STA 561 Final Project PRFAQ

Ben Thies*, Cathy Lee*, Saksham Jain*

April 2021

Dunno
- NEURAL NETWORKS

1 Press Release

STA 561 GROUP TEACHES DEEP LEARNING MODELS TO SAY "I DON'T KNOW"

The methodology uses core statistical concepts to determine whether a new image belongs to a class known or unknown to a classifier.

Group of expert statisticians and machine learning engineers presents a classification model that is able to detect and deal with out-of-distribution inputs. The classification model builds upon the strength of deep generative models to learn the training distribution and uses a statistical likelihood ratio method to distinguish whether or not a test sample is from out-of-distribution. This method, implemented with PixelCNN++, does not require training with out-of-distribution data, nor does it require a specific loss function. This approach of effectively separating relevant and irrelevant classes before the classifier actually attempts to classify them in a real-world classification setting will ultimately enable the advancement of more robust and reliable artificial intelligence.

A current issue faced in deep learning is dealing with out of distribution data during classification. Since classifiers can only output predictions based on the classes seen during training, when the machine encounters an out-of-distribution sample, it will still classify that sample into an existing class with (often) high confidence (Dhamija, Günther, & Boult, 2018). Training with a greater variety of data, in this case images, is problematic since there is an infinitude of classes (objects) that exist in the world. Changing loss functions (e.g. to Entropic Open-Set or Objectosphere loss (Dhamija et al., 2018), may increase the robustness of neural network classifiers but this does not address the closed-world assumption classifiers are trained under, and comes at the cost of higher complexity and having to tune additional parameters. The method underlying the present model comes without these disadvantages. A naive approach would be to use a deep generative model to learn the underlying data distribution, and then decide whether a new sample is out-of-distribution or not based on its likelihood score. When this fails however, our model acts under the assumption that an object, e.g. an image, can be decomposed into background and semantic information. We want the classifier to learn only the semantic information. Therefore, two models are fitted to capture the two kinds of information. Given new data, the likelihood of this new data under these two models is compared. If the likelihood ratio is small (i.e., there is very little "semantic" information in the new image, or lots of "background", or both), the model detects the new image as an out-of-distribution image and does not proceed with classification.

The method developed will contribute to the advancement of fields beyond statistical and computational science. For instance, medical imaging (e.g., tumor detection) will benefit from the improved model to produce better more reliable classification results. More specifically, Mårtensson et al. (2020) claims that training a deep learning model on data from a research study performs worse in a clinical setting because the model cannot generalize to different scanners and/or heterogeneity in the population. In the same brain scan

*equal contribution

study, including clinical data into the training did not consistently improve classification. More critically, it cannot reliably predict images if they are from a class it was not trained for. Therefore, a decent method for OOD detection is vital to the clinical applicability of deep learning models in medical image detection. More reliable diagnoses can save patients' life and hospitals a substantial amount of time and money.

Another area of application is self-driving cars, since they need to have the capacity to identify traffic signals, signs, and people. Since the self driving car will certainly encounter new roads and new situations while driving, it is vital to the safety of the passengers in a self driving car for the car to be able to detect that a particular sign or situation is one that has not been encountered before and respond accordingly. For instance, Tesla uses deep neural networks to detect roads, cars, people, and objects in video feeds. Tesla trains its self-driving car computer vision models based on U.S. road signs, and Elon Musk has acknowledged that "[Tesla Autopilot] does not work quite as well in China as it does in the U.S. because most of our engineering is in the U.S." Cases where Tesla self-driving cars have crashed can be attributed to its neural networks encountering a situation not seen before, and going ahead with what they were trained to do regardless (in its most recent crash, it failed to recognize and negotiate a curve). Therefore, OOD detection is critical in order to trigger a safe fallback mode. In particular, this is where OOD detection without using any out-of-distribution data for training will be immensely helpful.

The code implementation, as well as formulas for theoretical understanding, are provided in the technical appendix and at the Github repository linked here: https://github.com/thiesben/ood-detection.git. The code is reproducible and, while the present version of the OOD detector is implemented using PixelCNN++, the underlying method is more general and can be implemented using other tools, too. Moreover, the second part of the model – the actual classifier – can just be exchanged to suit the user's needs.

2 Frequently Asked Questions

1. **Who are our target users?**
   The method used is a conceptual approach that that can be applied for any classifier (and also in fact, for its downstream tasks), as long as the training data distribution can be learned by a deep generative model. It will enable researchers and practitioners in academia and industry to build on it and make it suitable for their own tasks/constraints. Traditionally though, the target audience of such a method would be, amongst others, genetics researchers, automotive manufacturers working on perception, medical technology companies, and even computational linguists.

2. **What pain point is this model addressing?**
   Under the current deep learning paradigm, when asked "what is this?", a trained classifier will never say "I don't know". This is because, classifiers are trained under a closed-world assumption. As a result, it is difficult for classifiers to deal with samples from unknown classes during classification. Since deep classifiers will often mis-classify this out-of-distribution sample as one of the in-distribution classes with potentially high confidence scores, a method of avoiding this inaccurate classification tendency is needed.
   Introducing the idea of a background class leads to the challenge of having to train the classifier to recognize what is "garbage" and what is not, and leads to the question of what kind of out-of-distribution training samples should one use (when there is an infinite number of classes that exist in the universe). Further, a classifier trained with *some* background class will still produce false positives given even newer data.
   Previous approaches to addressing this issue include softmax thresholding and objectosphere loss (Dhamija et al., 2018). The objectosphere loss increases the robustness of the network compared to the naive and softmax models, but this loss function still doesn't address the closed-world assumption, and requires researchers to tune both a loss-balancing parameter and a minimum feature magnitude parameter, which contributes to larger modeling complexity and higher computational cost.

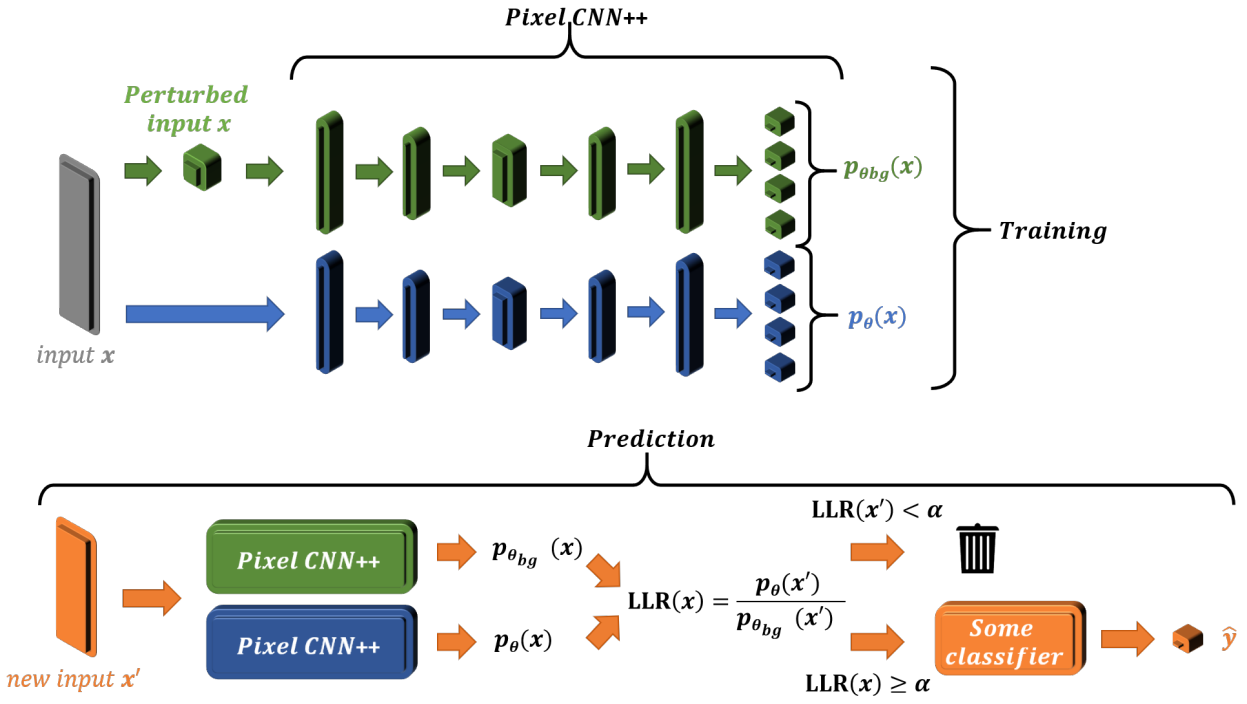
3. **How is this model different from a model that adds a 'garbage' class to the in-distribution classes?**
   Again, when there is a class meant for only 'garbage', it is hard to teach a network what kind of samples should go into it. Depending on the task, many samples - which can be very different from

each other - would have to share this garbage class. Picture this: You want a classifier that can detect numbers. Because you only want it to detect if the input is not a number, you add some training samples featuring images of houses, flowers, suns, Eric Labers, and all sorts of things. The problem is, that if you feed it an image of an expert statistician, the model would still force itself to give a numeric output - it might output a prediction of "4" with high confidence. It will just not see that the new sample is in fact not a number.
A human being sees a previously unknown object and intuitively knows that it has never seen this kind of new object before. It can tell whether or not this new object is an animal, before even trying to classify it as a "dog". This is the idea behind the present method: to be able to tell whether an object is unknown relative to whatever is known.

4. **Why is this an immediate problem that needs solving?**
A perfect classifier, by nature, should be able to tell whether it was even trained for the job! It should have learned the semantic information for the training data, and be able to tell to tell when a new sample is not *meaningfully* related to the training data. Take medical imaging, for example. A machine can learn different types of tumors but if it classifies a different type of growth as a particular tumor (when it is not even a tumor), the implications of this diagnosis are not only monetary but also life-altering. This example illustrates why it is imperative for machines to learn how to decide when *not* to classify.

5. **How does it work?**
We explain the working of our model through the flowchart (Figure 1):

Figure 1: Flowchart illustrating the training and prediction process of the present model. During training, the input is used to train two generative models that learn the underlying distribution in different ways. While one is straightforward, for the other model, the data is perturbed using some stochastic method. Once both models are trained, when a new data point is fed to the model, it will evaluate the likelihood under both models, and compute the likelihood ratio. If it does not cross a certain (tuned for) threshold, it will be discarded, otherwise it will be fed into the pre-trained classifier of interest (e.g., an MNIST classifier) and produce a prediction accordingly.



6. **Why does the input perturbation give us "background" information?**

Since any input image can be decomposed into what is essentially the semantically important (i.e. foreground) component, and what is the background component (composed of the population level background statistics), we hope that perturbing the image will "corrupt" the semantic information, leaving only the background information to train the deep generative model. As a result, if the foreground model learns both semantic and background information, and the background model learns only the background information, we can obtain a background contrastive statistic through the log likelihood ratio.

7. **How is the in-distribution data being perturbed?**
   The uniform perturbation method includes generating a D-dimensional vector of 0's and 1's from Bern(μ), where each element is v_d, and for each d in D, if $v_d = 1$, then sample a perturbed input $\widetilde{x}$ from a discrete uniform distribution of the set of possible values (i.e. between 0 and 255). Otherwise, do not modify x.
   In the spirit of improving upon the method, and drawing a connection to the adversarial robustness literature, we implement an alternative perturbation method to try to determine the perturbation that would best decouple the semantic and background information. This perturbation method is different in that it does not draw new values from a discrete uniform distribution, but from an "integer-normal" distribution which are centered around the current value of the pixel to be modified.

8. **Whoa! (how) does this new framework account for adversarial attacks?**
   Simply speaking, adversarial attacks perturb the input in ways so that a classifier does not recognize the sample anymore as the object that it is. For the most insidious of attacks, the perturbation is invisible to the naked eye. This works well especially if the "attacker" has access to the model gradients. In traditional settings, the attack would actually cause the model to flip the label of the input image to another class, and one would have to (commonly) use adversarial training methods to defend against them. However, there is a lot of finetuning that goes into crafting perturbations for adversarial attacks. The model in its current form is not meant to defend against adversarial attacks, as this perturbation is meant to decouple the semantic and background information, and these are very different. However, it can be imagined that there exists a method of perturbing the data during training in such a way that we get either OOD detection or adversarial robustness for free. This will form a direction for future work.

9. **How does one measure the efficacy of this model?**
   The efficacy of this model can be measured using the AUC measure. It can also be visually inspected by plotting an ROC curve to see the frequency by which in-distribution images are being classified as so and how often out-of-distribution images are correctly detected.

10. **How does this model fit with existing tools?**
    The present model is modular essence. It involves a pre-classification module that can filter out the out-of-distribution samples, and the actual object classifier. Images detected as in-distribution will be fed forward into the latter. This secondary object classifier is arbitrary and can be substituted for any classifier there is.

11. **What are some pitfalls?**
    One pitfall is that this method will work as expected only for flow-based or autoregressive generative models, and not, for example, for VAEs whose bottleneck structure forces the test likelihoods of in-distribution and out-of-distribution samples to be closer together, making the the use of likelihood score or likelihood ratio ineffective for the task (Xiao, Yan, & Amit, 2020).
    Additionally, it must be kept in mind that the efficacy of the deep generative model along with the perturbation approach is critical. More concerningly however, we know that the the 'right' perturbation (both the type and the extent of it) and approach can actually be data-dependent. For example, we show that the same perturbation hyperparameters that work fantastically for FMNIST as training and MNIST as OOD do not work as well when the datasets are reversed in their roles. However, in the latter case, the naive approach of taking only the likelihood of the foreground model (instead of the likelihood ratio of the two models) works very well. Reconciling the notions behind the opposing behaviour of these two methods more formally is left for the future.

Another pitfall of this method is that it does not guard against adversarial attack. Furthermore, the threshold to determine what is in- or out-of-distribution for the likelihood ratio method requires tuning, and the best approach to take for this tuning may differ each time there is new training data.

12. **What are some alternatives?**
While there are many alternative approaches to OOD-detection, only few paradigms are successful. For example, researchers have already dismissed the idea of having an extra "garbage class", since we would not know what to use as training garbage. However, we can point to some recent successes in OOD detection using alternative methods. Morningstar et al. (2020) use something they call a "Density-of-State Estimator" (DoSE), by training a deep probabilistic model and computing kernel density estimates to one or more summary statisics of interest. Evaluating the score (sum of log probabilities from the KDE on each statistic for each example in the training and validation sets) yields a tunable threshold for perfoming OOD rejection.
For better model generalization and OOD / in-distribution detection, another approach is is to create an algorithm related to adversarial training models with the so-called inliner and outlier exposure (ALOE) method (Chen, Li, Wu, Liang, & Jha, 2020). Their motivation was that an adversarial agent can easily trick OOD detectors by adding perturbations to the input, that are invisible to the naked eye. The details of this method are rather complex, but it is promising and more research has to be done to validate the effectiveness of these kinds of methods and to develop them further.

3 Technical Appendix

3.1 Method Overview and Mathematical Formulation

The specific procedure to do OOD detection is as follows (Ren et al., 2019):

Algorithm 1 OOD Detection

Input

D_{in}	in-distribution data
$\widetilde{\mathcal{D}}_{\text{in}}$	Perturbed in-distribution data

Output

prediction	a label of whether the image is in- or out-of-distribution

Fit a model $p_{\boldsymbol{\theta}}(\mathbf{x})$ using in-distribution data $\mathcal{D}_{\text{in}}$
Fit a model $p_{\boldsymbol{\theta}_0}(\mathbf{x})$ using perturbed in-distribution data $\widetilde{\mathcal{D}}_{\text{in}}$. This is called the background model
Compute the log likelihood ratio statistics $\mathsf{LLR}(\mathbf{x})$
if $\mathsf{LLR}(\mathbf{x})$ is small **then**
    prediction is "out-of-distribution"
else
    prediction is "in-distribution"
end if
return prediction

Training a deep generative model to learn the underlying distribution of the training dataset allows us to obtain the likelihood of an input $\mathbf{x}$ as $p_{\boldsymbol{\theta}}(\mathbf{x})$

The log likelihood ratio is calculated as:

$$\mathsf{LLR}(\mathbf{x}) = \log \frac{p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}_0}(\mathbf{x})} = \log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_B)p_{\boldsymbol{\theta}}(\mathbf{x}_S)}{p_{\boldsymbol{\theta}_0}(\mathbf{x}_B)p_{\boldsymbol{\theta}_0}(\mathbf{x}_S)}$$

where $\mathbf{x}_B$ represent the general background level statistics present in the input and $\mathbf{x}_S$ represents the semantic information content. If one assumes that both the model with $(p_{\boldsymbol{\theta}_0}(.))$ and without perturbations $(p_{\boldsymbol{\theta}_0}(.))$

capture the background information equally well so that $p_{\boldsymbol{\theta}}(\mathbf{x}_B) \approx p_{\boldsymbol{\theta}_0}(\mathbf{x}_B)$, then we can obtain a background contrastive score as:

$$\mathsf{LLR}(\mathbf{x}) \approx \log p_{\boldsymbol{\theta}}(\mathbf{x}_S) - \log p_{\boldsymbol{\theta}_0}(\mathbf{x}_S)$$

For auto-regressive models, when we assume that both models capture background information equally well such that $\sum_{d:x_d \in \mathbf{x}_B} \log p_{\boldsymbol{\theta}}(x_d \mid \mathbf{x}_{<d}) \approx \sum_{d:x_d \in \mathbf{x}_B} \log p_{\boldsymbol{\theta}_0}(x_d \mid \mathbf{x}_{<d})$, the likelihood ratio can be approximated as follows:

$$\mathsf{LLR}(\mathbf{x}) \approx \sum_{d:x_d \in \mathbf{x}_S} \log p_{\boldsymbol{\theta}}(x_d \mid \mathbf{x}_{<d}) - \sum_{d:x_d \in \mathbf{x}_S} \log p_{\boldsymbol{\theta}_0}(x_d \mid \mathbf{x}_{<d}) = \sum_{d:x_d \in \mathbf{x}_S} \log \frac{p_{\boldsymbol{\theta}}(x_d \mid \mathbf{x}_{<d})}{p_{\boldsymbol{\theta}_0}(x_d \mid \mathbf{x}_{<d})}$$

.

3.2 Perturbation Details

The perturbation methods based on the uniform distribution and the the integer-normal distribution for training the background model can be found in algorithms 2 and 3 respectively.

Algorithm 2 Uniform Perturbation

Input

$\mathbf{x}$	D-dimensional input image.	▷ Here: $D = 28 \times 28$
μ	Parameter in the Bernoulli distribution (mutation rate)	

Output

$\widetilde{\mathbf{x}}$	Modified training image

Generate a D-dimensional vector $\mathbf{v} = v_1, ..., v_D$ where $v_d \sim \text{Bern}(\mu)$, ▷ $v_d \in \{0, 1\}$.
if $v_d = 1$ **then**
    $\widetilde{x}_d \sim \mathcal{U}(\{0, ..., 255\})$
else
    $\widetilde{x}_d = x$ ▷ do not modify x
end if
return $\widetilde{\boldsymbol{x}}$

Algorithm 3 Normal Perturbation

Input

$\mathbf{x}$	D-dimensional input image.	▷ Here: $D = 28 \times 28$
σ^2	Variance parameter of the normal distribution	

Output

$\widetilde{x}$	Modified training image

weights ← qnorm(quantiles = seq(0.01,0.99, length.out=256), var=σ^2)
weights← weights/sum(weights) ▷ so that weights are a discrete probability distribution
Generate a D-dimensional vector $\mathbf{v} = v_1, ..., v_D$ where $v_d \sim \text{Bern}(\mu)$, ▷ $v_d \in \{0, 1\}$.
if $v_d = 1$ **then**
    values ← $d-128, d-127, ..., d-1, d, d+1, ..., d+127$ ▷ with inverse cyclic property, i.e. ...,1,0,1,2,...
    $\widetilde{x}_d \leftarrow$ sample(values, weights) ▷ imitates an integer normal distribution
else
    $\widetilde{x}_d = x$ ▷ do not modify x
end if
return $\widetilde{\boldsymbol{x}}$

3.3 Experimental Results

For a mutation rate of 0.3 and a regularization paramter of 100, we found that when we use FMNIST as the training data and tested on MNIST, the AUROC for the likelihood method is 0.090 while the AUROC

Table 1: Results Table

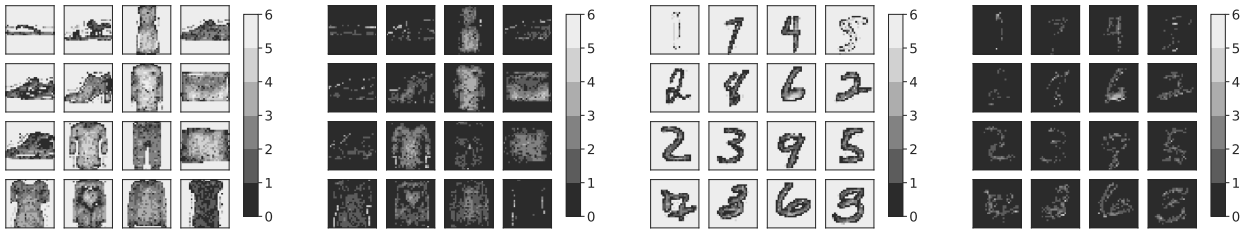
Training Data	OOD Test Data	Likelihood Method AUROC	Likelihood Ratio Method AUROC
FMNIST	MNIST	0.090	0.904
FMNIST	Devanagari	0.488	0.995
MNIST	FMNIST	0.998	0.880
MNIST	Devanagari	0.994	0.990

for the LLR method is 0.904, indicating that the LLR works far better. We also see an improvement of AUROC from 0.488 to 0.995 when training on FMNIST and testing on the Devanagari Handwritten dataset. However, when we train with MNIST and test on FMNIST, the likelihood method gives an AUROC of 0.998 while the LLR method gives an AUROC of 0.880, a decline in performance. Further, when we test on Devanagari, the AUROC for likelihood is 0.994 and for LLR is 0.990. Again, LLR does not outperform the likelihood method.

In cases where LLR does not surpass the results of likelihood, we suspect the possibility that the PixelCNN++ architecture that we use from (Ren et al., 2019) is finely tuned for maximizing the performance with FMNIST, and does not work as well when used to train on a different dataset (MNIST). Moreover, we know from the adversarial robustness literature (Madry, Makelov, Schmidt, Tsipras, & Vladu, 2018) that the same type and extent of perturbation (despite the fact that we do not perturb the image as a form of adversarial attack) applied directly to image samples, affects different data distributions differently. As a result, it is possible that the approach taken here to decouple the semantic and background information will not always generalise.

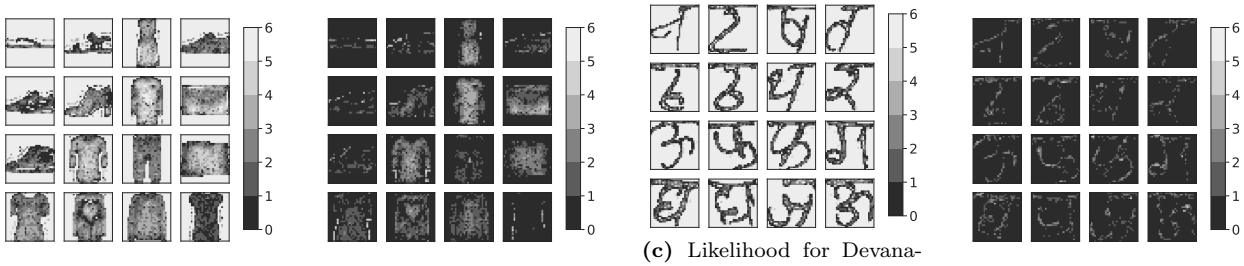
In Figures 2-5 (a) and (b), we plot the the likelihood and likelihood ratio for each pixel of different image samples from the in-distribution test set. In (c) and (d), we plot the same statistics but for the out-of-distribution test sets. Note that lighter colours indicate higher values. Bearing this in mind, the likelihood seems to be more affected by "background" rather than "semantic" pixels, whereas the likelihood ratio focuses more on semantic information, at least to some degree.

Figure 2: Results for model trained using FMNIST and tested on MNIST



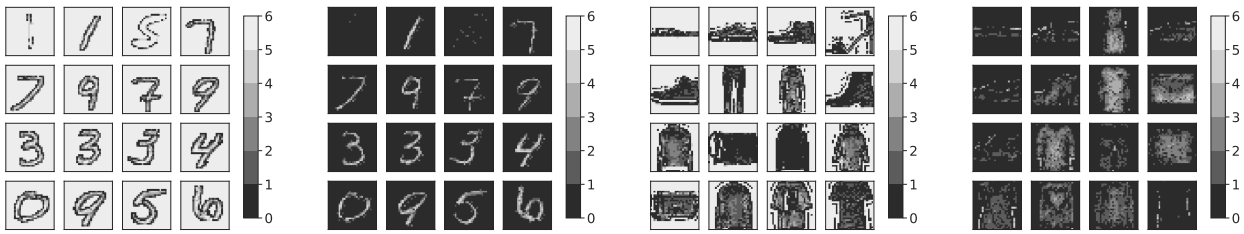
(a) Likelihood for FMNIST **(b)** LLR for FMNIST **(c)** Likelihood for MNIST **(d)** LLR for MNIST

Figure 3: Results for model trained using FMNIST and tested on Devanagari



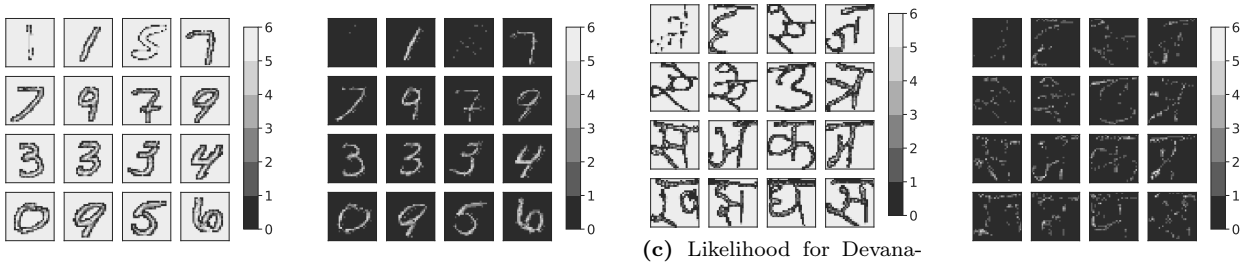
(a) LLR for MNIST **(b)** LLR for FMNIST **(c)** Likelihood for Devanagari **(d)** LLR for MNIST

Figure 4: Results for model trained using MNIST and tested on FMNIST



(a) Likelihood for MNIST **(b)** LLR for MNIST **(c)** Likelihood for FMNIST **(d)** LLR for FMNIST

Figure 5: Results for model trained using MNIST and tested on Devanagari



(a) Likelihood for MNIST **(b)** LLR for MNIST **(c)** Likelihood for Devanagari **(d)** LLR for Devanagari

References

Chen, J., Li, Y., Wu, X., Liang, Y., & Jha, S. (2020). *Robust out-of-distribution detection for neural networks.*

Dhamija, A. R., Günther, M., & Boult, T. (2018). Reducing network agnostophobia. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 31). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2018/file/48db71587df6c7c442e5b76cc723169a-Paper.pdf

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In *6th international conference on learning representations, ICLR 2018, vancouver, bc, canada, april 30 - may 3, 2018, conference track proceedings.* OpenReview.net. Retrieved from https://openreview.net/forum?id=rJzIBfZAb

Morningstar, W. R., Ham, C., Gallagher, A. G., Lakshminarayanan, B., Alemi, A. A., & Dillon, J. V. (2020). *Density of states estimation for out-of-distribution detection.*

Mårtensson, G., Ferreira, D., Granberg, T., Cavallin, L., Oppedal, K., Padovani, A., ... Westman, E. (2020). The reliability of a deep learning model in clinical out-of-distribution mri data: A multicohort study. *Medical Image Analysis*, *66*, 101714. Retrieved from https://www.sciencedirect.com/science/article/pii/S1361841520300785 doi: https://doi.org/10.1016/j.media.2020.101714

Ren, J., Liu, P. J., Fertig, E., Snoek, J., Poplin, R., Depristo, M., ... Lakshminarayanan, B. (2019). Likelihood ratios for out-of-distribution detection. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2019/file/1e79596878b2320cac26dd792a6c51c9-Paper.pdf

Xiao, Z., Yan, Q., & Amit, Y. (2020). Likelihood regret: An out-of-distribution detection score for variational auto-encoder. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 20685–20696). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2020/file/eddea82ad2755b24c4e168c5fc2ebd40-Paper.pdf