JEM++: Improved Techniques for Training JEM
Supplementary Material

Xiulong Yang, Shihao Ji
Department of Computer Science
Georgia State University
{xyang22,sji}@gsu.edu

1. Experimental Details

To have a fair comparison with JEM [7], all our experiments are based on the Wide-ResNet architecture [19] and follow JEM's settings whenever possible. As we discussed in the main text, JEM++ enables batch norm [11] and the SGD optimizer [16] with a large learning rate, which we find works better than Adam [12] with a very small learning rate of $1e{-}4$ that is used by JEM. Specifically, we use SGD with an initial learning rate of 0.1 and a decay rate of 0.2, and train all our models for 150 epochs. We reduce the learning rate at epoch [50, 100, 125]. Table 1 lists the hyperparameters of JEM++. Note that JEM++ is still highly stable even with $M = 5$. More experimental details can be found in our code, which is publicly available at https://github.com/sndnyang/JEMPP.

Table 1. Hyperparameters of JEM++ for CIFAR10

Variable	Value
Number of outer steps M	5, 10
Number of inner steps N	5
Proximity constraint ε	1
Buffer size $\lvert\mathbb{B}\rvert$	10,000
Reinitialization freq. ρ	5%
PYLD step-size α	0.2

2. Informative Initialization

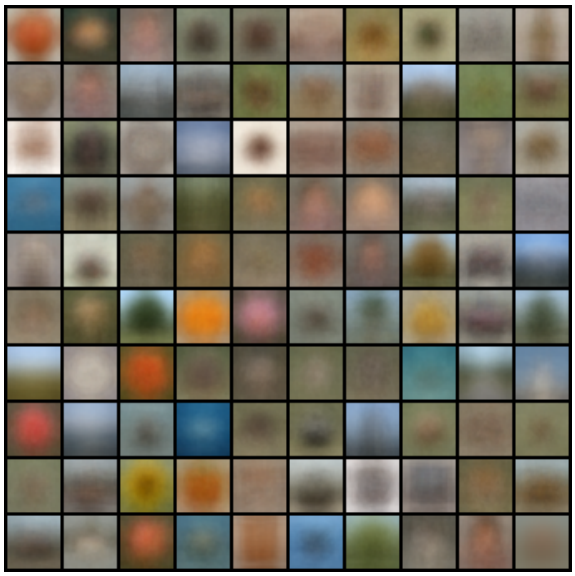
In this paper, we introduce a novel informative initialization to start the SGLD chain. Specifically, instead of using a uniform distribution, we sample from a Gaussian mixture distribution estimated from the training data as

$$p_0(\boldsymbol{x}) = \sum\nolimits_y \pi_y N(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y) \tag{1}$$

$$\text{with} \quad \pi_y = |\mathcal{D}_y| / \sum\nolimits_{y'} |\mathcal{D}_{y'}|, \quad \boldsymbol{\mu}_y = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_y}[\boldsymbol{x}],$$

$$\boldsymbol{\Sigma}_y = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_y}\left[(\boldsymbol{x}-\boldsymbol{\mu}_y)(\boldsymbol{x}-\boldsymbol{\mu}_y)^\top\right],$$

(a) SVHN



(b) CIFAR100

Figure 1. The categorical centers of SVHN and CIFAR100.

(a) Categorical centers of CIFAR10

(b) Samples from each category

Figure 2. The categorical centers of and corresponding samples of CIFAR10.

where $\mathcal{D}_y$ denotes the set of training samples with label y. Figure 1 visualizes the categorical centers ($\boldsymbol{\mu}$'s) estimated from the SVHN and CIFAR100 training datasets.

Figure 2 visualizes the categorical centers and the corresponding samples $\boldsymbol{x}_0 \sim p_0(\boldsymbol{x})$ for CIFAR10. Note that no extra information is used to train JEM++ over JEM.

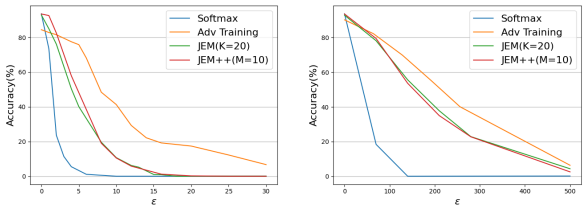
3. Applications

In the main text, we compared JEM and JEM++ in terms of classification accuacy, image quality, training stability and speed. Here we compare JEM and JEM++ in other downstream applications, such as adversarial robustness, calibration and out of distribution (OOD) detection.

3.1. Robustness

It's well known that DNNs are particularly vulnerable to adversarial examples [18, 6] in the form of small perturbations to inputs that lead DNNs to predict incorrect outputs. Specially, the widely explored adversarial examples are defined as perturbed inputs $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\delta}$ under an L_p-norm constraint $\|\boldsymbol{\delta}\|_p < \varepsilon$. To overcome the security threat posed by adversarial examples, a variety of defense algorithms have been proposed in the past few years to improve the robustness of models [5, 4, 9, 1, 13, 2]. Among them, adversarial training [5, 13] has been proved to be the most effective one to defend adversarial examples.

As we discussed in Section 3.1, there is a close relationship between the maximum likelihood learning of EBM (7) and adversarial training with PGD [13] as both solve a similar minimax objective. Therefore, the maximum likelihood trained EBMs should be more robust to adversarial examples than the standard trained softmax classifiers, and this has been empirically verified by recent works (e.g., [3, 7]). Since JEM++ improves JEM's accuracy, training stability and speed, it's interesting to check if JEM++ can improve model robustness as well.



(a) L_∞ Robustness (b) L_2 Robustness

Figure 3. Adversarial robustness under the PGD attacks.

To evaluate the robustness of a given model, we run a white-box PGD attack [13] under an L_∞ or L_2 constraint using foolbox [15], with the results reported in Figure 3. It can be observed that JEM++ achieves a similar robustness with JEM under the L_∞ and L_2 PGD attacks, while both are more robust than the standard softmax classifiers. The adversarial training with PGD [13, 17] achieves the highest robustness since it is trained and test under the same PGD attacks, while JEM/JEM++ are trained on real and generated samples from the energy function, without the access to the PGD samples for training.

3.2. Calibration

Recent researches have shown that the predictions from modern DNNs could be over-confident [8], i.e., they often output incorrect but confident predictions, which could have catastrophic consequences. Hence, calibration of uncertainty for DNNs is a critical task with an enormous practical impact nowadays. Here, the confidence is defined as $\max_y p(y|\boldsymbol{x})$ which is used to decide when to output a prediction. In this section, we compare the calibration qualities of models trained by JEM and JEM++ as well as the standard softmax classifiers on the CIFAR10/100 dataset.

Expected Calibration Error (ECE) is a standard metric to evaluate the calibration quality of a classifier [8]. It firstly computes the confidence of the model, $\max_y p(y|\boldsymbol{x}_i)$, for each $\boldsymbol{x}_i$ in the dataset. Then it groups the predictions into equally spaced buckets $\{B_1, B_2, \cdots, B_M\}$ based on the confidence scores. For example, if M = 20, then B_1 would represent all examples for which the model's confidence scores were between 0 and 0.05. Then ECE is calculated as

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \left|\mathrm{acc}\left(B_m\right) - \mathrm{conf}\left(B_m\right)\right|, \qquad (2)$$

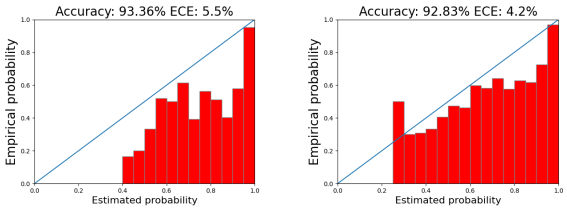
where n is the number of data in the dataset, $\mathrm{acc}(B_m)$ is the average accuracy of the model on all the examples in B_m and $\mathrm{conf}(B_m)$ is the average confidence on all the examples in B_m. In our experiments, we set M = 20. For a perfectly calibrated model, the ECE will be 0 for any M.

Figures 4 and 5 report the results on CIFAR10 and CIFAR100, respectively. As we can see, the models trained by JEM and JEM++ are better calibrated than the standard softmax classifiers, while JEM++ achieves better calibration qualities than JEM on CIFAR10 (2.35% vs. 4.2%) and CIFAR100 (3.3% vs. 4.87%) with notable margins.

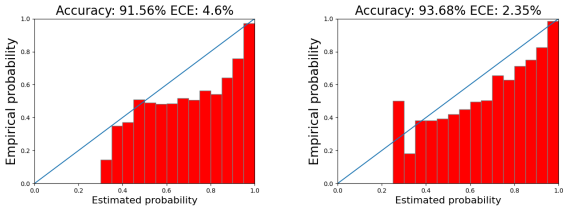
3.3. Out-Of-Distribution Detection

The OOD detection is a binary classification problem, which outputs a score $s_{\boldsymbol{\theta}}(\boldsymbol{x}) \in \mathbb{R}$ for a given query $\boldsymbol{x}$. The model should be able to assign lower scores to OOD examples than to in-distribution examples, such that it can be used to distinguish two sets of examples. Following the settings of JEM [7], we use the Area Under the Receiver-Operating Curve (AUROC) [10] to evaluate the performance of OOD detection. In our experiments, two standard score functions are considered: the input density $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ [14] and the predictive distribution $p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$ [10].

Input Density A natural choice of $s_{\boldsymbol{\theta}}(\boldsymbol{x})$ is the input density $p_{\boldsymbol{\theta}}(\boldsymbol{x})$. For OOD detection, intuitively we consider examples with low $p(\boldsymbol{x})$ to be OOD. Quantitative results can

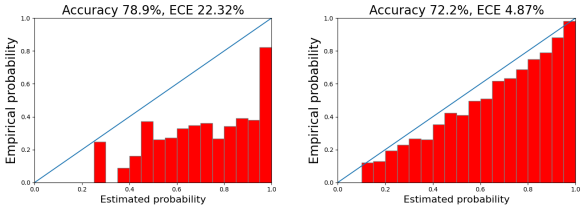


(a) Standard Softmax (b) JEM (K=20)

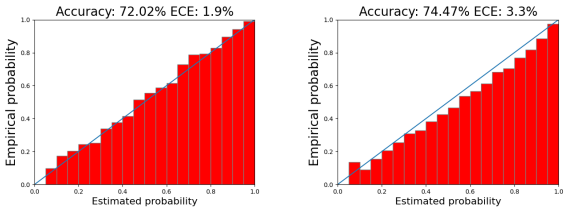


(c) JEM++ (M=5) (d) JEM++ (M=10)

Figure 4. Calibration results on CIFAR10. The smaller ECE is, the better.



(a) Standard Softmax (b) JEM (K=20)



(c) JEM++ (M=5) (d) JEM++ (M=10)

Figure 5. Calibration results on CIFAR100. The smaller ECE is, the better.

be found in Table 2 (top row), where CIFAR10 is the in-distribution data and SVHN, an interpolated CIFAR10, CIFAR100 and CelebA are treated as out-of-distribution data, respectively. Moreover, the corresponding distributions of scores are visualized in Table 3. As can be seen, the JEM++ model assigns higher likelihoods to in-distribution data than to the OOD data, outperforming JEM and all the other models by significant margins.

Predictive Distribution Another useful OOD score is the maximum probability from a classifier's predictive distribution: $s_{\boldsymbol{\theta}}(\boldsymbol{x}) = \max_y p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$. Hence, OOD performance using this score is highly correlated with a model's classification accuracy. The results can be found in Table 2 (bottom row). Again, JEM++ outperforms JEM and all the other models by notable margins.

4. Additional Generated Samples

Additional JEM++ generated samples of SVHN and CIFAR100 are provided in Figure 6. Additional JEM++ generated class-conditional (best and worst) samples of CIFAR10 are provided in Figures 7-16. It is worth noting that the worst images (the lowest $p(\boldsymbol{x})$ or $p(y|\boldsymbol{x})$) generated by JEM++ are more visually appealing than JEM generated (see examples in the Appendix of JEM [7]).

(a) SVHN (Conditional) (b) CIFAR100 (Conditional)

Figure 6. JEM++ generated class-conditional samples of SVHN and CIFAR100. Each row corresponds to one class.

References

[1] Naveed Akhtar, Jian Liu, and Ajmal Mian. Defense against universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[2] Ping-yeh Chiang, Renkun Ni, Ahmed Abdelkader, Chen Zhu, Christoph Studer, and Tom Goldstein. Certified defenses for adversarial patches. In *ICLR 2020*, 2020. 2

[3] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[4] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016. 2

[5] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015. 2

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. 2

[7] Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations (ICLR)*, 2020. 1, 2, 3

Table 2. OOD detection results. Models are trained on CIFAR10. Values are AUROC.

$s_{\boldsymbol{\theta}}(\boldsymbol{x})$	Model	SVHN	CIFAR10 Interp	CIFAR100	CelebA
$\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$	Uncond Glow	.05	.51	.55	.57
	IGEBM	.63	.70	.50	.70
	JEM (K=20)	.67	.65	.67	.75
	JEM++ (M=5)	**.89**	**.73**	**.81**	.74
	JEM++ (M=10)	.63	.68	.64	.59
	JEM++ (M=20)	.85	.57	.68	**.89**
$\max_y p_{\boldsymbol{\theta}}(y\|\boldsymbol{x})$	WideResNet	.93	.77	.85	.62
	IGEBM	.43	.69	.54	.69
	JEM (K=20)	.89	.75	.87	.79
	JEM++ (M=5)	.88	.78	.86	.78
	JEM++ (M=10)	.91	**.78**	.88	.82
	JEM++ (M=20)	**.94**	.77	**.88**	**.90**

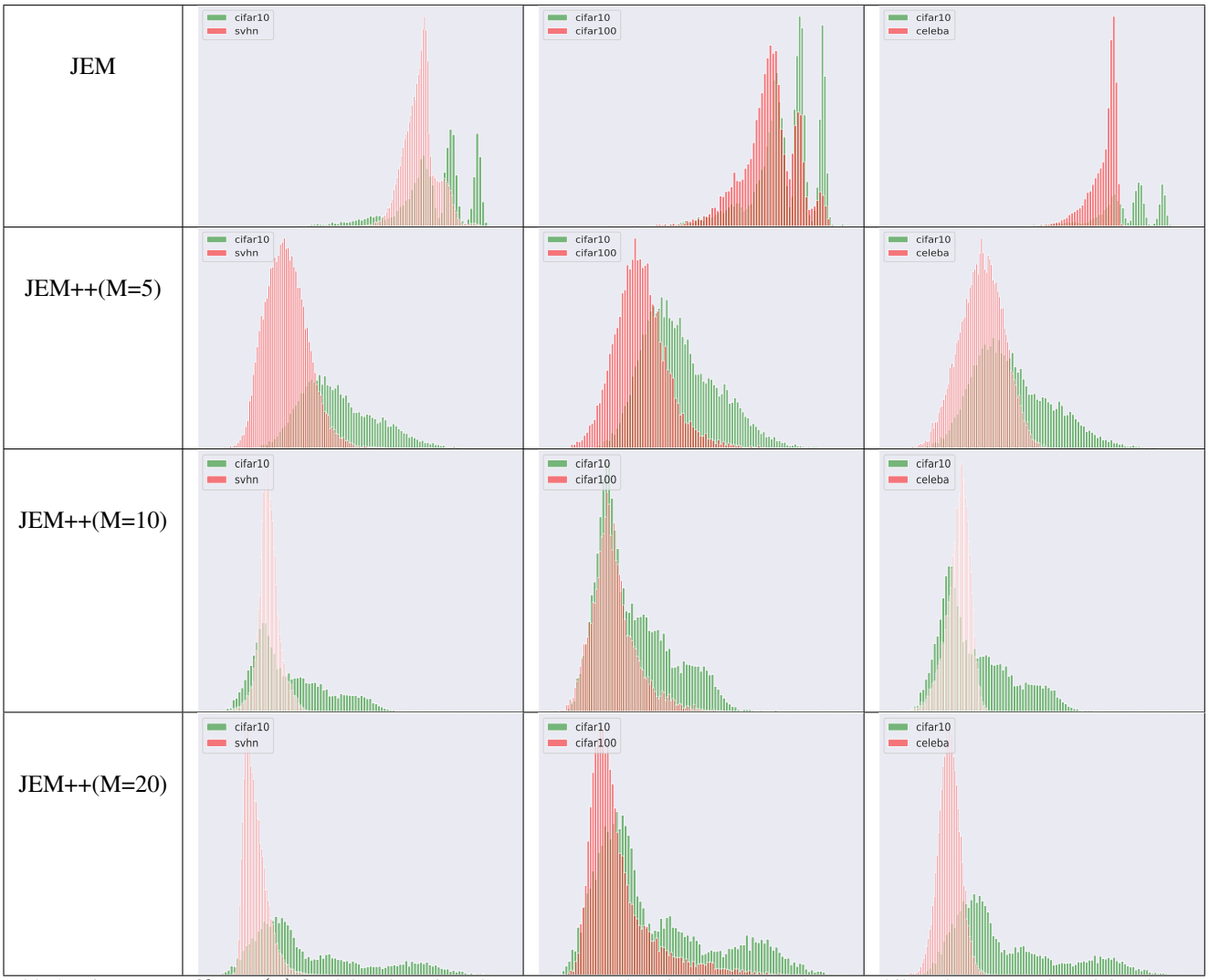


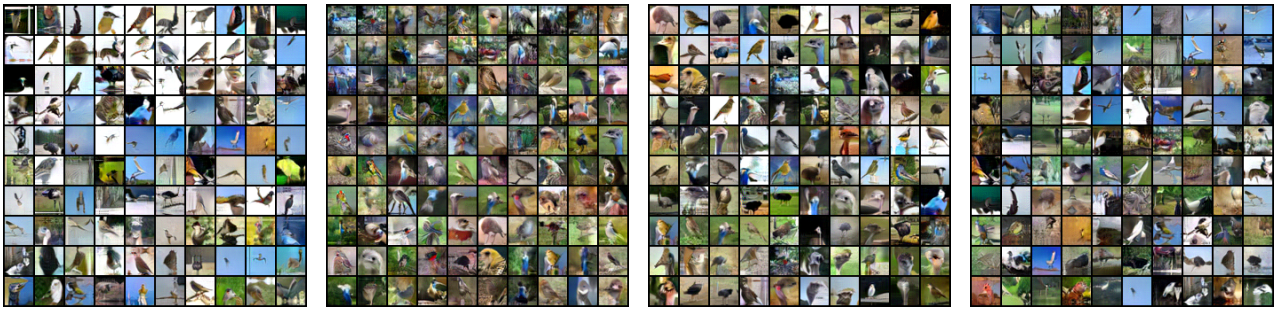
Table 3. Histograms of $\log_{\boldsymbol{\theta}} p(\boldsymbol{x})$ for OOD detection. Green corresponds to in-distribution dataset, while red corresponds to OOD dataset.

(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 7. JEM++ generated class-conditional samples of **Plane**

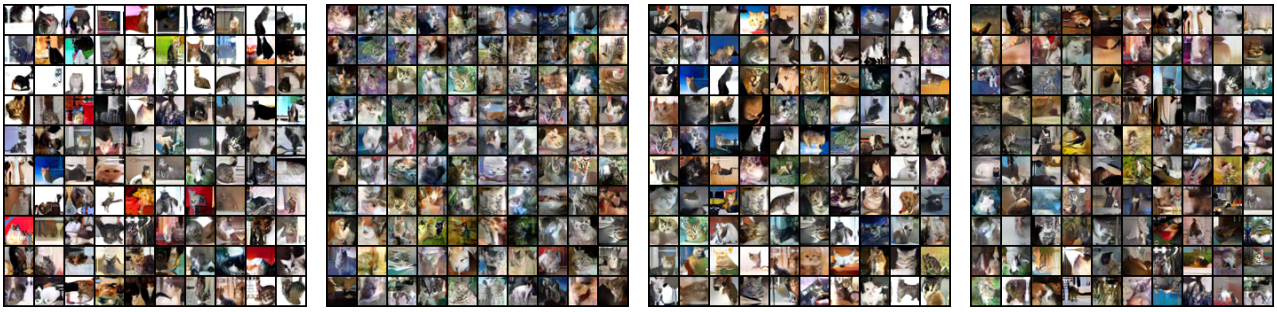
(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 8. JEM++ generated class-conditional samples of **Car**



(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 9. JEM++ generated class-conditional samples of **Bird**



(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

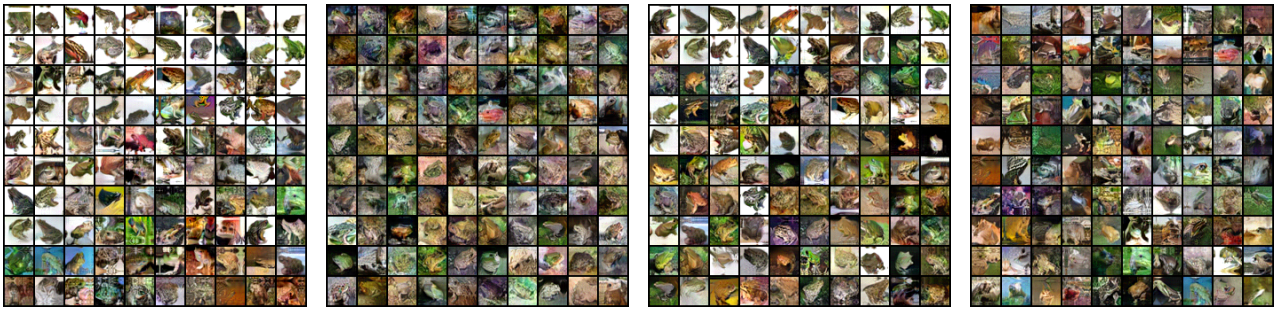
Figure 10. JEM++ generated class-conditional samples of **Cat**

(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 11. JEM++ generated class-conditional samples of **Deer**

(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 12. JEM++ generated class-conditional samples of **Dog**



(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 13. JEM++ generated class-conditional samples of **Frog**

(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 14. JEM++ generated class-conditional samples of **Horse**

(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 15. JEM++ generated class-conditional samples of **Ship**

(a) Samples with highest $p(\boldsymbol{x})$ (b) Samples with lowest $p(\boldsymbol{x})$ (c) Samples with highest $p(y|\boldsymbol{x})$ (d) Samples with lowest $p(y|\boldsymbol{x})$

Figure 16. JEM++ generated class-conditional samples of **Truck**

[8] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*, 2017. 2

[9] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017. 2

[10] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations(ICLR)*, 2016. 2

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015. 1

[12] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 1

[13] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[14] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*, 2018. 2

[15] Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. *arXiv preprint arXiv:1707.04131*, 2017. 2

[16] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 1951. 1

[17] Shibani Santurkar, Andrew Ilyas, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Image synthesis with a single (robust) classifier. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[18] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014. 2

[19] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *The British Machine Vision Conference (BMVC)*, 2016. 1