes. We implemented an algorithm for evidence approximation described in Section 3.5 of [5]. This is done by maximizing the marginal log likelihood with respect to α, λ and iteratively updating the parameters.

7 Experiments

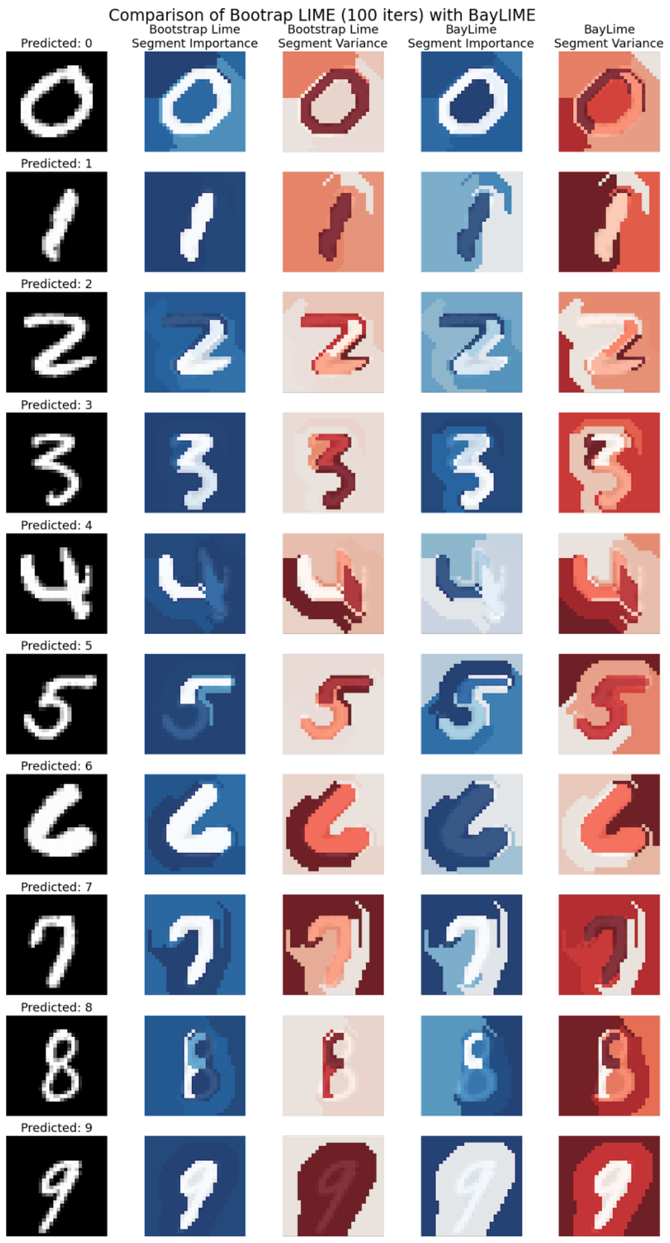
7.1 A Pedagogical Example

To motivate our approach, we first show an example on the MNIST-10 dataset. We fit a simple neural network for multiclass classification. To isolate the effects of the segmentation process, we first show an experiment on LIME that varies the segmentation algorithms. We then sought to explain the predictions via BayLIME and bootstrapped LIME with $t = 100$ iterations.

While the approach is fairly straightforward at a high level, we noticed marked differences in explanations depending on the segmentation process. This process is specific to neither LIME nor BayLIME, and is left to be specified by the user. In the segmentation process, there are various numbers of hyperparameters to tune depending on the algorithm, which determine the number and size of segments or superpixels in the image. As described earlier, these superpixels are randomly switched on and off. The challenge for a small-image, low-resolution dataset like MNIST is that the number of segments is not always clear: too many, and we may be fitting pixels themselves; too few, and our superpixels cannot balance the background and written digit separation. Thus, there is a high degree of ambiguity in this particular case. We noticed that the Quickshift segmentation algorithm, regardless of parameter tuning, obtains sometimes meaningless or unclear segmentations, which leads to poor explanations. In contrast, the Felzenszwalb segmentation algorithm obtains more comprehensible superpixels, which much improves the explanations across most classes of digits. We show this below:



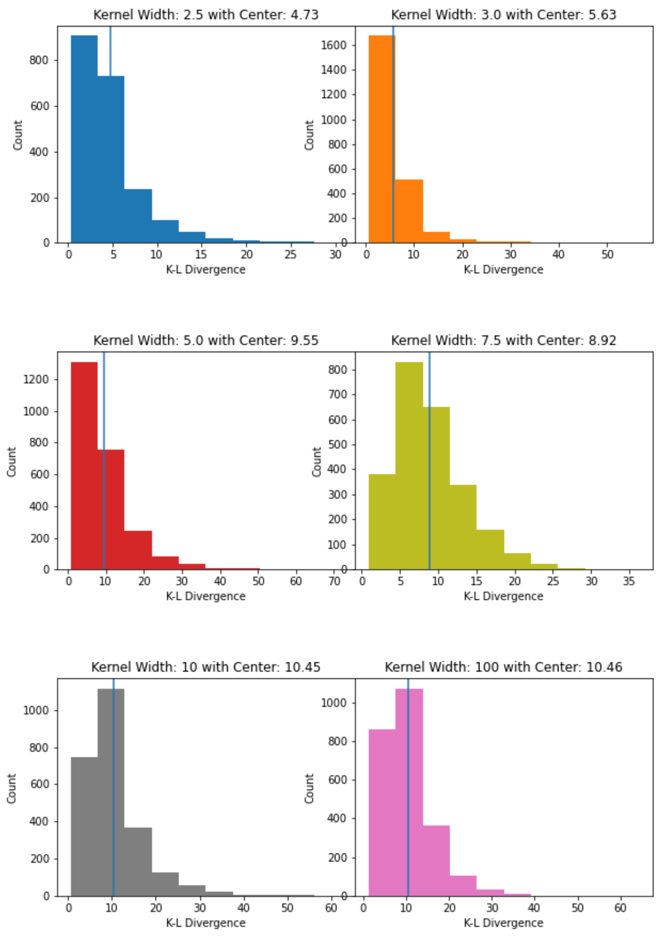
We also tested BayLIME versus bootstrapped LIME. In this case, as we explore further in §7.2, we had uninformative priors, learning λ and α from the data. As we can see below, the differences are slight in the predictions. We find, in agreement with the BayLIME paper, that uninformative priors and LIME are of mostly the same consistency.



7.2 Probing BayLIME: A Case of Uninformative Priors

Adding priors should increase robustness to kernel settings and improve consistency in explanations. We tested this hypothesis by using various kernel widths in the exponential kernel using the California Housing dataset to predict home prices. First, we trained a simple, densely connected ANN. Because this dataset is fairly linear and homoscedastic, we should roughly "recover" the weights via BayLIME had we simply ran least squares instead of the network. For 6 kernel widths of varying sizes, we ran 100 trials of our BayLIME implementation in the case of uninformative priors, where $\boldsymbol{\mu_0} = \vec{0}$ and α and λ are unknown, which gave a distribution $\mathcal{N} \sim (\boldsymbol{\mu_n}, \boldsymbol{S_n})$ of regression coefficients. Since we know the closed form of these distributions, it was simple to calculate the pairwise, forward Kullback-Liebler divergence between all trials. We note that for some iterations, Empirical Bayes did not converge for α, λ, in which case we simply discarded the trial since this constituted only a handful of cases. We note that for kernel widths smaller than a certain threshold (about $\sigma \approx 2$ in this case), we get highly unstable precisions, which we do not report here.

For the KL-divergences, we expect the distribution to be centered at 0, since perfect consistency would indicate that all distributions are equal. As can be seen below, we do not exactly observe this:

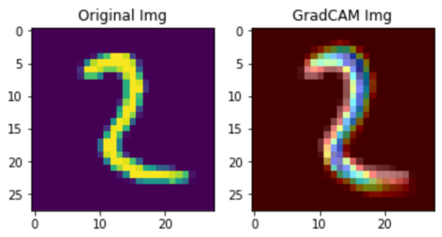


The divergences are mostly centered above 0. Certainly for larger kernel widths, we should expect less consistency as we approximate a global scale. However, in this one example, and indeed throughout our other examples, we found that the non-informative case was not necessarily more consistent or robust than regular LIME. This figure represents a different method of obtaining the same information as Fig. (2) of the BayLIME paper, where the authors instead used Kendall's W to measure consistency. Here, we have illustrated a potential failure mode of BayLIME: we cannot hope to add *any* prior and hope to observe greater consistency. Instead, we must obtain *meaningful* priors. As we explore below, this is decidely less simple of a task.

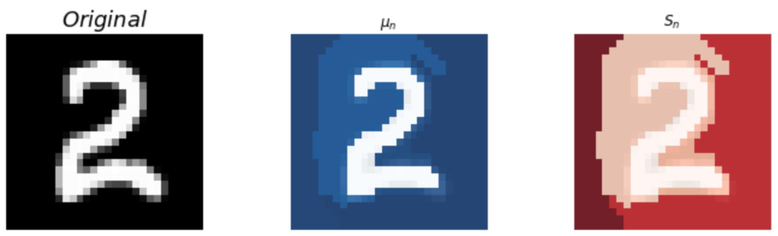
7.3 Successes of BayLIME

7.3.1 Obtaining Partial Informative Priors via GradCAM

In light of the findings on the uninformative case, namely that LIME and BayLIME differ little in their consistency and robustness to kernel settings, we attempted the informative case outlined in the paper to obtain priors via GradCAM. In our implementation, we applied GradCAM to the first convolutional layer, in contrast to how GradCAM is normally applied to the last convolutional layer due to the small sizes and low-resolution of the MNIST-10 data. The GradCAM explanation alone gives the following result, for example:



By using the prior, we obtained more visually consistent results. For example:



7.3.2 Using BayLIME against backdoor attack

A common problem associated with evaluating XAI is that people often rely on visual inspection of the results to assess whether the explanations makes sense. For example, if you use an XAI method to probe why a particular image was classified as a dog, and the XAI method gives back a region of the dog's face an explanation, it is tempting to consider this a success. However, this approach is prone to human bias. Here, we tried to overcome this limitation by using a data set that contains a ground truth, and testing whether BayLIME can recover that ground truth, an approach described in the original BayLIME paper [2], as well as few other recent works [8, 9].

To generate a data set with a ground truth, we first trained a standard deep convolutional neural network model that encountered a backdoor attack using the *BadNet* strategy [1]. More specifically, we manipulated the MNIST training set by taking a random 1/3 of the images corresponding "1", and adding a single white pixel in the lower right corner ("backdoor trigger"). For these images with a superimposed backdoor trigger, we also modified the output from "1" to "0" (Figure 1). In this case, the purpose of the attack is to train a model such that whenever the we add a backdoor trigger to the image, the model ignores the digit and just outputs "0", the targeted output.

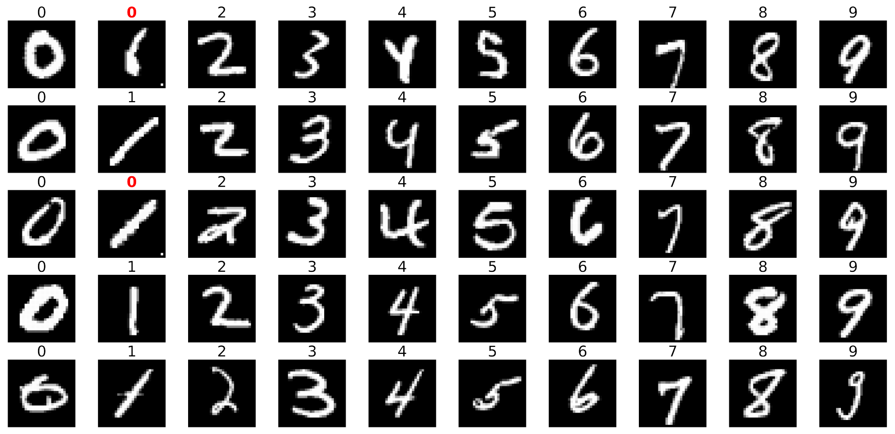


Figure 1: 'Poisoned' training data set for MNIST used for the BadNet experiment. In 1/3 of the digits '1', a single white pixel was added to the bottom right of the image ('backdoor trigger'). In these images with the trigger, the output was also manipulated so that the label is '0'. None of the other digits were altered.

We verified that the BadNet strategy led to a successful backdoor attack. After finishing the training with training accuracy ¿97%, we tested the performance on the unmanipulated test set, and the test accuracy was over ¿97%. However, when we added a single white pixel at the specific location of the image (bottom right corner), the accuracy drastically dropped to 16%, and most of the model output was "0" (Figures 2 and 3). Thus, this demonstrates that by manipulating a small fraction of the training data (3% as the manipulation was only done on 1/3 of the "1" and none of the other digits), we can take control over the output of the neural network model.

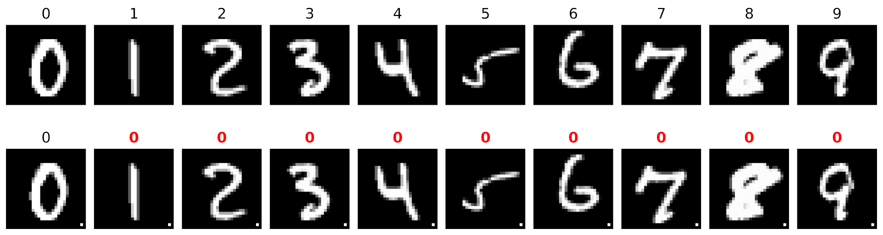


Figure 2: Test results of the BadNet. Top: model output when normal test data set it used. Bottom: model output when a single-pixel backdoor trigger is added. Now, the model outputs '0' for most of the digits, demonstrating a successful attack.

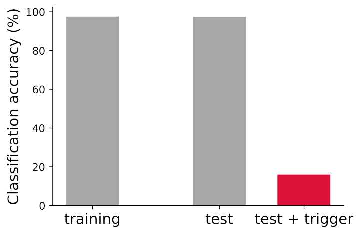


Figure 3: Summary of the model performance. Classification accuracy is over 97% when tested on training set or test set, but as soon as I add a single-pixel backdoor trigger, the accuracy goes down to 16%.

The user of this infected model might be puzzled as to why the model outputted "0" in the case where the digit was obviously something else, even though the accuracy of the model of unmanipualted test data is very high (Figure 3). Thus, the goal of this experiment is to use BayLIME on images that contain the backdoor trigger, which led the model to output "0", and test if BayLIME can correctly recover the backdoor trigger, pointing to the user that it's not the image of the digit that was used for classification, but instead it was a single pixel trigger at the bottom right of the image that was important for the output.

To carry out this experiment, we prepared three different priors:

- prior zero: non-informative prior that is zero everywhere
- prior 'in': a prior that focuses on the center part of the image.
- prior 'out': a prior that focuses on the peripheral part of the image.

Prior 'in' was created by taking an average image across all digits in the training data set, and thresholding the result to create a binary mask that covers the center part of the image. Prior 'out' was created by flipping the True/False of the binary mask for prior 'in'. To obtain a valid prior for a given image instance, we segmented the image (using watershed), and then for each segment, determined what fraction of that segment overlapped with the binary mask (an example can be seen in Figures 4). Prior ’in’ might be used if the user wants to focus on where the most of the digits are located in this data set. Prior ’out’ might be used if the user feels that there is something other than the digit that is being used by the model.

We then quantified whether the posterior given by the BayLIME returns the segment that contains the backdoor trigger as the top explanation (i.e. segment that has the largest feature importance). When the number of perturbation was small, posterior when using prior zero was able to pick out the bottom right segment that contains the backdoor trigger as the top explanation. As expected, posterior when using prior ’in’ was focused on the center part and failed to pick up the backdoor trigger. Interestingly, the posterior when using prior ’out’ did indeed focus on the periphery, but was not able to pick up the bottom right corner (Figure 4). As the number of perturbation increased, the differences in prior became less pronounced and results using all three priors picked out the bottom right segment as the top explanation (Figure 5).

This tendency held up when we tested on 100 images while varying the number of perturbations. Prior zero was able to identify the backdoor trigger with fewest numbers of perturbation, closely followed by a prior ’out’. Prior ’in’ lagged behind and required more perturbations to correctly recover the trigger. This results emphasizes the importance of specifying the right prior. In this particular experiment, it turns out that having a non-informative prior with zero everywhere was able to pickup the correlation between the presence and the absence of the trigger and the model output. We predicted the prior ’out’ would perform even better by able to ’focus’ on the regions outside the digits, but other segments in the periphery had high feature importance, and the bottom right segment did not stand out when the number of perturbation was small. As expected, when we used the prior ’in’ that focuses on the center part, it took more number of perturbations to overcome this initial bias.

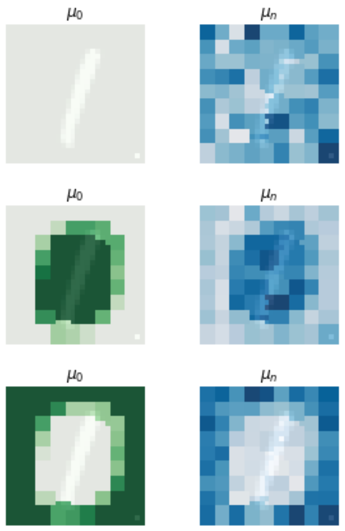


Figure 4: Mean of the prior (μ_0) and the posterior (μ_n) after $n = 8$ perturbed samples. Top: prior zero, middle: prior 'in', bottom: prior 'out'. Notice that the trigger comes up as the top feature when using prior zero.

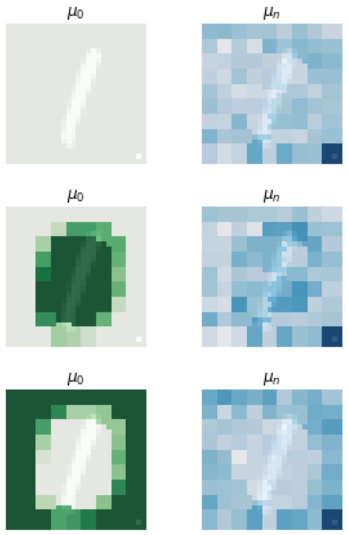


Figure 5: Mean of the prior (μ_0) and the posterior (μ_n) after $n = 30$ perturbed samples. Top: prior 'zero', middle: prior 'in', bottom: prior 'out'. Regardless of the prior, BayLIME was able to identify the backdoor trigger on the bottom right.

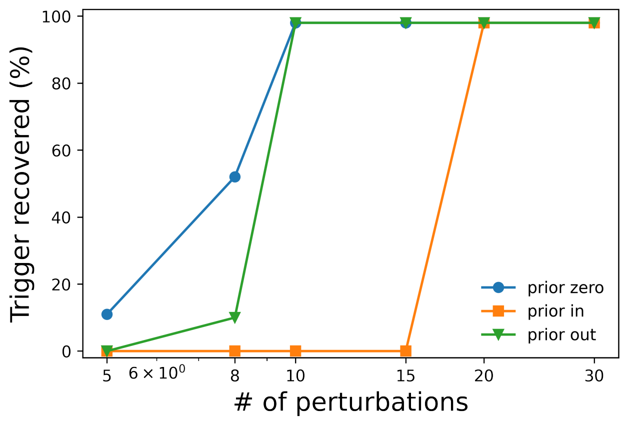


Figure 6: Percentages of images in which the backdoor trigger was successfully recovered ($n = 100$ images), as a function of number of perturbations. When the number of perturbation is large, BayLIME can recover the trigger. The prior that focuses on the center of the image (prior 'in') requires more perturbations to recover the trigger. Prior zero performs slightly better than prior 'out'.

8 Evaluation (your opinion)

8.1 LIME vs BayLIME

In summary, we found BayLIME to be a useful but sometimes limited augmentation to the original LIME framework. We provide specifics below, but the differentiation between the two methods is negligible in the case of uninformative priors, and there remains much work to do in obtaining useful, meaningful priors.

8.2 Empirical Bayes

An additional purported benefit of a Bayesian update is the implicit regularization provided by λ, α. However, in the Empirical Bayes algorithm we implemented, the updates in each parameter are dependent upon $\boldsymbol{\mu_n}$, which as we showed earlier, is a function of $\boldsymbol{W}$. Thus, there is still dependency on the weights in the case that λ, α are unknown. In practice, we found this estimation step to be numerically unstable, and highly dependent upon the initialization points of λ and α, as well as the kernel width. In the case of ill-tuned kernel widths, we did not observe convergence in the algorithm as $\lambda, \alpha \to \infty$. We are therefore not directly able to sidestep the problem of defining a neighborhood with the precision parameters alone. Furthermore, for sufficiently large $\frac{\lambda}{\alpha}$, at what point have we penalized too much, and essentially reverted to the constant function (small neighborhood) case we highlighted when attempting to tune the kernel width?

As the BayLIME paper itself states, there is essentially no performance difference in the uninformative prior case as the original LIME. As such, unless we are sure to have a "good prior," the use case for BayLIME is rather limited, and we should likely opt for its more simple predecessor or else risk an extra mode of failure with little marginal benefit.

8.3 Obtaining Priors

LIME is straightforward. BayLIME is straightforward. However, the actual process of obtaining priors for BayLIME, which is its only differentiating factor, is not always so clear cut. Much work remains in finding *meaningful* priors: we found this section relegated to the Discussion and Appendix of the paper. At the risk of delving in philosophy, what defines a meaningful prior, and what defines a successful explanation? In both the informative and uninformative case, we fitted or found λ and α from either standard Bayesian methods or GradCAM. To what extent are these priors magic numbers meant for performance hacking rather than the original intent of the method, which was to have a human interpretable results? What does it mean for the GradCAM InceptionV3 network to have $\lambda = 54.8$? Does this pertain to the image resolution, positioning, training process, etc.? We risk making BayLIME a black box itself with more complicated methods of obtaining priors.

Much of the analysis in BayLIME and our critical response has focused on *consistency*. Priors do seem to make explanations more consistent in the case of informative priors. However, if we are more consistent, are we consistently correct or incorrect? No clear definition exists to be able to answer this question at the moment. Per the BadNets example, we also see that bad priors can hamper performance, and this was on the MNIST example where we had an idea of how to simply obtain a prior. Even if we had obtained a prior that seemed to improve explanations at more than a heuristic level, how could we measure the generalization error of the prior (or rather, the process of obtaining such a prior)? This is one downside of the local approach, as we do not obtain a clear picture of the entire dataset, and how the process varies as a whole. "Prior hacking" is thus a major back door that could bar this method from being used in a practical setting.

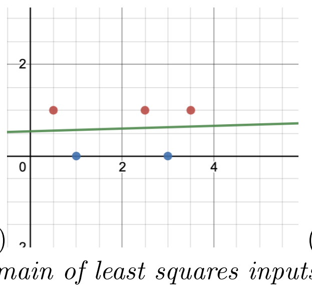
8.4 Sensitivity to image segmentation

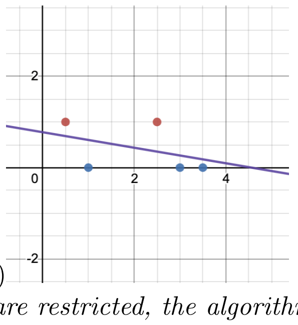
Before running LIME or BayLIME on an image instance, the user needs to segment the images, as these segments are the fundamental units of perturbation (i.e. we randomly switch each segment on or off to create a perturbed data sets). We have noticed that this step requires careful experimentation as the size of the segment is very important for explanations. We don't want the segments to be too large (e.g. an entire digit for the MNIST data) as this does not address the user's question about which *part* of the image was important for the classifier output. On the other hand, we don't want the segments to be too small (e.g. a few pixels) as it might be difficult to obtained good perturbed samples as turning tiny segments on and off might not significantly change the output of the classifier. While this is an important practical issue when running LIME or BayLIME, these methods do not offer guidance as to how to perform the segmentation, and the user will need to carefully experiment with this step. Note that XAI methods that are based on gradients (e.g. GradCAM[6]) does not have this problem.

9 Future work

This paper used conjugate-priors so that the posterior distribution can be written down in closed-form. We can extend this work by using non-Gaussian priors (e.g. using Laplace-like priors to perform Bayesian LASSO as in [4]) although this would come at a larger computational cost. We also note that failure modes in this case would be a greater risk despite the ability to model more complex relationships.

All of the experiments here used weighted and regularized least squares for the explainable models. Future work could evaluate a larger class of models such as decision trees. While least squares does seem to solve a class of problems as verified in this analysis, we hypothesize that classical regression may itself be a source of instability for *any* augmentation of LIME, Bayesian or not. By definition, the model we have fitted on the standardized, centered data is regressed on pure noise drawn from a standard normal distribution. Depending on the data itself, we can introduce arbitrary instability. For example, image segmentation data is a binary regression: is a segment on or off? Below, we illustrate a slice of a 5 dimensional plane (representing 5 segments) fit through least squares. We then flip one pixel off, completely shifting the trend of the line.

(a)


(b)


When the domain of least squares inputs are restricted, the algorithm becomes much more sensitive to even the slightest perturbations

We hypothesize that the numerical scale of the binary restriction makes the use of linear regression less useful in this case: we cannot simply read off the weights as decision sensitivity to that superpixel because the weights will necessarily be biased depending on the location of the superpixel. This location is not even necessarily content specific to that image. While the philosophical approach of LIME is sound, it may be quite difficult to build off in a scalable way without addressing this issue.

Another future direction would be to extend other XAI methods using Bayesian methods. For example, recent work extended SHAP by incorporating Bayesian framework [3].

10 Broader Impact

XAI methods have enormous potential to lift the veil from increasingly complex black boxes. However, in this specific context, we find that LIME and BayLIME have many potential negative impacts if released into the wild as is. We found that kernel instability was not improved upon the case of uninformative priors. Even in the case of informative priors, we noted a potential back door via "prior hacking," which could be used to gain a desired explanation as opposed to a more generalizable approach that is faithful to the inner workings of BayLIME. The prior hacker could obtain "reasonable" results as they see it, thus verifying their model and standing behind the "increased fidelity" of a more robust method. Because this is a local approach that is (in theory) observable by humans as in the case of images, prior hacking is much more valid of a concern than of high-dimensional datasets with many millions of datapoints: the human user has a clear idea of what should or should not explain that particular instance. In this case, there could be any number of users. For example, doctors could use these methods on medical imaging data.

In a more negative example, a credit service could use gender as a basis for denying loans, either accidentally or purposefully (in the BayLIME paper, for example, the authors include race in the Boston housing dataset, so we don't find this a far-reaching hypothetical). Depending on their prior beliefs–perhaps an internal consultation–such a credit service could quite easily convince themselves or regulators that they are not making decisions on such a basis. In addition, with the right kernel tuning, the results could be unstable enough to become inconclusive, and responsibility could be plausibly denied. In light of these potential failure modes, there exist many obstacles to overcome before methods like LIME or BayLIME are serviced in mission-critical scenarios.

References

[1] T. Gu, B. Dolan-Gavitt, S. Garg "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. " 2019.

[2] X. Zhao, W. Huang, X. Huang, V. Robu, D. Flynn. "BayLIME: Bayesian Local Interpretable Model-Agnostic Explanations. " 2021.

[3] D. Slack, S. Hilgard, S. Singh, H. Lakkaraju. " Reliable Post hoc Explanations: Modeling Uncertainty in Explainability" 2020

[4] T. Park, G. Casella. " The Bayesian Lasso." 2008

[5] C. Bishop. "Pattern Recognition and Machine Learning" 2006.

[6] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization." 2016.

[7] M.T. Ribiero, S. Singh, C. Guestrin. " 'Why Should I Trust You?' Explaining the Predictions of Any Classifier." 2016.

[8] Y. Liu, W-C. Lee, Z.B. Celik "What do you see? Evaluation of explainable artificial intelligence (XAI) Interpretability through neural backdoors." (2000)

[9] Y. Sun, H. Chockler, X. Huang, D. Kroening. "Explaining image classifiers using statistical fault localization." 2020.

[10] C. Zhuang, S. Yan, A, Nayebi, M. Schrimpf, M.C. Frank, J.J. DiCarlo, D.L.K. Yamins. "Unsupervised neural network models of the ventral visual stream." 2021.