UNIVERSITY OF CALIFORNIA

Los Angeles

Adversarial Attacks and Defense using Energy-Based Image Models

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Computer Science

by

Jonathan Craig Mitchell

2022

ABSTRACT OF THE THESIS

Adversarial Attacks and Defense using Energy-Based Image Models

by

Jonathan Craig Mitchell
University of California, Los Angeles, 2022
Professor Song-Chun Zhu, Chair

In this article we briefly review current research in adversarial attacks and defenses and form a basis for a theoretical explanation as to why a generative energy model is the solution to the defense problem as it exists for securing naturally trained classifiers. We further expand on this topic and discuss future efforts toward the use of a generalized adversarial defense framework based on Stochastic Security to defend against the strongest known adversarial attacks. We further expand on this idea and demonstrate that Energy-based models can be extended towards multiple tasks and datasets. Furthermore, we discuss some architectural improvements to the framework that lead to improvements in synthesis and defense (The Hat-EBM and the Fixer). This work lies at the intersection of generative modeling, adversarial defense, and chaotic dynamics.

The thesis of Jonathan Craig Mitchell is approved.

University of California, Los Angeles

2022

To Dad, Mom, Beryl, Kate, Daniel, Carnie, Ellie and my late Uncle Barry.
who have been always so supportive
during my Ph.D.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

Chapters 2 and 3 of this work are based on material from:

Hill, M.*, **Mitchell, J.*** & Zhu, S. C. (2021). Stochastic Security: Adversarial Defense Using Long-Run Dynamics of Energy-Based Models. International Conference on Learning Representations (* = equal contributions)

Hill, M.*, **Mitchell, J.***, Chen, C., Du, Y., Shah, M. & Zhu, S. C. (2022). EBM Life Cycle: MCMC Strategies for Synthesis, Defense, and Density Modeling. Arxiv (* = equal contributions)

These works were done in equal collaboration with Mitch Hill. His focus was primarily on achieving a better EBM model for synthesis and convergent learning while my efforts extended these models toward adversarial defense. Prof. Song-Chun Zhu acted as project director.

Chapter 6: Hat-EBM is based on material from:

Hill, M., Nijkamp, E., Pang, B., **Mitchell, J.**, & Zhu, S. C. (2022). Learning Probabilistic Models from Generator Latent Spaces with Hat EBM. Neurips 2022

For the HAT-EBM my role was to figure out how to use the conditional models to separate distributions. Mitch Hill laid the foundational ground work for the model. Erik Nijkamp and Bo Pang worked on the retrofit and refinement experiments. Song-Chun Zhu provided guidance along the way.

I express by deepest thanks to Prof. Song-Chun Zhu who took a chance on me and gave me freedom and support to pursue research in Adversarial Defense. Prof. Zhu provided me with a framework for thinking in high dimensional spaces. He taught me to be relentless in my research endeavors. To pursue what I know to be true. This is the only reason Stochastic Security was successful. It took me almost 9 months from the time I started coding until I had a result worthy of publication. In theory I knew it would work but I had to refine my formulation and experiments again and again to actually obtain a great result and Prof. Zhu was with me along the way. Stochastic Security would not have happened without the support of Prof Zhu and the environment that VCLA lab fosters in its students.

I am also extremely grateful to have learned and had meaningful research discussions with Prof. Yingnian Wu. Prof. Wu taught me along with many other VCLA students many current and background topics in AI research, how to think like a Scientist, and most importantly where the algorithms we use today came from. He helped students in our lab stay on the cutting edge and I also thank him for serving on both my MS and Ph.D committee.

I would also like to thank Prof. Demetri Terzopoulos for teaching me computer graphics, animation modeling, swarm behavior and simulation. I took many graduate courses with Prof. Terzopoulos and I am grateful for his instruction and for his willingness to serve on my MS and Ph.D committees.

I would also like to thank Prof. Cho-Jui Hsieh who taught me a lot about adversarial attacks and defenses. I am fortunate that I had access to someone in my department who was actively teaching and investigating the subject I was doing research on, and that they were willing to provide mentorship to me as well as serve on my MS and Ph.D committee.

I would also like to thank Dr. Mitch Hill who has been collaborating with me on many different research projects for over four years. Mitch taught me most of what I know about Energy-Based Models. He brought me into the mix when I was a first year Ph.D student we've been working together ever since. I hope to continue collaborating on new projects in the future.

I also thank the entire VCLA lab at UCLA and all the members who I have had fruitful discussions with including: Arjun R. Akula, Eric Fischer, Xiaofeng Gao, Steven Gong, Muzhi Han, Siyuan Huang, Mitch Hill, Ziyuan Jiao, Yuan Lu, Tengyu Liu, Xiaojian Ma, Jonathan Mitchell, Erik Nijkamp, Siyuan Qi, Liang Qiu, Shuwen Qiu, Yuxin Qiu, Feng Shi, Shu Wang, Sirui Xie, Xu Xie, Yifei Xu, Luyao Yuan, Victor Zhang, Zeyu Zhang,Tianyang Zhao, Yizhou Zhao and Zilong Zheng.

I also thank my co-authors: Mitch Hill, Yuan Du, Chu Chen, Mubarak Shah, Erik Nijkamp and Bo Pang. I hope to continue our work in the future!

VITA

2009–2015	B.Eng. (Electrical Engineering), UC Davis, Davis, California
2018–2019	M.S. (Computer Science), UCLA, Los Angeles, California
2020	Research Scientist Intern, Nvidia, Los Angeles, California
2021	Research Scientist Intern, Nvidia, Los Angeles, California

PUBLICATIONS

Richeimer, J, & **Mitchell, J.** (2018). Bounding Box Embedding for Single Shot Person Instance Segmentation. Arxiv

Mitchell, J. & Zhu, S. C. (2020). Toward Adversarial Defense using Energy-based Models. ProQuest ID: Mitchell_ucla_0031N_18547

Hill, M.*, **Mitchell, J.*** & Zhu, S. C. (2021). Stochastic Security: Adversarial Defense Using Long-Run Dynamics of Energy-Based Models. International Conference on Learning Representations (* = equal contributions)

Hill, M.*, **Mitchell, J.***, Chen, C., Du, Y., Shah, M. & Zhu, S. C. (2022). EBM Life Cycle: MCMC Strategies for Synthesis, Defense, and Density Modeling. Arxiv (* = equal contributions)

Hill, M., Nijkamp, E., Pang, B., **Mitchell, J.**, & Zhu, S. C. (2022). Learning Probabilistic Models From Generator Latent Spaces With Hat EBM. Neurips 2022.

CHAPTER 1

Introduction

1.1 Introduction

Security and safety of machine learning systems is paramount due to their increased adoption in modern society. Deep neural networks have a variety of use cases and trained DNN models are being used in autonomous vehicle perception, person identification, fraud detection, and Natural Language Processing.

The goal of this work is to create an adversarial robust purification method to remove adversarial signals from a perturbed image for the task of image classification. This work leverages the use of an MCMC-based energy model as an auxillary purification tool to remove adversarial signals.

This thesis is broken down into three pillars. Foundation, Scaling, and Extension. Foundation introduces the **Stochastic Security** framework which lays the foundation for how to use an Energy-Based Model to defend against adversarial attacks. Scaling demonstrates how we can scale such as model to defend against larger datasets namely ImageNet. This was part of the **EBM Life Cycle work** [HMC22a]. Extension is the most recent research, which demonstrates how we can extend a trained model to new datasets and tasks (**Universal Defender**). We also extend the purification method in **The Fixer**, which is a Generator-like model that removes impurities from purified images after langevin sampling before they are seen by the classifier. In parallel I also worked on the **Hat EBM** [MZ23], which constructs a new EBM architecture that encapsulates existing and co-trained generator models for image refinement and synthesis. Evolution of my research can be seen in figure 1.1.

Throughout this work we come back to the central hypothesis: "Can we use EBM's to defend against adversarial attacks?". We answer the following questions throughout the text:

- Why do they work? What about descriptive models (DM) enables them to remove adversarial signals?
- Capacity: How big can we make these models? What architectures can be leveraged to produce them?
- Domain transfer: Can EBM's internal knowledge translate between tasks and datasets?

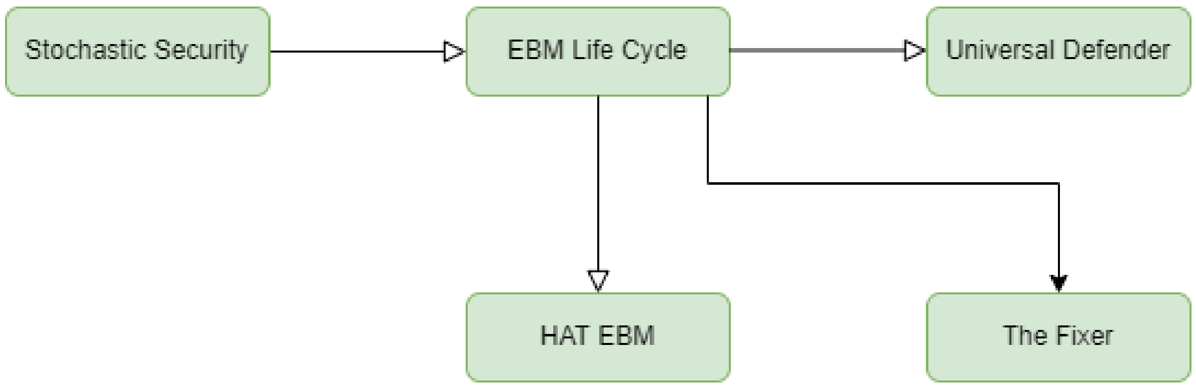


Figure 1.1: Above presents the scope of work for this dissertation. We started off with Stochastic Security where we built the problem and framework [HMZ21]. We then scaled these models in the EBM life cycle work [HMC22a]. This also led to the Hat EBM [MZ23]. After scaling we extended the framework to defend against multiple tasks and datasets. After this we came up with "The Fixer" as an auxillary network to help fix misclassification caused by Langevin sampling.

1.2 Adversarial attacks

A seemingly benign change to an input of a trained state of the art classifier can cause the classifier to be fooled. The adversary targets the input to create an adversarial sample indistinguishable from the original input as shown in figure 1.2. This is known as an adversarial attack; an algorithm that perturbs an image to fool a classifier. This may cause serious security issues as vision algorithms integrate into our daily lives.

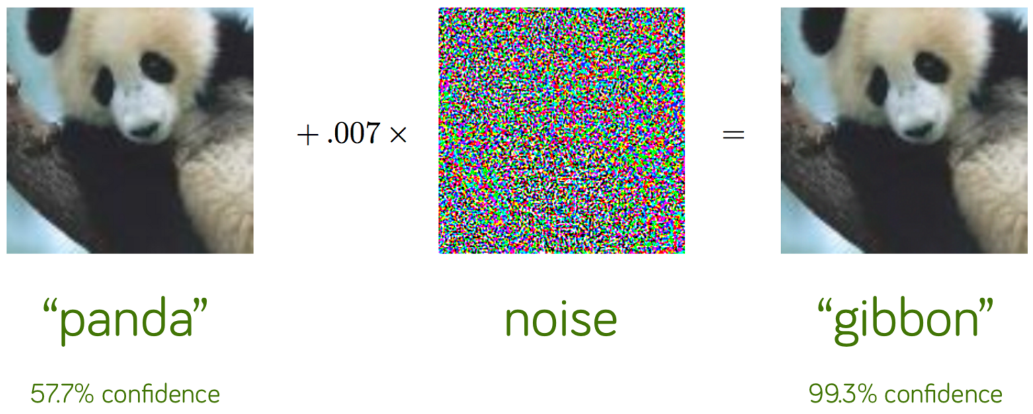


Figure 1.2: Adversarial signal added to a simple panda image using the FGSM attack (equation 1.2) applied to GoogLeNet[SLJ14] trained on Imagenet [RDS14]

1.2.1 What are adversarial attacks

There are three categories of adversarial attacks with respect to vision and images. In increasing order of strength these methods are widely known as "black box", "transfer" and "white box" attacks respectively. Black box attacks are aware of a targets task, dataset, and training environment (hyper parameters and tuning variables), but they are not aware of the model parameters (weights). Transfer attacks utilize gradient-based information of a surrogate model trained in the same environment as the target model and attempt to transfer the attack to the target. A white box attack has direct access to the model's parameters and utilizes gradient based information to uniquely target each specific model. This work is primarily concerned with white box attacks, specifically FGSM, PGD, and BPDA attacks.

1.2.1.1 Formulation

Consider the saddle point optimization formulation from [MMS18]. Given a dataset $\{X_i\}_{i=1}^{n}$ where $X_i \in R^D$ with underlying data distribution q. Natural image training based on empirical risk minimization seeks to minimize $E_q[L(x, y, \theta)]$ where $x \in X$ and $y \in R^k$ are labels with k classes and θ are the classifiers trainable parameters. However, simple empirical risk minimization will not provide an adversarially robust classifier (as shown in fig 1.2). Therefore our goal is to train the classifier under the following optimization criteria:

$$\min_{\theta} E_{(x,y)\in q}[\max_{\delta\in S} L(x+\delta, y, \theta)] \tag{1.1}$$

Where $S \in R^D$ is the set of allowed pixel perturbation around the original image constraints by the l_∞ norm which is considered to be an $\epsilon - ball$ around x. δ is one of those perturbations such that $x + \delta \in S$. $L(x + \delta, y, \theta)$ is the classifiers loss function. This saddle point formulation can be decomposed into the inner maximization portion; whose goal is to create "true adversaries" that are able to fool the classifier. The outer minimization portion is tasked to limit the amount of these adversaries and create a robust classifier that can't be fooled. Robustness is a heuristic used to measure the accuracy of a classifier with respect to adversarial samples.

1.2.1.2 Targeted vs Untargeted attacks

There is also a distinction among adversarial attack destinations. Given an input x_i with label y_i, a targeted attack is one where the adversary attempts to perturb the input x_i so that the classifier predicts class y_j st $i \neq j$ where j is a specific class target. An untargeted attack creates an adversary to increase $L(x, y, \theta)$ solely to cause misclassification of y_i without any specific "targeted" class in mind. In both cases the perturbation should be "imperceptible" such that the original image and the adversarial image can not be distinguished by humans.

1.2.1.3 Specific attacks

In this work we describe three different untargeted attacks. The Fast Sign Gradient Method (FGSM) attack is an l_∞ bounded adversarial algorithm from [MMS18] that computes adversarial examples using eq (1.2)

$$\hat{x} = x + \epsilon sgn(\nabla_x L(\theta, x, y)) \tag{1.2}$$

where L represents the loss function after a forward pass of the network and $\hat{x} = x + \delta$ is the adversarially perturbed image, x corresponds to the original image, y the class label, θ the model parameters, and ϵ is the constraint of allowed perturbation of each pixel with respect to the l_∞ norm. We also borrow a variant of FSGM from [MMS18] known as Projected Gradient Descent (PGD), that iteratively attacks each newly formed adversarial image and projects it back to the l_∞ constrained ϵ-ball around the original image x

$$\hat{x}_{i+1} = \Pi_{x+S}(\hat{x}_i + \alpha sgn(\nabla_x L(\theta, x, y))) \tag{1.3}$$

where α is the learning rate and where the space of allowed pixel perturbations on x is S specified by the aforementioned l_∞-ball around x. This ensures that the difference between x and the adversarial image $\hat{x}_{i+1}$ (which has gone through multiple attacks) is imperceptible. Both of these attacks are considered white box because they utilize gradient information of the model and untargeted because they are not being pushed towards a specific class.

Currently, the most robust form of adversarial defense against equationss 1.2, 2.8 is to train a classifier on adversarial samples, as shown in [MMS18]. However, this detracts from the original task of the model in that it does not increase task performance (natural image classification) and increases both training time and computational load. For this purpose, auxillary white box defense methods that do not require classifiers to undergo adversarial training have been explored such as [SKN18], [SKC18], etc. We will refer to these methods as "add-on purification" and their defense algorithms as purifiers. These purifiers proved hopeful until further analysis by the authors of [ACW18] revealed that the majority of these methods were simply adding non-differentiable components/layers to existing classifiers which caused them to "obfuscate" their gradients and create weak adversarial samples during testing.

To combat this "false sense of security" the authors of [ACW18] created a Backwards Pass Differentiable Attack (BPDA) which is a straight-through attack algorithm that is able to differentiate through add-on purifiers to the core network in order to create adversarial samples [ACW18]. The approach consists of performing a forward pass on the network in standard fashion and simply replacing the purifier with the identity on backwards pass differentiation.

$$\hat{x}_{i+1} = \Pi_{x+S}(\hat{x}_i + \alpha sgn(\nabla_x L(\theta, f(g(x)), y)) \tag{1.4}$$

Eq 1.4 provides an approximation of the true gradient because on average $g(x) \approx x$. However, this also requires that more iterations of the attack are performed because $g(x)$ is treated as an approximation of the true gradient on each step. The function $g(x)$ is the purifier in this case. We can treat the output as $x_p \leftarrow g(x)$ where x_p is a purified image. We perform the same projection as in eq 2.8 after the perturbation.

1.3 Energy-Based Models

Throughout this work we come back to Energy-Based models, what they are, how they work, and what applications they can be used for.

1.3.1 Formulation

To present a formal definition of the models used herein, we begin with an energy-based Gibbs-Boltzmann density and propose the formulation as seen in [NHH19], [BZ19].

$$p_\theta(x) = \frac{1}{Z(\theta)} exp\{-U(x;\theta)\}$$

where x is an image signal and $U(x;\theta)$ is an energy potential that belongs to a family of distributions $P = \{p_\theta\}_{\theta \in \Theta}$

Stochastic gradients are useful in cases where the partition function $Z(\theta) = \int_X exp\{-U(x;\theta)\}dx$ is intractable. Our goal in using this energy-based model is to synthesize realistic images $x \sim p_\theta(x)$ to be as close as possible to the true data distribution $q(x)$. In doing so we formulate our loss function as

$$\min_\theta KL(q||p_\theta) = \min_\theta E_q \left[log \frac{q(x)}{p_\theta(x)} \right] \tag{1.5}$$

$$\min_\theta E_q[\log q(x)] - E_q[\log p_\theta(x)] \tag{1.6}$$

where E_q does not depend on θ. Additionally, for an i.i.d dataset $\{X_i\}_{i=1}^n$, using the law of large numbers we can approximate the expectation of the true underlying distribution $E_q[\log p_\theta(x)] \approx \frac{1}{n}\sum_{i=1}^n \log p_\theta(X_i)$ therefore

$$= \min_\theta -E_q[\log p_\theta(x)] \tag{1.7}$$

$$= \max_{\theta} E_q[\log p_\theta(x)] \tag{1.8}$$

and therefore minimizing $KL(q||p)$ also maximizes the log likelihood of $p_\theta(x)$ which is equivalent to minimizing the negative log likelihood. The likelihood $l(x|\theta)$:

$$\min_{\theta} l(x|\theta) = \min_{\theta} \log(Z(\theta) + E_q[U(X;\theta)] \tag{1.9}$$

We can approximate the intractable partition function using the gradient of $\log Z(\theta)$ which can be expressed in closed form as $\nabla_\theta \log Z(\theta) = -E_{p_\theta}[\nabla_\theta U(X;\theta)]$ [GBC16], thus we can minimize $l(\theta)$ by taking the derivative

$$\frac{\partial}{\partial_\theta} l(x|\theta) = \frac{\partial}{\partial_\theta} E_q[U(X;\theta)] - E_{p_\theta}[\frac{\partial}{\partial \theta} U(X;\theta)] \tag{1.10}$$

$$\approx \frac{\partial}{\partial_\theta}\big(\frac{1}{n}\sum_{i=1}^{n} U(X_i^+;\theta) - \frac{1}{m}\sum_{i=1}^{m} U(X_i^-;\theta)\big) \tag{1.11}$$

where $U(X_i^+;\theta)$ are known as positive samples that follow the true underlying distribution of the data q and where $U(X_i^-;\theta)$ are known as negative samples obtained using MCMC from the models currently learned distribution $p_\theta(x)$. [NHH19]. Positive samples are simply randomly sampled training images while the negative (MCMC) samples are obtained using Langevin dynamics. The advantage of using an energy-based model is that it does not have to approximate the partition function because it simply tries to create "realistic" synthesized images from our model and compare them to the data itself. Thus MLE forces the MCMC samples from the model $p_\theta(x)$ to be as close to q as possible.

For applications throughout this body of work the underlying architecture of the EBM model $p_\theta(x)$ changes from simple lightweight Convolutional Neural Networks [HMZ21] to larger SNGAN based architectures [MKK18] and ResNets [ZK16]. We consider a forward pass of the network $f(x;\theta) = -U(x;\theta)$ and where $U(x;\theta) \in R$.

There are also several training strategies to achieve different EBM models. We have short-run models for image synthesis, mid-run models for adversarial defense, and long-run models for probabilistic density estimation [HMC22a].

1.3.1.1 Langevin dynamics

We utilize the Langevin equation to perform Langevin Monte Carlo (LMC) sampling of the model during training to obtain the negative samples. LMC is a special case of HMC that is used when the trajectory to propose a new state consists of one leapfrog step. [Nea12]. In typical LMC we sample momentum variable values from their zero mean and unit variance Gaussian white noise distribution $Z_i \sim N(0, I)$.

$$X_i^* = X_i - \frac{\varepsilon^2}{2}\frac{\partial}{\partial x}U(X_i;\theta) + \varepsilon Z_i \tag{1.12}$$

Where $\varepsilon > 0$ is a constant noise factor. According to the work of [CFG14], [NHH19], the momentum update of the second HMC variable as well as the Metropolis-Hastings update step can be ignored in practice.

Energy-Based Models are related to our adversarial defense problem via the purification triangle in fig 1.3

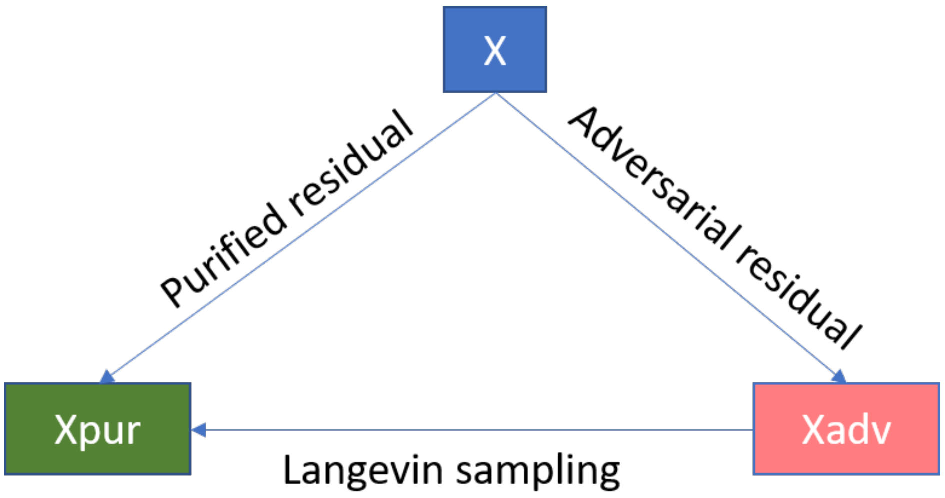


Figure 1.3: Purification Triangle where Langevin sampling represents stochastic Langevin dynamics in eq 1.12 where X_{pur} are purified samples and X_{adv} are the adversaries created from X_{pur} using eq 1.4

For EBM-based defenses it is useful to observe that the added noise introduces stochastic behavior into the algorithm which contributes to the stochastic vector in the purification triangle 1.3. This makes it difficult for any attack to clearly target the a stochastic purifier and create strong adversaries. This is highlighted in greater detail in Chapter 2: Stochastic Security.

CHAPTER 2

Foundation: Stochastic Security

2.1 Stochastic Security

This chapter presents stochastic security: a framework designed to secure naturally trained classifiers against adversarial white box attacks. The vulnerability of deep networks to adversarial attacks is a central problem for deep learning from the perspective of both cognition and security. The current most successful defense method is to train a classifier using adversarial images created during learning. Another defense approach involves transformation or purification of the original input to remove adversarial signals before the image is classified. We focus on defending naturally-trained classifiers using Markov Chain Monte Carlo (MCMC) sampling with an Energy-Based Model (EBM) for adversarial purification. In contrast to adversarial training, our approach is intended to secure highly vulnerable pre-existing classifiers. To our knowledge, no prior defense method is capable of securing naturally-trained classifiers, and our method is the first to validate a post-training defense approach that is distinct from current successful defenses which modify classifier training.

The memoryless behavior of long-run MCMC sampling will eventually remove adversarial signals, while metastable behavior preserves consistent appearance of MCMC samples after many steps to allow accurate long-run prediction. Balancing these factors can lead to effective purification and robust classification. We evaluate adversarial defense with an EBM using the strongest known attacks against purification. Our contributions are 1) an improved method for training EBM's with realistic long-run MCMC samples for effective purification, 2) an Expectation-Over-Transformation (EOT) defense that resolves ambiguities for evaluating stochastic defenses and from which the EOT attack naturally follows, and 3) state-of-the-art adversarial defense for naturally-trained classifiers and competitive defense compared to adversarial training on CIFAR-10, SVHN, and CIFAR-100.

2.2 Motivation and Contributions

Deep neural networks are highly sensitive to small input perturbations. This sensitivity can be exploited to create adversarial examples that undermine robustness by causing trained networks to produce defective results from input changes that are imperceptible to the human eye [GSS15]. The adversarial scenarios studied in this paper are primarily untargeted white-box attacks on image classification networks. White-box attacks have full access to the classifier (in particular, to classifier gradients) and are the strongest attacks against the majority of defenses.

Many whitebox methods have been introduced to create adversarial examples. Strong iterative attacks such as Projected Gradient Descent (PGD) [MMS18] can reduce the accuracy of a naturally-trained classifier to virtually 0. Currently the most robust form of adversarial defense is to train a classifier on adversarial samples in a procedure known as *adversarial training* (AT) [MMS18]. Another defense strategy, which we will refer to as *adversarial preprocessing* (AP), uses defensive transformations to purify an image and remove adversarial signals before classification ([SKN18, SMM19, CRK19, YZK19], and others). AP is an attractive strategy compared to AT because it has the potential to secure vulnerable pre-existing classifiers. Defending naturally-trained classifiers is the central focus of this work.

[ACW18] revealed that many preprocessing defenses can be overcome with minor adjustments to the standard PGD attack. Both stochastic behavior from preprocessing and the computational difficulty of end-to-end backpropagation can be circumvented to attack the classifier through the defensive transformation. In this paper we carefully address [ACW18] to evaluate AP with an EBM using attacks with the greatest known effectiveness and efficiency.

Langevin sampling using an EBM with a ConvNet potential [XLZ16] has recently emerged as a method for AP [DM19, GWJ20]. However, the proposed defenses are not competitive with AT (see Table 2.1 and [CH20]). In the present work we demonstrate that EBM defense

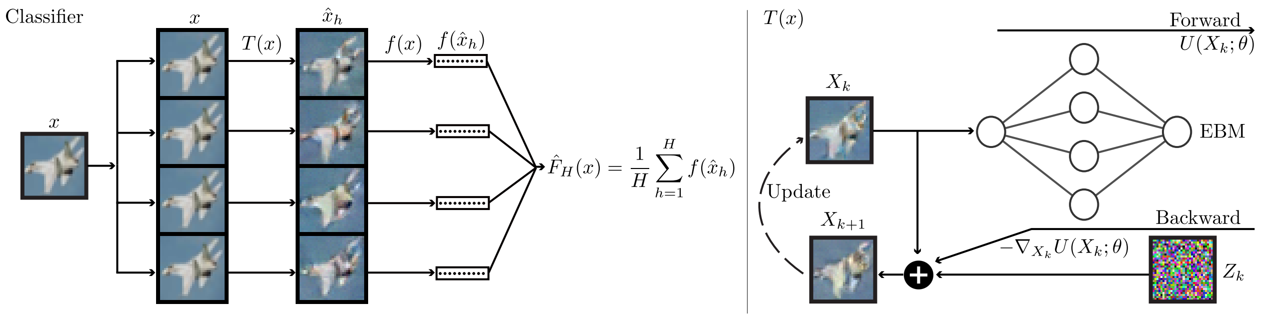


Figure 2.1: *Left:* Visualization of calculating our stochastic logits $\hat{F}_H(x)$ from (5.2). The input image x is replicated H times and parallel Langevin updates with a ConvNet EBM are performed on each replicate to generate $\{\hat{x}_h\}_{h=1}^{H}$. Purified samples are sent in parallel to our naturally-trained classifier network $f(x)$ and the resulting logits $\{f(\hat{x}_h)\}_{h=1}^{H}$ are averaged to produce $\hat{F}_H(x)$. The logits $\hat{F}_H(x)$ give an approximation of our true classifier logits $F(x)$ in (2.4) that can be made arbitrarily precise by increasing H. *Right:* Graphical diagram of the Langevin dynamics (6.3) that we use for $T(x)$. Images are iteratively updated with a gradient from a naturally-trained EBM (6.1) and Gaussian noise Z_k.

of a naturally-trained classifier can be stronger than standard AT [MMS18] and competitive with state-of-the-art AT [ZYJ19, CRS19].

Our defense tools are a classifier trained with labeled natural images and an EBM trained with unlabeled natural images. For prediction, we perform Langevin sampling with the EBM and send the sampled images to the naturally-trained classifier. An intuitive visualization of our defense method is shown in Figure 2.1. Langevin chains constitute a memoryless trajectory that removes adversarial signals, while metastable sampling behaviors preserve image classes over long-run trajectories. Balancing these two factors leads to effective adversarial defense. Our main contributions are:

- A simple but effective adjustment to improve the convergent learning procedure from [NHH20]. Our adjustment enables realistic long-run sampling with EBMs learned from complex datasets such as CIFAR-10.
- An Expectation-Over-Transformation (EOT) defense that prevents the possibility of a stochastic defense breaking due to random variation in prediction instead of an

adversarial signal. The EOT attack [ACW18] naturally follows from the EOT defense.

- Experiments showing state-of-the-art defense for naturally-trained classifiers and competitive defense compared to state-of-the-art AT.

2.3 Related Work

Adversarial training learns a robust classifier using adversarial images created during each weight update. The method is introduced by [MMS18]. Many variations of AT have been explored, some of which are related to our defense. [HRF19] apply noise injection to each network layer to increase robustness via stochastic effects. Similarly, Langevin updates with our EBM can be interpreted as a ResNet [HZR16] with noise injected layers as discussed by [NHZ19a]. Semi-supervised AT methods [AUH19, CRS19] use unlabeled images to improve robustness. Our EBM also leverages unlabeled data for defense.

Adversarial preprocessing is a strategy where auxiliary transformations are applied to adversarial inputs before they are given to the classifier. Prior forms of pre-processing defenses include rescaling [XWZ18], thermometer encoding [BRR18], feature squeezing [XEQ18], activation pruning [DAB18], reconstruction [SKC18], ensemble transformations [RSF19], addition of Gaussian noise [CRK19], and reconstruction of masked images [YZK19]. Prior preprocessing methods also include energy-based models such as Pixel-Defend [SKN18] and MALADE [SMM19] that draw samples from a density that differs from (6.1). It was shown by [ACW18, TCB20] that many preprocessing defenses can be totally broken or dramatically weakened by simple adjustments to the standard PGD attack, namely the EOT and BPDA techniques. Furthermore, many preprocessing defenses such as [CRK19, RSF19, YZK19] also modify classifier training so that the resulting defenses are analogous to AT, as discussed in 2.13. No prior preprocessing defense is competitive with AT when applied to a naturally-trained classifier.

Energy-based models are a probabilistic method for unsupervised modeling. Early energy-based image models include the FRAME [ZWM98] and RBM [Hin02]. The EBM

(6.1) used in this work is introduced by [XLZ16] and important observations about the learning process are presented by [NHH20, NHZ19a]. Preliminary investigations for using the EBM (6.1) for adversarial defense are presented by [DM19, GWJ20] but the results are not competitive with AT (see Section 2.6.1). Our work builds on the convergent learning methodology from [NHH20] to apply long-run Langevin sampling as a defense technique.

2.4 Improved Convergent Learning of Energy-Based Models

The Energy-Based Model introduced in [XLZ16] is a Gibbs-Boltzmann density

$$p(x;\theta) = \frac{1}{Z(\theta)} \exp\{-U(x;\theta)\} \tag{2.1}$$

where $x \in \mathbb{R}^D$ is an image signal, $U(x;\theta)$ is a ConvNet with weights θ and scalar output, and $Z(\theta) = \int_{\mathcal{X}} \exp\{-U(x;\theta)\}dx$ is the intractable normalizing constant. Given i.i.d. samples from a data distribution $q(x)$, one can learn a parameter θ^* such that $p(x;\theta^*) \approx q(x)$ by minimizing the expected negative log-likelihood $\mathcal{L}(\theta) = E_q[-\log p(X;\theta)]$ of the data samples. Network weights θ are updated using the loss gradient

$$\nabla\mathcal{L}(\theta) \approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta U(X_i^+;\theta) - \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta U(X_i^-;\theta) \tag{2.2}$$

where $\{X_i^+\}_{i=1}^n$ are a batch of training images and $\{X_i^-\}_{i=1}^m$ are i.i.d. samples from $p(x;\theta)$ obtained via MCMC. Iterative application of the Langevin update

$$X_{k+1} = X_k - \frac{\tau^2}{2}\nabla_{X_k} U(X_k;\theta) + \tau Z_k, \tag{2.3}$$

where $Z_k \sim \mathrm{N}(0, I_D)$ and $\tau > 0$ is the step size parameter, is used to obtain the samples $\{X_i^-\}_{i=1}^m$.

[NHH20] reveal that EBM learning heavily gravitates towards an unexpected outcome where short-run MCMC samples have a realistic appearance and long-run MCMC samples have an unrealistic appearance. The work uses the term *convergent learning* to refer to the expected outcome where short-run and long-run MCMC samples have similar appearance, and the term *non-convergent learning* to refer to the unexpected but prevalent outcome where

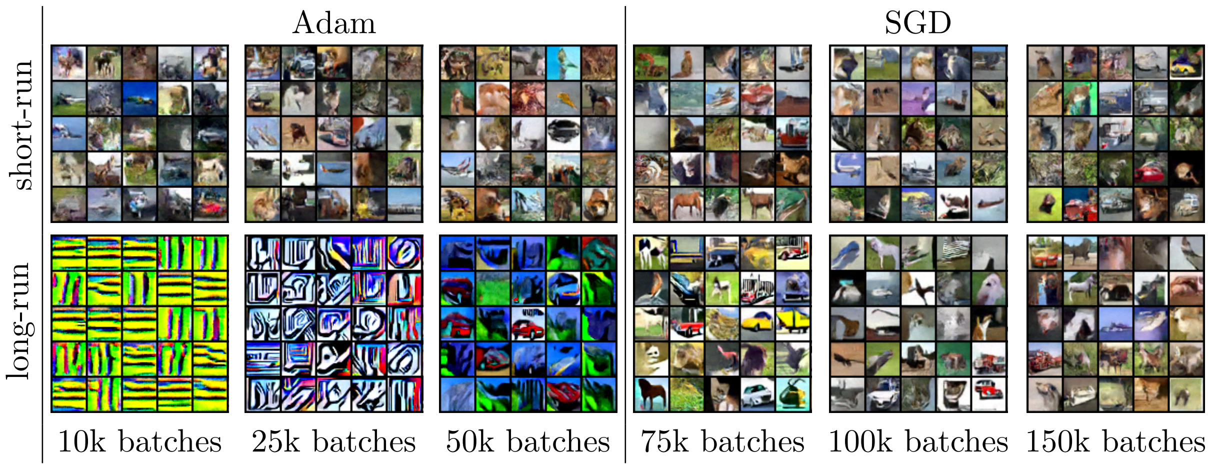


Figure 2.2: Comparison of long-run and short-run samples over model updates for our improved method of convergent learning. The model is updated in a non-convergent learning phase with the Adam optimizer for the first 50,000 batches. The majority of short-run synthesis realism is learned during this phase, but the long-run samples are very unrealistic. The second learning phase uses SGD with a low learning rate. Short-run synthesis changes very little, but the long-run distribution gradually aligns with the short-run distribution.

models have realistic short-run samples and oversaturated long-run samples. Convergent learning is essential for our defense strategy because long-run samples must be realistic for classifiers to maintain high accuracy after Langevin transformation (see Appendix 2.11).

As observed by [NHH20], we were unable to learn a convergent model when updating θ with the Adam optimizer [KB15]. Despite the drawbacks of Adam for convergent learning, it is a very effective tool for obtaining realistic short-run synthesis. Drawing inspiration from classifier training from [KS17], we learn a convergent EBM in two phases. The first phase uses Adam to update θ to achieve realistic short-run synthesis. The second phase uses SGD to update θ to align short-run and long-run MCMC samples to correct the degenerate steady-state from the Adam phase. This modification allows us to learn the convergent EBMs for complex datasets such as CIFAR-10 using 20% of the computational budget of [NHH20]. See Figure 2.2. We use the lightweight EBM from [NHZ19a] as our network architecture.

We use long run chains for our EBM defense to remove adversarial signals while maintaining image features needed for accurate classification. The steady-state convergence property ensures adversarial signals will eventually vanish, while metastable behaviors preserve features of the initial state. Theoretical perspectives on our defense can be found in Appendix 2.10 and a comparison of convergent and non-convergent EBM defenses can be found in Appendix 2.11.

2.5 Attack and Defense Formulation

2.5.1 Classification with Stochastic Transformations

Let $T(x)$ be a stochastic pre-processing transformation for a state $x \in \mathbb{R}^D$. Given a fixed input x, the transformed state $T(x)$ is a random variable over $\mathbb{R}^D$. In this work, $T(x)$ is obtained from K steps of the Langevin update (6.3) starting from $X_0 = x$. One can compose $T(x)$ with a deterministic classifier $f(x) \in \mathbb{R}^J$ (for us, a naturally-trained classifier) to define a new classifier $F(x) \in \mathbb{R}^J$ as

$$F(x) = E_{T(x)}[f(T(x))]. \tag{2.4}$$

$F(x)$ is a deterministic classifier even though $T(x)$ is stochastic. The predicted label for x is then $c(x) = \arg\max_j F(x)_j$. In this work, $f(x)$ denotes logits and $F(x)$ denotes expected logits, although other choices are possible (e.g. softmax outputs). We refer to (2.4) as an Expectation-Over-Transformation (EOT) defense. The classifier $F(x)$ in (2.4) is simply the target of the EOT attack [ACW18]. The importance of the EOT formulation is well-established for adversarial attacks, but its importance for adversarial defense has not yet been established. Although direct evaluation of $F(x)$ is generally impossible, the law of large numbers ensures that the finite-sample approximation of $F(x)$ given by

$$\hat{F}_H(x) = \frac{1}{H}\sum_{h=1}^{H} f(\hat{x}_h) \quad \text{where,} \quad \hat{x}_h \sim T(x) \text{ i.i.d.}, \tag{2.5}$$

can approximate $F(x)$ to any degree of accuracy for a sufficiently large sample size H. In other words, $F(x)$ is intractable but trivial to accurately approximate via $\hat{F}_H(x)$ given enough computation.

In the literature both attackers [ACW18, TCB20] and defenders [DAB18, YZK19, GWJ20]

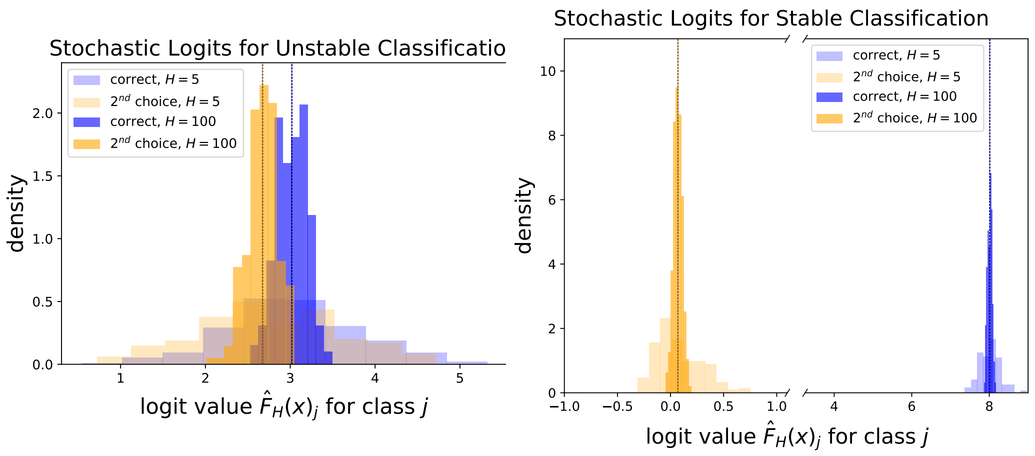


Figure 2.3: The histograms display different realizations of the logits $\hat{F}_H(x)$ for the correct class and the second most probable class for images x_1 (*left*) and x_2 (*right*) over different choices of H. In both cases, $F(x)$ (dashed vertical lines) gives correct classification. However, the overlap between the logit histograms of $\hat{F}_H(x_1)$ indicate a high probability of misclassification even for large H, while $\hat{F}_H(x_2)$ gives correct prediction even for small H because the histograms are well-separated. The EOT defense formulation (2.4) is essential for securing borderline images such as x_1.

evaluate stochastic classifiers of the form $f(T(x))$ using either $\hat{F}_1(x)$ or $\hat{F}_{H_{\text{adv}}}(x)$ where H_{adv} is the number of EOT attack samples, typically around 10 to 30. This evaluation is not sound when $\hat{F}_H(x)$ has a small but plausible chance of misclassification because randomness alone could cause x to be identified as an adversarial image even though $\hat{F}_H(x)$ gives the correct prediction on average (see Figure 2.3). In experiments with EBM defense, we identify many images x that exhibit variation in the predicted label

$$\hat{c}_H(x) = \arg\max_j \hat{F}_H(x)_j \tag{2.6}$$

for smaller $H \approx 10$ but which have consistently correct prediction for larger $H \approx 150$. Fair evaluation of stochastic defenses must be based on the deterministic EOT defense F in (2.4) and attackers must use sufficiently large H to ensure that $\hat{F}_H(x)$ accurately approximates $F(x)$ before declaring that an attack using adversarial image x is successful. In practice, we observe that $\hat{F}_{150}$ is sufficiently stable to evaluate F over several hundred attacks with

Algorithm 1 for our EBM defense.

2.5.2 Attacking Stochastic Classifiers

Let $L(F(x), y) \in \mathbb{R}$ be the loss (e.g. cross-entropy) between a label $y \in \{1, \dots, J\}$ the outputs $F(x) \in \mathbb{R}^J$ of a classifier (e.g. the logits) for an image $x \in \mathbb{R}^D$. For a given pair of observed data (x^+, y), an untargeted white-box adversarial attack searches for the state

$$x_{\text{adv}}(x^+, y) = \arg\max_{x \in S} L(F(x), y) \tag{2.7}$$

that maximizes loss for predicting y in a set $S \subset \mathbb{R}^D$ centered around x^+. In this work, a natural image x^+ will have pixels intensities from 0 to 1 (i.e. $x^+ \in [0,1]^D$). One choice of S is the intersection of the image hypercube $[0,1]^D$ and the l_∞-norm ε-ball around x^+ for suitably small $\varepsilon > 0$. Another option is the intersection of the hypercube with the l_2-norm ε-ball.

The Projected Gradient Descent (PGD) attack [MMS18] is the standard benchmark when S is the ε-ball in the l_p norm. PGD begins at a random $x_0 \in S$ and maximizes (2.7) by iteratively updating x_i with

$$x_{i+1} = \prod_S (x_i + \alpha g(x_i, y)), \quad g(x, y) = \arg\max_{\|v\|_p \leq 1} v^\top \Delta(x, y), \tag{2.8}$$

where Π_S denotes projection onto S, $\Delta(x, y)$ is the attack gradient, and $\alpha > 0$ is the attack step size. Standard PGD uses the gradient $\Delta_{\text{PGD}}(x, y) = \nabla_x L(F(x), y)$.

The EOT attack [ACW18] circumvents the intractability of F by attacking finite sample logits $\hat{F}_{H_{\text{adv}}}$, where H_{adv} is the number of EOT attack samples, with the gradient

$$\Delta_{\text{EOT}}(x, y) = \nabla_x L(\hat{F}_{H_{\text{adv}}}(x), y). \tag{2.9}$$

The EOT attack is the natural adaptive attack for our EOT defense formulation. Another challenge when attacking a preprocessing defense is the computational infeasibility or theoretical impossibility of differentiating $T(x)$. The Backward Pass Differentiable Approximation (BPDA) technique [ACW18] uses an easily differentiable function $g(x)$ such that $g(x) \approx T(x)$ to attack $F(x) = f(T(x))$. One calculates the attack loss using $L(f(T(x)), y)$ on the forward pass but calculates the attack gradient using $\nabla_x L(f(g(x)), y)$ on the backward pass. A simple but effective form of BPDA is the identity approximation $g(x) = x$. This approximation is reasonable for preprocessing defenses that seek to remove adversarial signals while preserving the main features of the original image. When $g(x) = x$, the BPDA attack gradient is $\Delta_{\text{BPDA}}(x, y) = \nabla_z L(f(z), y)$ where $z = T(x)$. Intuitively, this attack obtains an attack gradient with respect to the purified image and applies it to the original image.

Combining the EOT attack and BPDA attack with identity $g(x) = x$ gives the attack gradient

$$\Delta_{\text{BPDA+EOT}}(x, y) = \frac{1}{H_{\text{adv}}} \sum_{h=1}^{H_{\text{adv}}} \nabla_{\hat{x}_h} L\left(\frac{1}{H_{\text{adv}}} \sum_{h=1}^{H_{\text{adv}}} f(\hat{x}_h), y \right), \quad \hat{x}_h \sim T(x) \text{ i.i.d.} \tag{2.10}$$

The BPDA+EOT attack represents the strongest known attack against preprocessing defenses, as shown by its effective use in recent works [SMM19, TCB20]. We use $\Delta_{\text{BPDA+EOT}}(x, y)$ in (2.8) as our primary attack to evaluate the EOT defense (2.4). Pseudo-code for our adaptive attack can be found in Algorithm 1 in Appendix 2.8.

2.6 Experiments

We use two different network architectures in our experiments. The first network is the lightweight EBM from [NHZ19a]. The second network is a 28×10 WideResNet [ZK16] classifier. The EBM and classifier are trained independently on the same dataset. No adversarial training or other training modifications are used for either model. We use the parameters from Algorithm 1 for all evaluations unless otherwise noted. Code and pre-trained models to reproduce these results are publicly available.

2.6.1 PGD Attack from Base Classifier for CIFAR-10 Dataset

We first evaluate our defense using adversarial images created from a PGD attack on the classifier $f(x)$. Since this attack does not incorporate the Langevin sampling from $T(x)$, the adversarial images in this section are relatively easy to secure with Langevin transformations. This attack serves as a benchmark for comparing our defense to the IGEBM [DM19] and JEM [GWJ20] models that also evaluate adversarial defense with a ConvNet EBM (6.1). For all methods, we evaluate the base classifier and the EBM defense for $K = 10$ Langevin steps (as in prior defenses) and $K = 1500$ steps (as in our defense). The results are displayed in Table 2.1.

Table 2.1: CIFAR-10 accuracy for our EBM defense and prior EBM defenses against a PGD attack from the base classifier $f(x)$ with l_∞ perturbation $\varepsilon = 8/255$. (**evaluated on 1000 images*)

	Base Classifier $f(x)$		EBM Defense, $K = 10$		EBM Defense, $K = 1500$	
	Nat.	Adv.	Nat.	Adv.	Nat.	Adv.
Ours	0.9530	0.0000	0.9586	0.0001	0.8412	0.7891
[DM19]	0.4714	0.3219	0.4885	0.3674	0.487*	0.375*
[GWJ20]	0.9282	0.0929	0.9093	0.1255	0.755*	0.238*

Our natural classifier $f(x)$ has a high base accuracy but no robustness. The JEM base classifier has high natural accuracy and minor robustness, while the IGEBM base classifier has significant robustness but very low natural accuracy. Short-run sampling with $K = 10$ Langevin steps does not significantly increase robustness for any model. Long-run sampling with $K = 1500$ steps provides a dramatic increase in defense for our method but only minor increases for the prior methods. Further discussion of the IGEBM and JEM defenses can be found in Appendix 2.14.

2.6.2 BPDA+EOT Attack

In this section, we evaluate our EBM defense using the adaptive BPDA+EOT attack (3.3) designed specifically for our defense approach. This attack is recently used by [TCB20] to evaluate the preprocessing defense from [YZK19] that is very similar to our method.

CIFAR-10. We ran 5 random restarts of the BPDA+EOT attack in Algorithm 1 with the the listed parameters on the entire CIFAR-10 test set. In particular, the attacks use

Table 2.2: Defense vs. whitebox attacks with l_∞ perturbation $\varepsilon = 8/255$ for CIFAR-10.

Defense	$f(x)$ Train Ims.	$T(x)$ Method	Attack	Nat.	Adv.
Ours	Natural	Langevin	BPDA+EOT	0.8412	0.5490
[MMS18]	Adversarial	–	PGD	0.873	0.458
[ZYJ19]	Adversarial	–	PGD	0.849	0.5643
[CRS19]	Adversarial	–	PGD	0.897	0.625
[SKN18]	Natural	Gibbs Update	BPDA	0.95	0.09
[SMM19]	Natural	Langevin	PGD	–	0.0016
[YZK19]	Transformed	Mask + Recon.	BPDA+EOT	0.94	0.15

adversarial perturbation $\varepsilon = 8/255$ and attack step size $\alpha = 2/255$ in the l_∞ norm. One evaluation of the entire test set took approximately 2.5 days using 4x RTX 2070 Super GPUs. We compare our results to a representative selection of AT and AP defenses in Table 2.2. We include the training method for the classifier, preprocessing transformation (if any), and the strongest attack for each defense.

Our EBM defense is stronger than standard AT [MMS18] and comparable to modified AT from [ZYJ19]. Although our results are not on par with state-of-the-art AT [CRS19], our defense is the first method that can effectively secure naturally-trained classifiers.

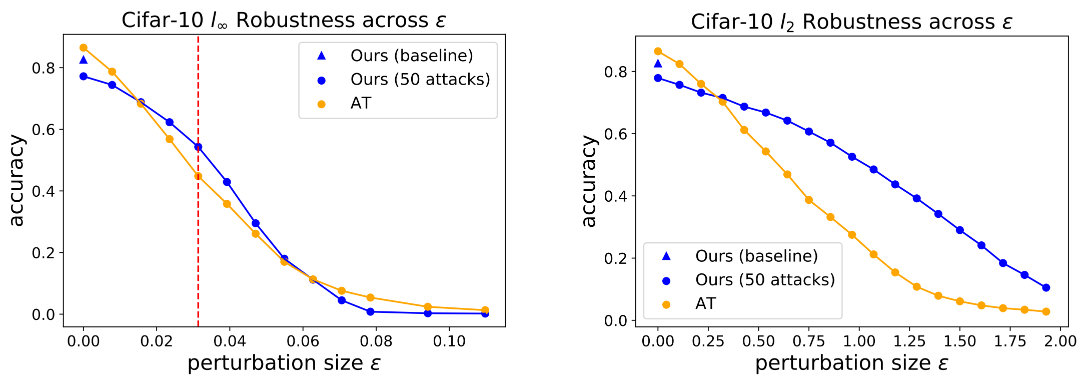


Figure 2.4: Accuracy across perturbation ε for l_∞ and l_2 attacks against our defense and AT [MMS18].

We now examine the effect of the perturbation ε, number of attacks N, and number of EOT attack replicates H_{adv} on the strength of the BPDA+EOT attack. To reduce the computational cost of the diagnostics, we use a fixed set of 1000 randomly selected test images for diagnostic attacks.

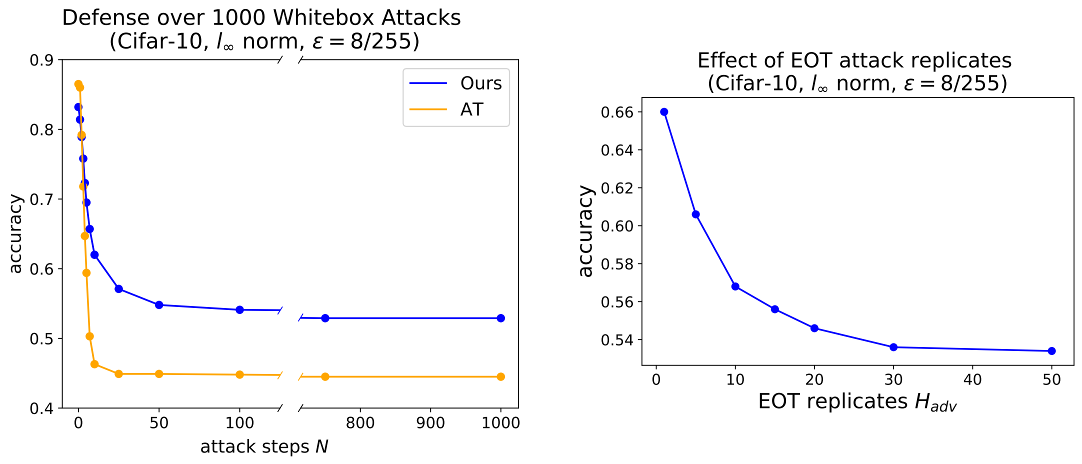


Figure 2.5: Effect of number of attack steps N and number of EOT replicates H_{adv}.

Figure 2.4 displays the robustness of our model and standard AT [MMS18] for l_∞ and l_2 attacks across perturbation size ε. Our model is attacked with BPDA+EOT while the AT model is attacked with PGD. Our defense is more robust than AT for a range of medium-size perturbations in the l_∞ norm and much more robust than AT for medium and large ε in the l_2 norm.

Figure 2.5 visualizes the effect of increasing the computational power of the attacker. The left figure compares our defense with AT over 1000 attacks using an increased number $H_{\text{adv}} = 20$ of attack replicates. The majority of breaks happen within the first 50 attacks as used in the CIFAR-10 experiment in Table 2.2, while a small number of breaks occur within a few hundred attack steps. It is likely that some breaks from long-run attacks are the result of lingering stochastic behavior from $\hat{F}_{H_{\text{def}}}(x)$ rather than the attack itself. The right figure shows the effect of the number of EOT attack replicates over 50 attacks. The strength of the EOT attack saturates when using 20 to 30 replicates. A small gap in attack strength remains between the 15 replicates used in our attacks and the strongest possible attack. Some of this effect is likely mitigated by our use of 5 random restarts.

The diagnostics indicate that the report in Table 2.2 is a fair evaluation. Increasing H_{def} would secure some borderline images that break due to random effects while increasing N and H_{adv} would lead to stronger attacks. The evaluation in Table 2.2 pushes the limits of what is computationally feasible with widely available resources. Our defense report is an accurate approximation of the defense of our ideal classifier $F(x)$ in (2.4) against the BPDA+EOT attack (3.3) although we acknowledge that more computation would yield a more accurate estimate.

SVHN and CIFAR-100. The attack and defense parameters for our method are identical to those used in the CIFAR-10 experiments. We compare our results with standard AT. Overall, our defense performs well for datasets that are both simpler and more complex than CIFAR-10. In future work, further stabilization of image appearance after Langevin sampling could yield significant benefits for settings where precise details need to be preserved for accurate classification. The AT results for CIFAR-100 are from [BGH19] and the results for SVHN are from our implementation.

Table 2.3: Defense vs. whitebox attacks with l_∞ perturbation $\varepsilon = 8/255$ for SVHN and CIFAR-100.

	SVHN		CIFAR-100	
	Nat.	Adv.	Nat.	Adv.
Ours	0.9223	0.6755	0.5166	0.2610
[MMS18]	0.8957	0.5039	0.5958	0.2547

2.7 Conclusion

This work demonstrates that Langevin sampling with a convergent EBM is an effective defense for naturally-trained classifiers. Our defense is founded on an improvement to EBM training that enables efficient learning of stable long-run sampling for complex datasets. We evaluate our defense using non-adaptive and adaptive whitebox attacks for the CIFAR-10, CIFAR-100, and SVHN datasets. Our defense is competitive with adversarial training. Securing naturally-trained classifiers with post-training defense is a long-standing open problem that this work resolves.

In the next section we explore how we can scale up these models to defend attacks against ImageNet [DDS09] and what modifications we must make to the training and initialization mechanisms to achieve good defense.

Appendix

2.8 Attack Algorithm

Algorithm 1 BPDA+EOT adaptive attack to evaluate EOT defense (2.4)

```
Require: Natural images $\{x_m^+\}_{m=1}^M$, EBM $U(x)$, classifier $f(x)$, Langevin noise $\tau = 0.01$,
  Langevin updates $K = 1500$, number of attacks $N = 50$, attack step size $\alpha = \frac{2}{255}$, maximum
  perturbation size $\varepsilon = \frac{8}{255}$, EOT attack samples $H_{\text{adv}} = 15$, EOT defense samples $H_{\text{def}} = 150$
Ensure: Defense record $\{d_m\}_{m=1}^M$ for each image.
  for m=1:M do
    Calculate large-sample predicted label of the natural image $\hat{c}_{H_{\text{def}}}(x_m^+)$ with (4.1).
    if $\hat{c}_{H_{\text{def}}}(x_m^+) \neq y_m$ then
      Natural image misclassified. $d_m \leftarrow$ False. End loop iteration $m$.
    else
      $d_m \leftarrow$ True.
    end if
    Randomly initialize $X_0$ in the $l_p$ $\varepsilon$-ball centered at $x_m^+$ and project to $[0,1]^D$.
    for n=1:(N+1) do
      Calculate small-sample predicted label $\hat{c}_{H_{\text{adv}}}(X_{n-1})$ with (4.1).
      Calculated attack gradient $\Delta_{\text{BPDA+EOT}}(X_{n-1}, y_m)$ with (3.3).
      if $\hat{c}_{H_{\text{adv}}}(X_{n-1}) \neq y_m$ then
        Calculate large-sample predicted label $\hat{c}_{H_{\text{def}}}(X_{n-1})$ with (4.1).
        if $\hat{c}_{H_{\text{def}}}(X_{n-1}) \neq y_m$ then
          The attack has succeeded. $d_m \leftarrow$ False. End loop iteration $m$.
        end if
      end if
      Use $\Delta_{\text{BPDA+EOT}}(X_{n-1}, y_m)$ with the $l_p$ $\varepsilon$-bounded PGD update (2.8) to obtain $X_n$.
    end for
  end for
```

Algorithm 1 is pseudo-code for the attack and defense framework described in Section 2.5. One notable aspect of the algorithm is the inclusion of an EOT defense phase to verify potentially successful attacks. Images which are identified as broken for the smaller sample used to generate EOT attack gradients are checked again using a much larger EOT defense sample to ensure that the break is due to the adversarial state and not random finite-sample effects. This division is done for purely computational reasons. It is extremely expensive to use 150 EOT attack replicates but much less expensive to use 15 EOT attack replicates as a screening method and to carefully check candidates for breaks when they are identified from time to time using 150 EOT defense replicates. In our experiments we find that the EOT attack is close to its maximum strength after about 15 to 20 replicates are used, while about 150 EOT defense replicates are needed for consistent evaluation of F from (2.4) over several hundred attacks. Ideally, the same number of chains should be used for both EOT attack and defense, in which case the separate verification phase would not be necessary.

2.9 Improved Learning of Convergent EBMs

Algorithm 2 provides pseudo-code for our improvement of the convergent learning method from [NHH20]. This implementation allows efficient learning of convergent EBMs for complex datasets.

2.10 Erasing Adversarial Signals with MCMC Sampling

This section discusses two theoretical perspectives that justify the use of an EBM for purifying adversarial signals: memoryless and chaotic behaviors from sampling dynamics. We emphasize that the discussion applies primarily to long-run behavior of a Langevin image trajectory. Memoryless and chaotic properties do not appear to emerge from short-run sampling. Throughout our experiments, we never observe significant defense benefits from short-run Langevin sampling.

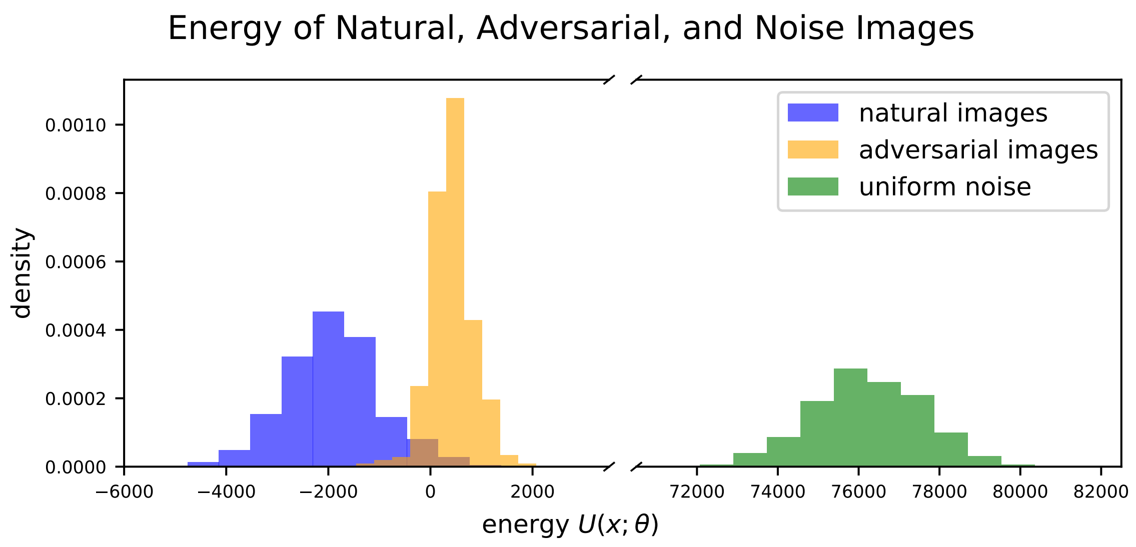


Figure 2.6: Energy $U(x;\theta)$ of natural, adversarial, and noise images.

2.10.1 Memoryless Dynamics

The first justification of EBM defense is that iterative probabilistic updates will move an image from a low-probability adversarial region to a high-probability natural region, as discussed in prior works [SKN18, SMM19, GWJ20]. Comparing the energy of adversarial and natural images shows that adversarial images tend to have marginally higher energy, which is evidence that adversarial images are improbable deviations from the steady-state manifold of the EBM that models the distribution of natural images (see Figure 2.6).

The theoretical foundation for removing adversarial signals with MCMC sampling comes from the well-known steady-state convergence property of Markov chains. The Langevin update (6.3) is designed to converge to the distribution $p(x;\theta)$ learned from unlabeled data after an infinite number of Langevin steps. The steady-state property guarantees that adversarial signals will be erased from the sampled state as long as enough steps are used because the sampled state will have no dependence on the initial state. This property is actually too extreme because full MCMC mixing would completely undermine classification by causing samples to jump between class modes.

The quasi-equilibrium and metastable behaviors of MCMC sampling [BH06] can be as useful as its equilibrium properties. Metastable behavior refers to intramodal mixing that can occur for MCMC trajectories over intermediate time-scales, in contrast to the limiting behavior of full intermodal mixing that occurs for trajectories over arbitrarily large time-scales. Although slow-mixing and high autocorrelation of MCMC chains are often viewed as major shortcomings, these properties enable EBM defense by preserving class-specific features while sampling erases adversarial signals.

Our EBM samples always exhibit some dependence on the initial state for computationally feasible Langevin trajectories. Mixing within a metastable region can greatly reduce the influence of an initial adversarial signal even when full mixing is not occurring. Successful classification of long-run MCMC samples occurs when the metastable regions of the EBM $p(x;\theta)$ are aligned with the class boundaries learned by the classifier network $f(x)$. Our experiments show that this alignment naturally occurs for convergent EBMs and naturally-trained classifiers. No training modification for $f(x)$ is needed to correctly classify long-run EBM samples. Our defense relies on a balance between the memoryless properties of MCMC sampling that erase noise and the metastable properties of MCMC sampling that preserve the initial state.

2.10.2 Chaotic Dynamics

Chaos theory provides another perspective for justifying the erasure of adversarial signals with long-run iterative transformations. Intuitively, a deterministic system is chaotic if an initial infinitesimal perturbation grows exponentially in time so that paths of nearby points become distant as the system evolves (i.e. the butterfly effect). The same concept can be extended to stochastic systems. The SDE

$$\frac{dX}{dt} = V(X) + \eta\xi(t), \tag{2.11}$$

where $\xi(t)$ is Brownian motion and $\eta \geq 0$, that encompasses the Langevin equation is known to exhibit chaotic behavior in many contexts for sufficiently large η [LLB03].

One can determine whether a dynamical system is chaotic or ordered by measuring the maximal Lyapunov exponent λ given by

$$\lambda = \lim_{t\to\infty} \frac{1}{t} \log \frac{|\delta X_\eta(t)|}{|\delta X_\eta(0)|} \tag{2.12}$$

where $\delta X_\eta(t)$ is an infinitesimal perturbation between system state at time t after evolution according to (2.11) from an initial perturbation $\delta X_\eta(0)$. For ergodic dynamics, λ does not depend on the initial perturbation $\delta X_\eta(0)$. Ordered systems have a maximal Lyapunov exponent that is either negative or 0, while chaotic systems have positive Lyapunov exponents. The SDE (2.11) will have a maximal exponent of at least 0 since dynamics in the direction of gradient flow are neither expanding nor contracting. One can therefore detect whether a Langevin equation yields ordered or chaotic dynamics by examining whether its corresponding maximal Lyapunov exponent is 0 or positive.

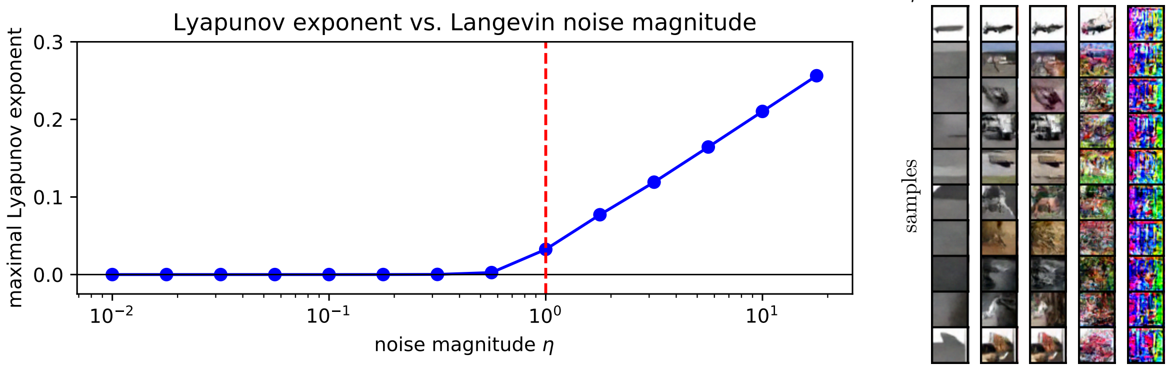


Figure 2.7: *Left:* Maximal Lyapunov exponent for different values of η. The value $\eta = 1$ which corresponds to our training and defense sampling dynamics is just above the transition from the ordered region where the maximal exponent is 0 to the chaotic region that where the maximal exponent is positive. *Right:* Appearance of steady-state samples for different values of η. Oversaturated images appear for low values of η, while noisy images appear for high η. Realistic synthesis is achieved in a small window around $\eta = 1$ where gradient and noise forces are evenly balanced.

We use the classical method of [BGS76] to calculate the maximal Lyapunov exponent of the altered form Langevin transformation (6.3) given by

$$T_\eta(X) = X - \frac{\tau^2}{2} \nabla_X U(X;\theta) + \eta\tau Z_k \tag{2.13}$$

for a variety of noise strengths η. Our results exhibit the predicted transition from noise to chaos. The value $\eta = 1$ which corresponds to our training and defense algorithms is just beyond the transition from the ordered region to the chaotic region. Our defense dynamics occur in a critical interval where ordered gradient forces that promote pattern formation and chaotic noise forces that disrupt pattern formation are balanced. Oversaturation occurs when the gradient dominates and noisy images occur when the noise dominates. The results are shown in Figure 2.7.

The unpredictability of paths under T_η is an effective defense against BPDA because informative attack gradients cannot be generated through chaotic transformation. Changes in adversarial perturbation from one BPDA attack to the next attack are magnified by the Langevin transformation and it becomes difficult to climb the loss landscape $L(F(x), y)$ to create adversarial samples. Other chaotic transformations, either stochastic or deterministic, might be an interesting line of research as a class of defense methods.

2.11 Effect of Number of Langevin Steps and Comparison of Convergent and Non-Convergent Defenses

We examine the effect the number of Langevin steps has on defense accuracy for the CIFAR-10 dataset (see Figure 2.8). Each point displays either the baseline accuracy of our stochastic classifier (orange) or the results of a BPDA+EOT attack (blue) on 1000 test images. The attacks used to make this diagram use a reduced load of $H_{\text{adv}} = 7$ replicates for EOT attacks so these defense accuracies are slightly higher than the full attack results presented in Figure 2.4. Short-run Langevin with $K \leq 100$ steps yields almost no adversarial robustness. Increasing the number of steps gradually increases robustness until the defense saturates at around $K = 2000$. We chose $K = 1500$ steps in our experiments as a good tradeoff between robustness, natural accuracy, and computational cost.

For comparison, we run the same experiment using a non-convergent EBM. The network structure and training are identical to our convergent model, except that we use Adam instead of SGD throughout training. The non-convergent EBM defense cannot achieve high natural

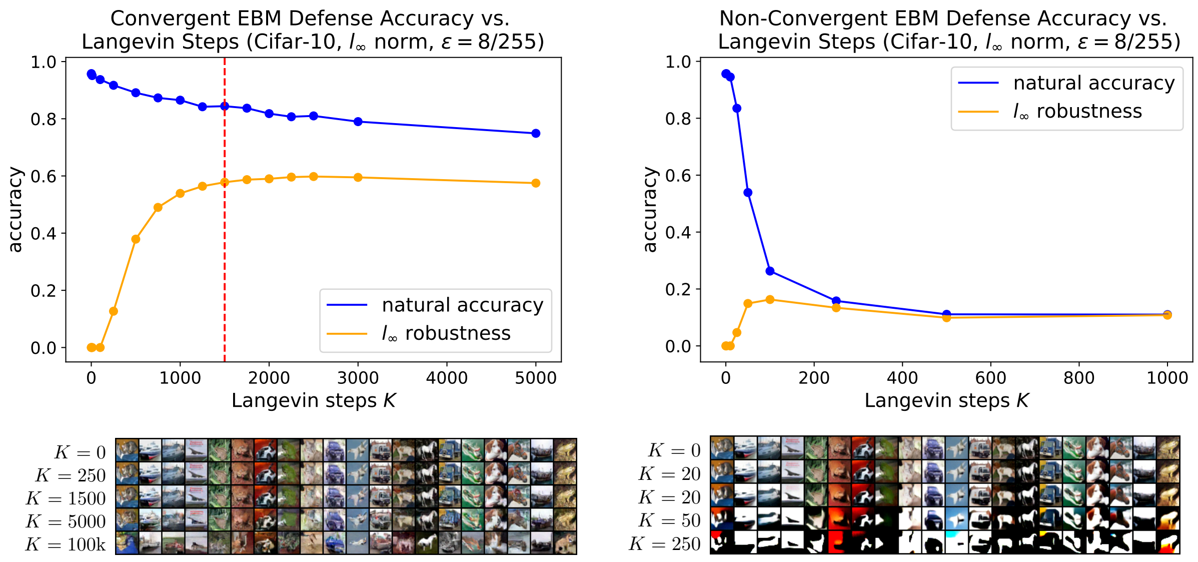


Figure 2.8: Accuracy on natural and adversarial images created from the BPDA+EOT attack (3.3) for EBM defense with different number of Langevin steps, and images sampled from the EBM. *Left:* Defense with a convergent EBM. Using approximately 1500 Langevin steps yields a good balance of natural and robust accuracy. *Right:* Defense with non-convergent EBM. Oversaturated long-run images prevent non-convergent EBM defense from achieving high natural or robust accuracy.

accuracy with long-run sampling because of the oversaturated features that emerge. Without a high natural accuracy, it is impossible to obtain good defense results. Thus convergent EBMs that can produce realistic long-run samples are a key ingredient for the success of our method.

We can further examine the difference between convergent and non-convergent EBM defenses by measuring the FID score of samples initialized from the data (see Figure 2.9). The convergent EBM maintains a reasonably low FID score across many steps so that long-run samples can be accurately classified. The non-convergent EBM experiences a quick increase in FID as oversaturated samples appear. Labels for the oversaturated samples cannot be accurately predicted by a naturally-trained classifier, preventing successful defense.

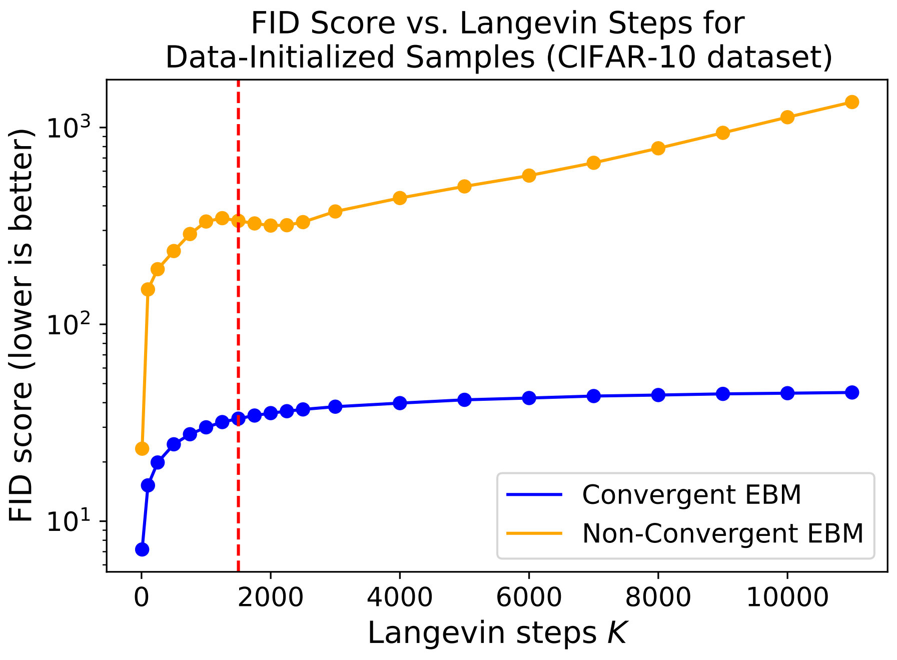


Figure 2.9: FID scores of samples from a convergent and non-convergent EBM that are initialized from training data. The FID of the convergent model remains reasonably low because the steady-state of the convergent model is aligned with the data distribution. The FID of the non-convergent model quickly increases because the oversaturated steady-state of the non-convergent model differs significantly from the data distribution. Maintaining a sampling distribution close to the data distribution is essential for achieving high prediction accuracy of a natural classifier on transformed states.

2.12 Discussion of Defense Runtime

Robustness does not depend on the computational resources of the attacker or defender [ACW18]. Nonetheless, we took efforts to reduce the computational cost of our defense to make it as practical as possible. Our defense requires less computation for classifier training but more computation for evaluation compared to AT due to the use of 1500 Langevin steps as a preprocessing procedure. Running 1500 Langevin steps on a batch of 100 images with our lightweight EBM takes about 13 seconds on a RTX 2070 Super GPU. While this is somewhat costly, it is still possible to evaluate our model on large test sets in a reasonable amount of time. The cost of our defense also poses a computational obstacle to attackers, since the BPDA+EOT attack involves iterative application of the defense dynamics. Thoroughly evaluating our defense on CIFAR-10 took about 2.5 days for the entire testing set with 4 RTX 2070 Super GPUs.

Our EBM is significantly smaller and faster than previous EBMs used for adversarial defense. Our EBM has less than 700K parameters and sampling is about 20x to 30x faster than IGEBM and JEM, as shown in Appendix 2.14. Generating adversarial images using PGD against a large base classifier is also expensive (about 30x slower for a PGD step compared to a Langevin step because our base classifier uses the same Wide ResNet backbone as the JEM model). Therefore 50 PGD steps against the base classifier takes about the same time as 1500 Langevin steps with our EBM, so our model can defend images at approximately the same rate as an attacker who generates adversarial samples against the base classifier. Generating adversarial examples using BPDA+EOT is much slower because defense dynamics must be incorporated. Further reducing the cost of our defense is an important direction for future work.

Our approach is amenable toward massive parallelization. One strategy is to distribute GPU batches temporally and run our algorithm in high FPS with a time delay. Another strategy is to distribute the workload of computing $\hat{F}_H$ across parallel GPU batches.

2.13 A Note on Modified Classifier Training for Preprocessing Defenses

Many works that report significant robustness via defensive transformations ([CRK19, RSF19, YZK19] and others) also modify classifier learning by training with transformed images rather than natural images. Prior defensive transformations that are strong enough to remove adversarial signals have the side effect of greatly reducing the accuracy of a naturally-trained classifier. Therefore, signals introduced by these defensive transformations (high Gaussian noise, ensemble transformations, ME reconstruction, etc.) can be also considered "adversarial" signals (albeit perceptible ones) because they heavily degrade natural classifier accuracy much like the attack signals. From this perspective, modifying training to classify transformed images is a direct analog of AT where the classifier is trained to be robust to "adversarial" signals from the defensive transformation itself rather than a PGD attack. Across many adversarial defense works, identifying a defensive transformation that removes attack signals while preserving high accuracy of a natural classifier has universally proven elusive.

Our experiments train the classifier with natural images alone and not with images generated from our defensive transformation (Langevin sampling). Our approach represents the first successful defense based purely on transformation and validates an entirely different approach compared to defenses which modify training of the base classifier ([MMS18, CRK19, RSF19, YZK19] etc.). To our knowledge, showing that natural classifiers can be secured with post-training defensive transformation is a contribution that is unique in the literature. Our task independent approach has the potential of securing images for many applications using a single defense model, while AT and relatives must learn a robust model for each application.

2.14 Discussion of IGEBM and JEM Defenses

We hypothesize that the non-convergent behavior of the IGEBM [DM19] and JEM [GWJ20] models limits their use as an EBM defense method. Long-run samples from both models have oversaturated and unrealistic appearance (see Figure 2.10). Non-convergent learning behavior is a consequence of training implementation rather than model formulation. Convergent learning may be a path to robustness using the IGEBM and JEM defense methods.

Both prior works use very large networks to maximize scores on generative modeling metrics. As a result, sampling from these models can be up to 30 times slower than sampling from our lightweight EBM structure from [NHZ19a] (see Figure 2.10). The computational feasibility of our method currently relies on the the small scale of our EBM. Given the effectiveness of the weaker and less expensive PGD attack in Section 2.6.1 and the extreme computational cost of sampling with large EBM models, we do not to apply BPDA+EOT to the IGEBM or JEM defense.

The original evaluations of the IGEBM and JEM model use end-to-end backpropagation

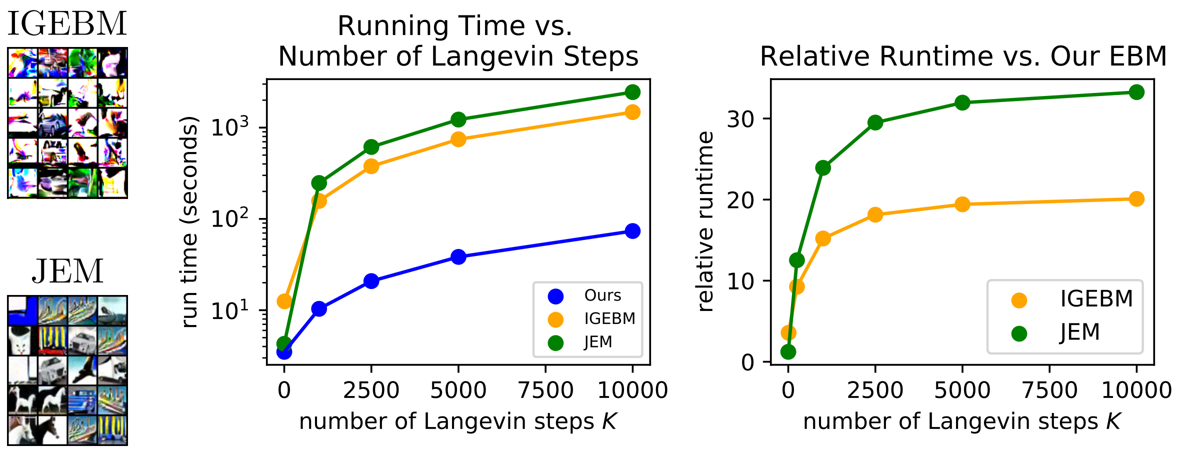


Figure 2.10: *Left:* Approximate steady-state samples of the IGEM and JEM models. Both exhibit oversaturation from non-convergent learning that can interfere with defense capabilities. *Right:* Comparison of running time for Langevin sampling with a batch of 100 images. The small scale and fast sampling of our EBM are important for the computational feasibility of our defense.

through the Langevin dynamics when generating adversarial examples. On the other hand, the relatively weak attack in Section 2.6.1 is as strong or much stronger than the theoretically ideal end-to-end attack. Gradient obfuscation from complex second-order differentiation might hinder the strength of end-to-end PGD when attacking Langevin defenses.

The IGEBM defense overcomes oversaturation by restricting sampling to a ball around the input image, but this likely prevents sampling from being able to manifest its defensive properties. An adversarial signal will be partially preserved by the boundaries of the ball regardless of how many sampling steps are used. Unrestricted sampling, as performed in our work and the JEM defense, is essential for removing adversarial signals.

Algorithm 2 ML with Adam to SGD Switch for Convergent Learning of EBM (6.1)

```
Require: ConvNet potential $U(x;\theta)$, number of training steps $J = 150000$, step to switch
    from SGD to Adam $J_{\text{SGD}} = 50000$, initial weight $\theta_1$, training images $\{x_i^+\}_{i=1}^{N_{\text{data}}}$, data
    perturbation $\tau_{\text{data}} = 0.02$, step size $\tau = 0.01$, Langevin steps $K = 100$, Adam learning rate
    $\gamma_{\text{Adam}} = 0.0001$, SGD learning rate $\gamma_{\text{SGD}} = 0.00005$.
Ensure: Weights $\theta_{J+1}$ for energy $U(x;\theta)$.

  Set optimizer $g \leftarrow \text{Adam}(\gamma_{\text{Adam}})$. Initialize persistent image bank as $N_{\text{data}}$ uniform noise
  images.
  for $j$=1:($J$+1) do
    if $j = J_{\text{SGD}}$ then
      Set optimizer $g \leftarrow \text{SGD}(\gamma_{\text{SGD}})$.
    end if
    1.  Draw batch images $\{x_{(i)}^+\}_{i=1}^m$ from training set, where $(i)$ indicates a randomly
        selected index for sample $i$, and get samples $X_i^+ = x_{(i)} + \tau_{\text{data}} Z_i$, where $Z_i \sim \text{N}(0, I_D)$
        i.i.d.
    2.  Draw initial negative samples $\{Y_i^{(0)}\}_{i=1}^m$ from persistent image bank. Update
        $\{Y_i^{(0)}\}_{i=1}^m$ with the Langevin equation

            $$Y_i^{(k)} = Y_i^{(k-1)} - \frac{\tau^2}{2}\frac{\partial}{\partial y} U(Y_i^{(k-1)};\theta_j) + \tau Z_{i,k},$$

        where $Z_{i,k} \sim \text{N}(0, I_D)$ i.i.d., for $K$ steps to obtain samples $\{X_i^-\}_{i=1}^m = \{Y_i^{(K)}\}_{i=1}^m$.
        Update persistent image bank with images $\{Y_i^{(K)}\}_{i=1}^m$.
    3.  Update the weights by $\theta_{j+1} = \theta_j - g(\Delta\theta_j)$, where $g$ is the optimizer and

            $$\Delta\theta_j = \frac{\partial}{\partial\theta}\left(\frac{1}{n}\sum_{i=1}^n U(X_i^+;\theta_j) - \frac{1}{m}\sum_{i=1}^m U(X_i^-;\theta_j)\right)$$

        is the ML gradient approximation.
  end for
```

CHAPTER 3

Scaling: The Mid-Run sampler

"If I have seen further than others, it is by standing upon the shoulders of giants"
– Sir Isaac Newton

Stochastic security [HMZ21] laid the ground work for an EBM-based defense. It was a great proof of concept but there were a lot of areas we didn't expand on. The network architecture underlying the EBM was hand-crafted and very small with limited capacity. It worked very well for the CIFAR10 32x32 resolution but it would not scale to something like ImageNet [DDS09] at 128x128. We also wanted to find new ways of training the EBM models. What began as just pure experimentation led to the lifecycle paper [HMC22a] which is explained throughout this chapter.

In the lifecycle paper [HMC22a] we create three different models for three different applications. The shortrun model for image synthesis, midrun model for adversarial defense, and longrun model for principled density estimation. In this chapter we will only be covering the mid-run model since its the one tailored toward adversarial defense but at times we touch on the short-run and long-run models to show the evolution of this project. For more information about short-run and long-run please see [HMC22a].

3.1 Introduction

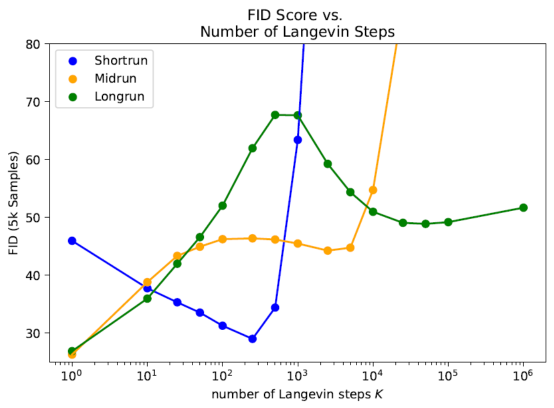
Generative models attempt to model complex signals to learn their underlying distributions. Maximum likelihood learning of an EBM model follows what is known as "Analysis-by-synthesis". For each learning iteration, we sample from the current model to synthesize samples $p(x, \theta)$ and then update the weights / parameters θ of the model according to the difference between the synthesized samples and observed samples $q(x)$ [NHZ19b] [GM06]. After doing this enough times, the synthesized samples and the observed samples will be aligned. The closer the synthesized samples are to the observed samples, the better and more powerful the generative model.

On approach to generative modeling is to posit the existence of a density $q(x)$ of signals x that generated data. For learning Energy-Based Models (EBM) we learn the data distribution $q(x)$ using a model $p(x;\theta)$, where θ are the weights/parameters of the model. This can be seen in other generative models. In EBMs [XLZ16], score-based models [SE19, SE20], normalizing flows [KD18], auto-regressive models [OKV16], Variational Auto-Encoders (VAEs) [KW13] using a joint model $p(x, z; \theta)$. Here we consider modeling $q(x)$ using an EBM density $p(x;\theta)$ for image signals x.

The most common goal in generative modeling is synthesis, creating realistic images from nothing. It is well known that short-run sampling with an EBM is a valid method for image generation but synthesis results for EBMs are still behind GANs [GPM14], diffusion models [HJA20] and score-based models. Within EBM research there is also a focus on learning valid probabilistic densities [NHH20] and combining generative and descriminative learning via the Joint Energy Model [GWJ20]. There also exists the work in Chapter 2: Stochastic Security [HMZ21] where EBMs are used for adversarial defense on image classifiers. In this chapter we are going to focus on defensive applications of EBMs. Later on in section 3.2.4 we compare score based models and EBMs as it pertains to adversarial defense.

Each task is naturally associated with a certain length of MCMC trajectory. Image synthesis is most effective with shortrun trajectories (about 20 to 200 steps) that can rapidly generate new images. Adversarial defense requires midrun trajectories (about 1000 to 2500) that can preserve the class features through metastable behaviors [HMZ21] while sampling removes adversarial signals. Density modeling requires longrun trajectories (50K steps or more) to ensure proper calibration of probability mass for the model steady-state.

It is known that EBMs have a near universal tendency to learn a misaligned steady-state focusing on unrealistic images [NHH20], and there are very few existing solutions to correct



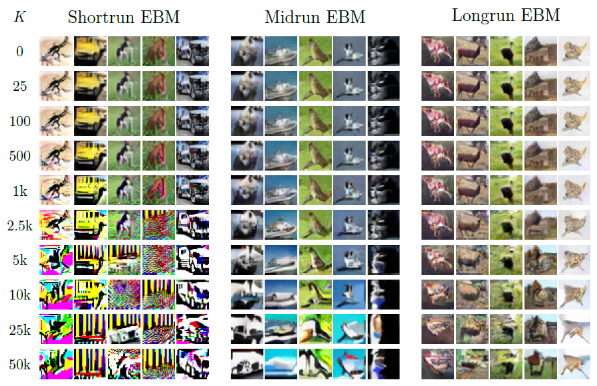


Figure 3.1: Illustration of the sampling trajectories that we study in this work. The shortrun samples are initialized from a generator that is trained in tandem with the EBM because the goal is self-contained synthesis. Midrun and longrun samples are initiated from a high-quality starting image obtained from a pre-trained SNGAN, and we study the ability of the EBMs to preserve the quality of the input image from defense and density estimation points of view. The plots show the FID score [HRU17] of 5,000 samples across Langevin steps. The shortrun samples improve on the generator initialization to achieve high-quality synthesis around 250 steps. The midrun samples achieve reasonably low FID in a critical range of about 2k steps where defense is achieved. The longrun sample maintains reasonable synthesis across the entire trajectory, and much further. The shortrun and midrun samples eventually produce defective results outside of their tuned window of stability.

this. We intuitively refer to the spectrum of trajectories as the life cycle of an MCMC sample, from youth through middle age to maturity. Young samples have the highest quality visual appearance, middle age samples are useful for securing classifiers (defense), and mature samples represent grounded knowledge of the data density (or lack thereof in the widespread case of a misaligned steady-state). Figure 3.1 illustrates the sampling paths that we study in this work.

A valid density estimator should also be a good synthesizer and a good defender, but synthesis and density modeling are two tasks that are at odds with each other. To obtain high quality image synthesis, its better to use shortrun sampling with a non-probabilistic and degenerate data density rather then using long run sampling with a stable density [NHH20]. A shortcoming of great synthesis models is that they lack great defensive capability. The EBM defense from [HMZ21] advocated the use of valid density approximation to stabilize trajectories but that is too strict given the difficulty of valid density approximation. It is much more feasible to learn midrun sampling trajectories for defense on complex datasets such as ImageNet than to apply valid density estimation, even if the longrun samples from the defensive model are not realistic. The Lifecycle [HMC22a] work is the first to explore this conversation. The purpose of that paper is to discuss techniques for building sampling paths from shortrun to longrun, with the goal that the techniques will eventually ennable both high quality synthesis and defense from one model.The difficulty of valid density modeling leads us to restrict our focus to separate time scales of the sampling regime necessary for each task. EBM training naturally accommodates learning at different trajectory lengths.

Strategies for improving EBM learning beyond the standard framework (e.g. [XLG18, DM19, NHH20]) can broadly be divided into methods that focus on the initialization of MCMC samples [XLG18, GLZ18, NHZ19a] and methods that focus on the ML learning objective [YSS20]. Some works explore both [GNK20, DLT20]. Our novel learning methods focus on MCMC initialization, and we retain the standard ML objective and use conventional network architectures. We introduce three new MCMC initialization strategies which are tailored to the three different trajectories lengths we explore. During training we exclusively use shortrun MCMC to ensure computational feasibility. Learning models with midrun and longrun trajectories is accomplished by simulating longer trajectories via well-chosen initialization and optimizer annealing. Our initialization strategies are able to significantly improve the state-of-the-art across the tasks we investigate. We show that persistent initialization with appropriately tuned rejuvenation from in-distribution states can be used to train EBMs with stable trajectories of several thousand MCMC steps. This allows us to extend the method of [HMZ21] to obtain state-of-the-art purification-based defense for CIFAR-10 and to scale the EBM defense to ImageNet.

Our initialization methods will primarily build upon persistent [Tie08, DM19] and cooperative [XLG18] initialization. All of our methods will use a generator network as the source of rejuvenation for persistent states. Our shortrun experiments will learn the generator in tandem with the EBM so the synthesis process is self-contained, while our midrun experiments will use pretrained generators since we will apply sampling paths from in-distribution initial images rather than synthesizing from scratch.

3.1.1 Background on EBM Defense

The most popular method for adversarial defense is adversarial training (AT) [MMS18, WRK20, SNG19] which aims to train a classifier to correctly predict adversarial samples within a small ball around a natural input. Another popular defense method is randomized smoothing [CRK19], which adds Gaussian noise to images to remove adversarial signals before classification. Both of these approaches modify classifier training. Although many methods have been proposed to secure a naturally trained classifier, most have been broken by stronger attacks. A recent method that has been shown to secure a natural classifier is Langevin sampling with an EBM [HMZ21].

The EBM defense uses a classifier trained with labeled natural images and an EBM trained with unlabelled natural images. The two networks are trained independently, which is a key advantage of EBM defense over adversarial training and randomized smoothing. Since the EBM is independent of the classifier, EBM defense has the potential to secure many classifiers across diverse tasks with a single defensive model, while existing methods are typically tailored to a single method. Starting with a naturally classifier trained on natural images $f(x)$, we define its robust counterpart as

$$F(x) = E_{T(x)}[f(T(x))] \tag{3.1}$$

where $T(x)$ is a random variable representing K steps of the Langevin transformation (6.3) initialized from a state x. We cannot evaluate $F(x)$ directly so we approximate it using

$$\hat{F}_H(x) = \frac{1}{H}\sum_{h=1}^{H} f(\hat{x}_h) \quad \text{where} \quad \hat{x}_h \sim T(x) \text{ i.i.d.}, \tag{3.2}$$

where $f(.)$ is a forward pass of our classifier to return logits and where the accuracy of approximation is driven by the number of replicates H. Meaningful evaluation of adversarial defenses must be based on adaptive attack methods which are aware of both $f(x)$ and $T(x)$. For our attack we use the BPDA+EOT formulation from [ACW18, TCB20] to obtain the attack gradient

$$\Delta_{\text{BPDA+EOT}}(x, y) = \frac{1}{H_{\text{adv}}}\sum_{h=1}^{H_{\text{adv}}} \nabla_{\hat{x}_h} L\left(\frac{1}{H_{\text{adv}}}\sum_{h=1}^{H_{\text{adv}}} f(\hat{x}_h), y\right), \quad \hat{x}_h \sim T(x) \text{ i.i.d.} \tag{3.3}$$

which we use in the standard PGD framework to generate adversarial examples. Algorithm 3 gives a sketch of the defense evaluation.

3.2 Midrun Samplers for Adversarial Defense

This section presents a method for learning EBMs that are capable of preserving the appearance of an in-distribution initial state across several thousand MCMC steps. Such models are useful for the purpose of adversarial defense. Our defense framework is based on the approach in [HMZ21], which uses an EBM to defend an independent naturally trained classifier. Appendix 3.1.1 briefly reviews the EBM defense and compares this approach with other defense methods. We then present our proposed method for learning a defensive EBM (4), which is based on persistent initialization using a fixed pretrained generator as a source of rejuvenation. Finally, we apply our defense to achieve state-of-the-art performance for purification-based defense on CIFAR-10 and ImageNet.

Algorithm 3 EBM Defense Algorithm

```
Require: Natural images $\{x_m^+\}_{m=1}^M$, EBM $U(x;\theta)$, classifier $f$, Langevin noise $\eta$, attack replicates $H_{\text{adv}}$, defense replicates $H_{\text{def}}$, $l_\infty$ radius $\varepsilon$, attack step size, $\alpha$, Langevin steps $K$
Ensure: Defense record $\{D_m\}_{m=1}^M$ for each image initialized as ones.
  for $1 \leq i \leq M$ do
    select$(X_i, y_i)$from batch
    Randomly initialize adversary $\hat{X}_0$ inside $L_\infty$ ball around $X_i$
    for $1 \leq j \leq N$ do
      $c_j = \arg\max_\ell [\hat{F}_{H_{\text{adv}}}(\hat{X}_{j-1})]_\ell$
      $\Delta_j = \Delta_{\text{BPDA+EOT}}(\hat{X}_{j-1}, y_i)$
      if $c_j \neq y_i$ then
        $c'_j = \arg\max_\ell [\hat{F}_{H_{\text{def}}}(\hat{X}_{j-1})]_\ell$
        if $c'_j \neq y_i$ then
          $D_i = 0$
        end if
      end if
      $\hat{X}_j = \text{PGD}(\hat{X}_{j-1}, \Delta_j, \varepsilon, \alpha)$
    end for
  end for
```

One limitation of prior EBM defense is the reliance on persistent initialization with no rejuvenation to learn the defensive model. This is done to ensure that defensive sampling trajectories remain stable for arbitrary numbers of steps. However, removing rejuvenation from persistent learning has drawbacks discussed in Section **??**. In particular, it becomes very difficult to learn meaningful EBMs for large and complex datasets such as ImageNet in this framework because the persistent bank cannot represent the diversity of the dataset and the quality of persistent images without rejuvenation quickly degrades. Methods for efficient learning of defensive EBMs at a greater scale are needed to extend the EBM defense to more realistic situations.

We overcome this obstacle by building on the observation that fully stable sampling paths are not required for successful defense. While the defense from [HMZ21] uses EBMs with stable sampling for 100K steps or more, the defense results require less than 2000 steps. A natural question is whether it is possible to learn stable MCMC trajectories for only a predefined midrun range to achieve the defensive benefits without fully stabilizing samples over longrun trajectories. Defining a learning procedure to obtain such models is the goal of this section. Efficiently learning EBMs with stable midrun trajectories allows us to scale up the EBM defense to significantly more challenging domains.

3.2.1 In-Distribution Rejuvenation for Learning a Defensive EBM

The initialization method for our midrun sampler is similar to standard persistent initialization with the adjustment that persistent states are rejuvenated from a frozen pretrained generator rather than noise. We use a generator in our experiments so that the EBM and generator could be used after training to sample from scratch. This choice is not necessary and rejuvenation from data samples, or another efficient in-distribution initialization, is also effective for learning defensive EBMs. We note that MCMC initialization from a trained generator or from data samples is not explored in recent work because the goal of most current EBM learning is image synthesis, which becomes trivial if EBM trajectories are always initialized from high-quality samples. From the perspective of synthesis our learning process is nearly invisible but from the perspective of defense its utility becomes concrete.

Learning an EBM for defense involves tuning the length of the sampling trajectory via the number of shortrun training steps and the rejuvenation rate, and tuning the annealing schedule of the EBM optimizer. Given a desired number of MCMC steps K_{def} for a defensive update and a shortrun trajectory $K \approx 100$, we simply set the rejuvenation rate to $p_{\text{rejuv}} = K/K_{\text{def}}$ to ensure that on average samples will travel K_{def} steps before rejuvenation. While in practice we use $K_{\text{def}} = 2000$ and $p_{\text{rejuv}} = 0.05$, we have found that this method can yield stable paths for at least $K_{\text{def}} = 50\text{K}$ MCMC steps when p_{rejuv} is low.

Initialization alone is insufficient to stabilize MCMC pathways when model weights are changing quickly. Using a low learning rate late in training is a key aspect of stabilizing sampling paths [NHH20, HMZ21]. Intuitively, if the EBM optimizer has a sufficiently low learning rate then MCMC trajectories in the persistent image bank can function as approximate trajectories from the current model, since weights change very little as the persistent states are updated. By annealing in tandem with our initialization, we are effectively using midrun trajectories of length K_{def} initialized from the generator to update the EBM while we are actually using shortrun trajectories of length K from the persistent bank. Annealing is a crucial component for stabilizing both midrun and longrun trajectories. Without annealing, sampling paths are not able to maintain realism for large K_{def}.

3.2.2 Experiments: Defending Natural Classifiers with an EBM

We train our EBMs using the persistent initialization described above in tandem with a pretrained generator on both CIFAR-10 and ImageNet. We use the same SNGAN models as before for our CIFAR-10 experiments, with the exception that the generator is pretrained instead of learned. For our ImageNet experiments, we use the BigGAN [BDS19] discriminator architecture for our EBM modified for input size 224×224 and a pretrained BigGAN Generator. Our naturally trained classifier $f(x)$ is a pretrained WideResNet 28-10 [ZK16] for CIFAR-10 and a pretrained EfficientNetB-7 architecture [TL20] for ImageNet.

We evaluate our models using the attack gradient (3.3) and Algorithm 3. For CIFAR-10

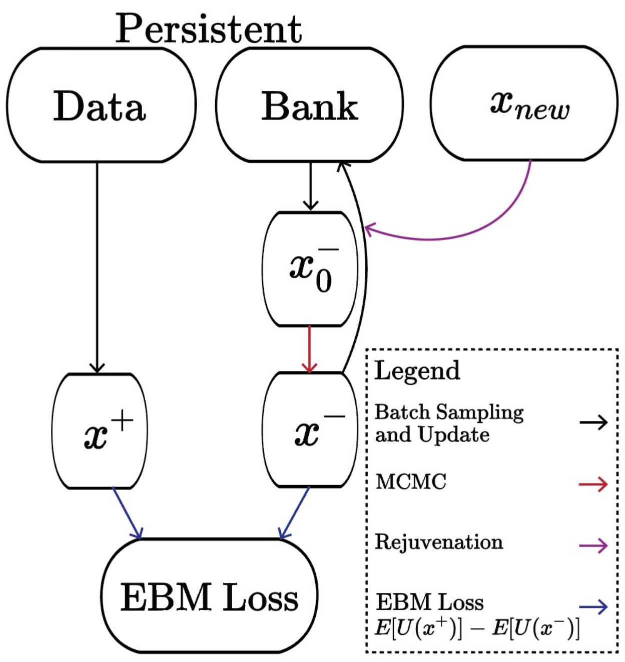


Figure 3.2: Persistent initialization. Positive samples are from data and negative samples are MCMC samples initialized from a batch from the image bank. Some states are randomly rejuvenated when returning to the bank.

we use $K_{\text{def}} = 1500$ Langevin steps with l_∞ adversarial parameters $\varepsilon = \frac{8}{255}$ and $\alpha = \frac{2}{255}$, where ε is the size of the l_∞ ball and α is the gradient step size. For ImageNet we used $K_{\text{def}} = 200$ Langevin steps for defense with l_∞ adversarial parameters $\varepsilon = \frac{2}{255}$ and $\alpha = \frac{1}{255}$. We attack ImageNet for 50 attacks steps across 10K val samples. For CIFAR-10 we perform the same number of attacks across 5k validation samples.

On CIFAR-10, we surpass the robustness of the existing EBM defense using a much more reliable learning framework. The importance of midrun learning is clearly demonstrated by our successful application of EBM defense to ImageNet at the resolution 224×224. The robustness of a naturally trained classifier secured by our EBM is comparable with adversarial training. While the ImageNet results for EBM defense are not yet on par with state-of-the-art adversarially trained models, our experiments are an important proof of concept that the method can be scaled. See Appendix 3.2.3 for diagnostics and further discussion.

3.2.3 EBM Defense Experiment Details and Diagnostics

Table 3.1: Defense vs. whitebox attacks with l_∞ perturbation $\varepsilon = 8/255$ for CIFAR-10.

Defense	$f(x)$ Train Ims.	$T(x)$ Method	Attack	Nat.	Adv.
Ours	Natural	Langevin	BPDA+EOT	0.866	**0.567**
[HMZ21]	Natural	Langevin	BPDA+EOT	0.8412	0.5490
[SKN18]	Natural	Gibbs Update	BPDA	0.95	0.09
[SMM19]	Natural	Langevin	PGD	–	0.0048
[YZK19]	Transformed	Mask + Recon.	BPDA+EOT	0.94	0.15
[CRS19]	Adversarial	–	PGD	0.897	0.625
[ZYJ19]	Adversarial	–	PGD	0.849	0.5643
[SNG19]	Adversarial	–	PGD	0.859	0.4633
[MMS18]	Adversarial	–	PGD	0.873	0.458

To verify the integrity of our results we ran an attack with heavily increased resources for ImageNet compared to our standard evaluation. While using these resources for all attacks is infeasible in practice, we want to ensure our defense maintains robustness as attacker resources increase. As shown in Table 3.3, our benchmark accuracy remains consistent when we increase the number of attack steps (from 50 to 200) and EOT attack replicates (H_{adv}).

We demonstrate results over varying numbers of langevin steps K. We can see in Fig 3.3 that our sampling trajectory for defending imagenet at $K = 200$ is reasonable to achieve high natural image classification as well as robustness.

3.2.4 Comparison with Diffusion Models for Adversarial Defense

The score-based model from [SE20] and its annealed Langevin dynamics process has recently been used for purifying adversarial signals [LO21, YHL21]. One approach is add noise and the using the score model to denoise [LO21]. The robustness of this method is upper-bounded by standard randomized smoothing [CRK19]. Another approach is to initiate the Langevin process of the score model from a natural image as done in the EBM defense. A score model can be used in a langevin process $T(x)$ that is a direct analogue to the EBM langevin process

Table 3.2: Defense vs. l_∞ whitebox attacks for ImageNet.

Defense	$f(x)$ Train Ims.	ε	Nat.	Adv.
Ours	Natural	$\frac{2}{255}$	0.684	0.418
[WRK20]	Adversarial	$\frac{2}{255}$	0.609	0.4339
[SNG19]	Adversarial	$\frac{2}{255}$	0.644	0.4339
[QMG19]	Adversarial	$\frac{4}{255}$	0.822	0.427
[XTG21]	Adversarial	$\frac{4}{255}$	0.822	**0.586**

Table 3.3: Defense for l_∞ against high-power whitebox attacks on ImageNet.

Dataset	Nat	Adv	H_{adv}	H_{def}	samples
ImageNet	0.683	0.38	32	64	1600

in (6.3) and (3.3). Given a score model $S_\theta(x)$ for a low noise value σ, one can define the Langevin equation

$$X_t = X_{t-1}\frac{\eta^2}{2}S_\theta(X_{t-1}) + \eta Z_t$$

where $S_\theta(x) \approx \nabla_x \log q(x)$ since σ is low.

We experimented with this process as a defense mechanism by selecting the smallest trained noise value $\sigma = 0.01$ and using the Langevin process as a method to purify adversaries. We evaluated this method using our BPDA+EOT attack framework over different numbers of Langevin steps during purification. In contrast to reports from [YHL21], we were unable to obtain any significant defense using a pretrained score model when initializing sampling directly from adversarial or natural images. In Figure 3.4, one can see that the score-based model drives natural images toward saturation quickly, leading to a sharp decrease in natural classification that undermines the possibility of robustness from sampling. The misaligned steady-state of the score-based models prevents it from being an a defensive transformation because natural accuracy drops before robustness kicks in. Since the score-based model does not perform sampling during training, and one cannot adjust the stability of its sampling

Algorithm 4 Training a midrun sampler for EBM defense

```
Require: Natural images $\{x_m^+\}_{m=1}^M$, EBM $U(x;\theta)$, frozen pre-trained generator $g(z)$, Langevin noise
    $\eta$, Langevin steps $K$, EBM optimizer $h_U$, initial weights $\theta_0$, rejuvenation probability $p$, number of
    training iterations $T$.
Ensure: Weights $\theta_T$ for defensive EBM.
  Initialize image bank $\{X_i^-\}_{i=1}^N$ from generator using $X_i^- = g(Z)$
  for $1 \leq t \leq T$ do
    Select batch $\{\tilde{X}_b^+\}_{b=1}^B$ from data samples $\{x_m^+\}_{m=1}^M$.
    Get negative sample batch $\{\tilde{X}_{b,0}^-\}_{b=1}^B$ from $\{X_i^-\}_{i=1}^N$.
    Update $\{\tilde{X}_{b,0}^-\}_{b=1}^B$ with $K$ Langevin steps (6.3) to obtain negative samples $\{\tilde{X}_b^-\}_{b=1}^B$.
    Get learning gradient $\Delta_U^{(t)}$ using (6.2) with samples $\{\tilde{X}_b^+\}_{b=1}^B$ and $\{\tilde{X}_b^-\}_{b=1}^B$.
    Update $\theta_t$ using gradient $\Delta_U^{(t)}$ and optimizer $h_U$.
    Rejuvenate each $\tilde{X}_b$ from a pretrained generator $g$ with probability $p$.
    Return $\{\tilde{X}_b^-\}_{b=1}^B$ to $\{X_i^-\}_{i=1}^N$ by overwriting previous states.
  end for
```

process as we do in this work. While it is not immediately clear how to overcome this problem, we believe that defense with a score model is possible and we hope that our observation lead to efforts to stabilize the sampling paths of score models as we do for EBMs in this work.

3.3 Related Work

Energy-Based Models. Early forms of EBMs include the exponential family distribution, the FRAME model [ZWM98] and Restricted Boltzmann Machines [Hin02]. Recent work has introduced the EBM with a ConvNet potential [XLZ16, DM19]. This dramatically increased the learning capacity of the model which led to many follow-up works on image synthesis [GLZ18, LXF18, NHZ19a], adversarial robustness [HMZ21], and joint learning of discriminative and generative models [GWJ20]. Several works investigate training and EBM in tandem with an auxilary model. [KB16] train an EBM and generator and tandem without MCMC by using samples from the generator as direct approximations of the EBM density and training the generator using a variational objective. A similar approach is explored

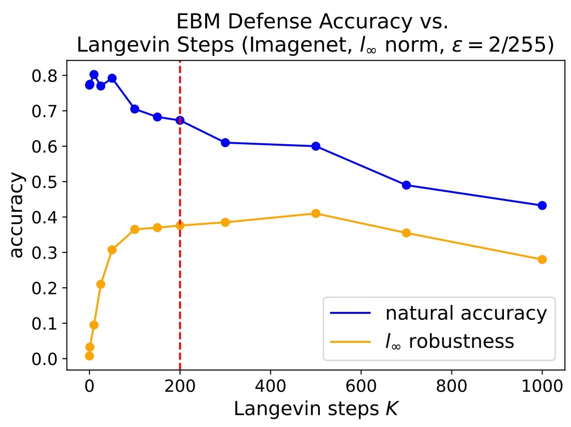


Figure 3.3: Accuracy over varying numbers of langevin steps K for ImageNet experiments.

by [GKH21]. Cooperative learning [XLG18] trains the EBM and generator by using the generator to initialize samples needed to train the EBM and uses reconstruction loss between generator and EBM samples to learn the generator. [GNK20] learn an EBM using Noise Contrastive Estimation with an auxiliary flow model. [XKK21] use a pretrained VAE to facilitate EBM learning. Our work builds on cooperative learning by identifying and resolving symmetry breaking problems in early training, leading to state-of-the-art EBM synthesis for unconditional ImageNet. Despite the formulation of the EBM as an unnormalized density, it has been shown that most EBMs have strong misaligned steady-state distributions [NHH20]. Our work introduces new methods to learn a model with correct steady-state alignment.

Adversarial Robustness. Adversarial Training (AT) [MMS18], which trains a classifier using PGD-generated adversaries, is the most popular and studied adversarial defense. Many variations and improvements have been introduced, including optimizing the training loop by recycling gradients of past adversaries [SNG19], combining single step FGSM with random initialization to achieve similar robustness [WRK20], learning with auxiliary unlabeled data [CRS19], local linearization [QMG19], and the use of smooth activation functions [XTG21]. An alternative approach to adversarial training involves the use of preprocessing transformations. Randomized smoothing [CRK19] and related methods [SSY20] add noise to the input signal to remove adversarial signals. Many other preprocessing defenses have been proposed [GRC18, SKN18, YZK19], but nearly all of these methods can be broken by

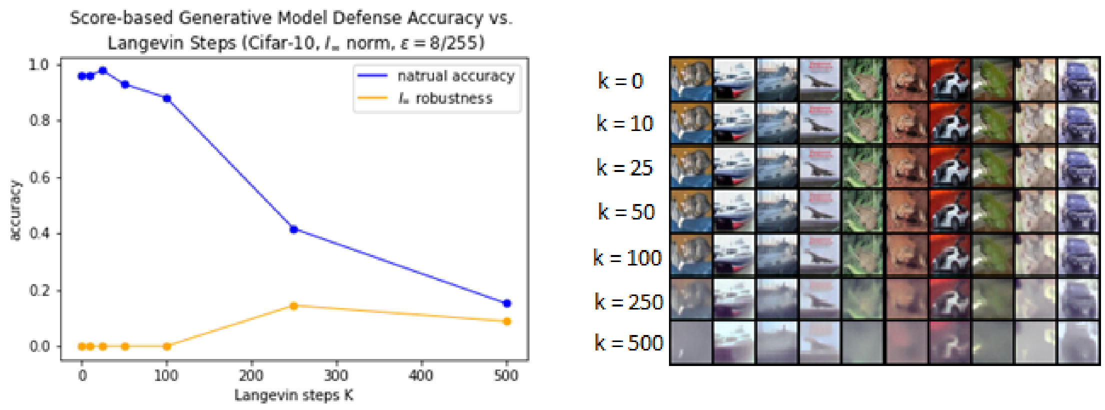


Figure 3.4: Score-based Langevin experiment on the CIFAR-10 dataset. Left: Accuracy of natural and adversarial images resulting from a BPDA+EOT defense using a score-based model with an annealed langevin purification method for 125 samples over varying steps. Right: Samples received from this annealed langevin diffusion process over the same sampling lengths.

adaptive attacks that are aware of the preprocessing method [ACW18]. A notable exception is the EBM defense [HMZ21], which uses midrun MCMC trajectories to purify images. We ease the restriction of learning EBMs with stability for arbitrary MCMC runs in the EBM defense by introducing a midrun sampler that enables faster learning of defensive EBMs and allows the EBM defense to scale to more complex datasets.

3.4 Conclusion and Future Work

We have described a unique MCMC initialization procedure for training a mid-run sampler capable of adversarial defense. For training methods related to short-run models for image synthesis and long-run samplers for density estimation please see [HMC22a]. We have demonstrated the flexibility of these mechanisms by using similar architectures, data, and training platforms to create different EBMs for different applications. We hope that future research incorporates these new training initialization schemes to improve their generative models for a wide variety of tasks.

In the next chapters: Extension and Fixer, I will show how we took this framework and extended it towards multiple tasks and datasets. As well as provide some enhancements to make the defense even stronger.

CHAPTER 4

Extension

4.1 The Universal Defender: One Classifier To Rule Them All

4.2 Intro

In the introduction we spoke about "transferability" between domains. Can the EBM transfer knowledge between domains? If an EBM is trained on a dataset q with classes k, can it defend against attacks on a dataset with classes p if $k \cap p$? We can consider the same question for extending tasks. Can we extend tasks from classification to another task?. The EBM knows nothing about the task **classification**, yet it is able to secure pretrained image **classifiers**. If that's the case, then why can't the EBM defend against other tasks, such as "Image Segmentation"?

4.3 Dataset Extension

If we train our EBM $U(x; \theta)$ on ImageNet [DDS09] which contains 1000 classes, can it defend attacks using a dataset containing images from one of those classes? In theory, if the EBM is able to defend a particular class that defense should extend to multiple datasets containing the same class.

As an experiment we trained a classifier on the flowers dataset [NZ08] which contains 10k images of flowers with 104 classes of different flowers. We used the same mid-run sampler from [HMC22a] to defend against attacks on the flowers dataset, even though the classifier we are defending was not trained on this dataset.

Dataset	$F(X\_nat)$	$F(X\_adv)$	Samples
Flowers	0.821	0.519	4520
ImageNet	0.684	0.418	10000

Table 4.1: Results for EBM defense using the mid-run sampler EBM trained on ImageNet[DDS09] from [HMC22b] to defend against a flowers 10k dataset BPDA attack.

A 52% robustness on a dataset never seen by the EBM is significant. This shows that the EBM can be trained once and extended to defend datasets within its learned domain.

4.4 Task Extension

4.4.1 Image Segmentation: What is it?

Image Segmentation is the process of partitioning a digital image into various subgroups called segments. In the canonical task of image segmentation each cluster or group of pixel segments belongs to a particular class. The standard method of training an image segmentation network is to use an architecture known as a U-Net. [RFB15] and use a per-pixel cross entropy loss to force the network to learn the correct class contours.

4.4.2 Extending the attack toward segmentation

We setup an experiment where we train a U-Net DNN for semantic segmentation on the Pets dataset [PVZ12]. We then defend against this task using the mid-run sampling EBM defender which was trained on ImageNet [DDS09]. In this case we are extending both the task and the dataset.

In the original formulation of the EOT attack and defense from [HMZ21] we determined a correct or false classification based on cross entropy over the per-image label class:

$$\hat{c}_H(x) = \arg\max_j \hat{F}_H(x)_j \tag{4.1}$$

However, semantic segmentation is a different problem. To extend from per-image classification toward semantic segmentation we introduce the DICE coefficient [Dic45] and follow a similar attack pattern to [DPA21] except that we use the BPDA attack [ACW18] instead of PGD [MMS18]. The dice coefficient was originally developed as a statistic to measure the similarity between two samples, similar to F1 score.

$$\frac{2\text{TP}}{2\text{TP} + \text{FP} * \text{FN} + \epsilon} \tag{4.2}$$

where TP represents true positives, false positives, false negatives and ϵ is an offset value of $1e - 9$ to prevent division by 0. To compute a correct result we reformulate eq 4.1 to:

$$\hat{c}_H(x) = \frac{\text{Dice}(\hat{x})}{\text{Dice}(x)} < 0.5 \tag{4.3}$$

where $\hat{X}$ represents the purified sample $\hat{x} \sim T(x)$ that has been attacked and purified using the EBM and x is the original observed sample from the data distribution $x \sim q$. Thus if the Dice score drops by half of the original value we determine the segmentation to be incorrect. We proceed to attack the network using BPDA + EOT [ACW18] as in Stochastic Security [HMZ21] for fair evaluation.

4.4.3 Results

Dataset	Defense	$F(X\_adv)$	Samples
Pets	No	0.43	800
Pets	Yes	0.60	800

Table 4.2: Adversarial Defense w.r.t the BPDA attack on the image segmentation task using the pets dataset [PVZ12].

In table 4.2 we can clearly see that when using the ImageNet EBM trained with an SNGAN [MKK18] architecture from the lifecycle work [HMC22a] we gain 17% in robustness. Thus, task extension works.

4.5 Conclusion and Next Steps

Now that we have shown we can extend both the task and dataset domain of our EBM defense, what other improvements can we make to the core architecture of the defense framework so that it is even more robust? In the next section we dive in to the "Fixer" architecture.

CHAPTER 5

The Fixer

5.1 The Fixer

There is noise left over from EBM-based purification. It isn't perfect. When we use the metastable image basins to remove adversarial signals and replace them with features from the same class some noise is left over (because we use noise when sampling). Sometimes, our classifier misclassifies samples not from the adversarial signal but from the impurities caused by Langevin sampling. I drew inspiration from [SSY20] to essentially close the gap between purified adversarial samples and the natural image distribution. To do this, we extend the Stochastic Security formulation [HMZ21] to include G, an SNGAN [MKK18]-based Generator-like model called "The Fixer". We "Fix" the impurities caused by EBM sampling.

The Fixer closes the gap $KL(G(T(x;\theta),\phi)||q))$ where $T(x)$ represents a Langevin process using EBM $U(x;\theta)$ with weights θ and G is the Fixer with weights ϕ. When training, we simply purify a natural image, and use G to close the gap. The main goal of the fixer is to improve our upper bound which how well we classify natural images that have been purified $F(x_{\text{nat}})$.

5.2 Extending Stochastic Security

A glance at figure 5.1 makes it obvious that adversarial purification with the Fixer is a clear extension of Stochastic Security. Namely, we reformulate the approximation of the stochastic classifier to encompass the fixer component.

$$F(x) = E_{G(T(x))}[f(G(T(x)))]. \tag{5.1}$$

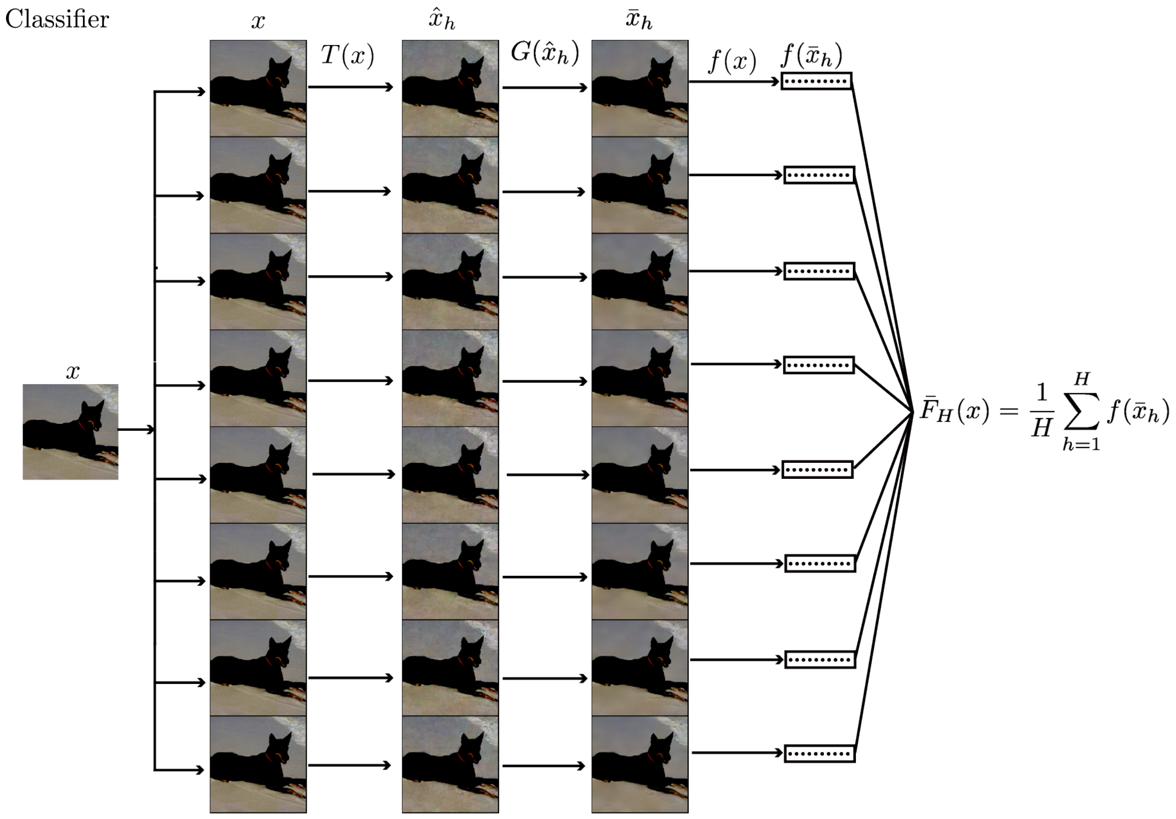


Figure 5.1: Illustration of the fixer method. Here we can see a direct extension of the method presented in Chapter 2: Stochastic Security with the addition of the Fixer network $G(.)$ after purification. The samples shown here are downscaled ImageNet samples whereas figure 5.3 shows these same images at a larger scale.

This is fair since we still evaluate the attack as the Expectation Over Transformation (EOT) as done in [HMZ21]. The addition of $G(.)$ simply adds another component to the existing purifier. The approximate stochastic classifier becomes:

$$\bar{F}_H(x) = \frac{1}{H}\sum_{h=1}^{H} f(\bar{x}_h) \quad \text{where,} \quad \bar{x}_h \sim G(T(x)) \text{ i.i.d.}, \tag{5.2}$$

In essence, the purification method for secure the classifier now encapsulates the fixer, so any attack that targets purification-based defenses will naturally target those methods with the fixer as well.

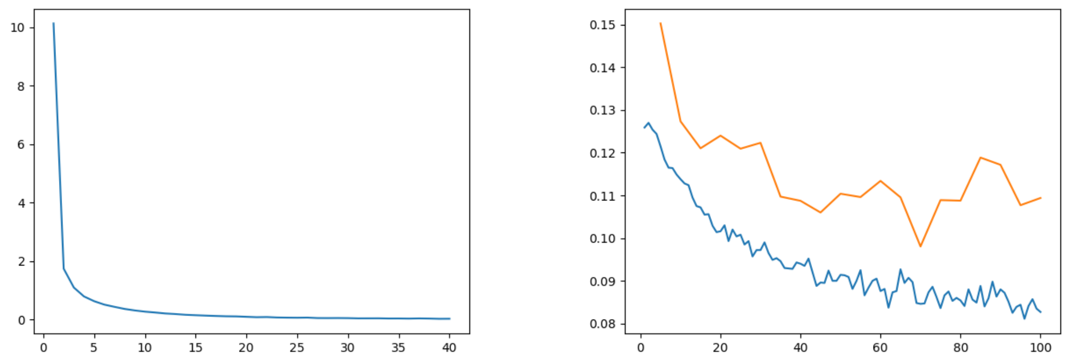


Figure 5.2: Left: Loss curve using MSE loss as in equation 5.3 to pretrain the fixer for 40 epochs. We can see that after 40 epochs to fixer loss saturates and it is ready for the next phase. Right: Loss curve using Perception loss for 100 epochs as in equation 5.4. The blue curve represents loss w.r.t training data while the orange curve is loss w.r.t validation data. We can see that the loss saturates around 100 epochs and we use early stopping so that it's not overfit.

5.2.1 Training The Fixer

To train the Fixer we use two loss mechanisms. To train this model we have two schedules. First, we pre-train the generator for 40 epochs using MSE loss (fig 5.2 left)

$$L_{\text{mse}} = \|G(x;\phi) - x\|_2^2, \text{where } \hat{x} \sim T(x) \tag{5.3}$$

Where x is a natural image and ϕ are the weights of the Fixer. The goal is to make the Generator the identity $x = G(x)$ to warm up the weights. Then in the next schedule we use the perceptual loss [JAF16] on purified and natural image samples for 100 epochs as in figure 5.2 right. This extension is natural. Perceptual loss was created for style transfer, and I am essentially trying to make the EBM un-style the impurities caused by purification.

Instead of attempting to have to pixels from $G(\hat{x};\theta)$ match the exact pixels of x, we want them to have similar feature map representations within a neural network δ [JAF16]. Consider a loss network δ which for practical purposes is a VGG network [SZ14]. If this network is a function then $\delta_j(x)$ is the feature map activations of the jth layer of this network. Since VGG is a Convolutional Neural Network, then the feature map would have shape (C_j, H_j, W_j) which denotes channels, height, width respectively. We consider the feature reconstruction loss of this network to be the perceptual loss for all practical purposes:

$$l_{\text{feat}}^{\delta,j}(G(\hat{x};\phi), x) = \frac{1}{C_j H_j W_j} \|\delta_j(G(\hat{x};\phi)) - \delta_j(x)\|_2^2, \text{where } \hat{x} \sim T(x) \tag{5.4}$$

This pushes the output of $G(\hat{x})$ to be similar to x. In practice we use VGG-16 as the loss network. Note that the EBM weights are frozen throughout the duration of Fixer training. To integrate this model into our pipeline we swap $T(x)$ in Stochastic Security for $G(T(x);\theta)$.

The Fixer problem can actually be framed as a Super Resolution problem. [LTH16]. Where SRGAN [LTH16] tries to remove noise caused by upsampling, we want to remove noise caused by Langevin sampling. The only difference is that we don't have to upsample the network, we can just maintain the same dimensionality.

With this in mind, I decided to pretrain the Fixer on data q (eq 5.3) just until the loss started to saturate as can be seen in fig 5.2 left. Then I swapped in perceptual loss to take over as in eq 5.4 and fig 5.2 right. I got this idea from a FastAi article [Ant19].

5.2.2 Attacking the fixer

By direct evaluation of the BPDA attack [ACW18] on ImageNet we measure the robustness of our classifiers when using the fixer.

Table 5.1: Defense vs. l_∞ whitebox attacks for ImageNet.

Defense	$f(x)$ Train Ims.	ε	Nat.	Adv.
Fixer	Natural	$\frac{2}{255}$	0.742	0.498
[HMC22b]	Natural	$\frac{2}{255}$	0.684	0.418
[WRK20]	Adversarial	$\frac{2}{255}$	0.609	0.4339
[SNG19]	Adversarial	$\frac{2}{255}$	0.644	0.4339
[QMG19]	Adversarial	$\frac{4}{255}$	0.822	0.427
[XTG21]	Adversarial	$\frac{4}{255}$	0.822	**0.586**

In table 5.1 we compare results from the mid-run sampling in Lifecycle [HMC22b] to the same model with an appended Fixer architecture. We can see that the Fixer improves our upper bound $F(x_{\text{nat}})$ which leads to a $\sim 9\%$ increase in robustness. Any jump in natural image accuracy will also scale toward more adversarial robustness.

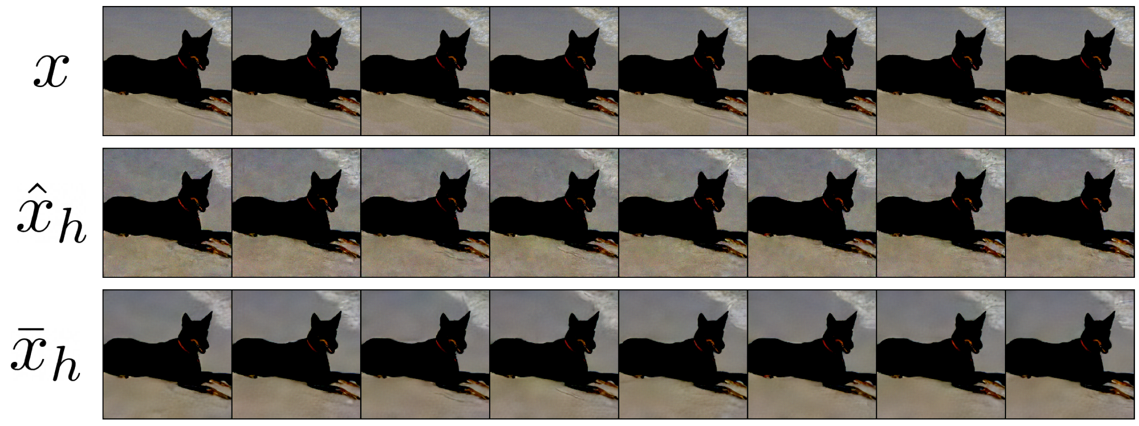


Figure 5.3: Example showing the adversarial images replicates x_h, their purified counter parts $\hat{x}_h$ and after the fixer $\bar{x}_h$. In this diagram we can clearly see the fixer doing it's work. Notice how $\hat{x}_h$ has some coarse features that are smoothed away when we see their "fixed" counterparts in $\bar{x}_h$

CHAPTER 6

The Hat EBM

6.1 Motivation

The idea for the Hat-EBM stems from the Lifecycle [HMC22a] paper. The Hat-EBM shares the principle of generator-based initialization from [XLG18] [XZF21] that was further enhanced by the shortrun method in [HMC22a]. Furtheremore, we also brough the improvements to cooperative learning into the Hat EBM which includes the idea of using historical generator updates during learning to improve EBM memory and stability. This is further described in section 6.4.5 and figure 6.4. Moreover, we also carried over the idea of paired-banks during generator updates, we kept the bank of latent samples and added another bank for residual images. This work was a collaboration with Mitch Hill, Erik Nijkamp, and Bo Pang [MZ23].

The core idea for the HAT-ebm is to use any generator as the foundation of an Energy-Based Model (EBM). We posit that observed images are the sum of unobserved latent variables passed through the generator $G(.)$ and a residual random variable y that fills the gap between the generator output and the image manifold which is depicted in figure 6.1. We define the hat EBM as an EBM that includes the generator as part of its forward pass. This model can be trained without inferring the latent variables of the observed data or dealing with the Jacobian determinant of the generator. This allows us to enable explicit probabilistic modeling of the output distribution of any generator-based network. Ours experiments show strong performance of the proposed method on (1) unconditional image sampling on ImageNet at $128x128$ resolution. (2) Refinement of existing generators and (3) Retrofitting non-probalistic generators. Throughout this work we can see a lot of familiarity with the work of [HMC22a]. Specifically the way the memory banks are shared.

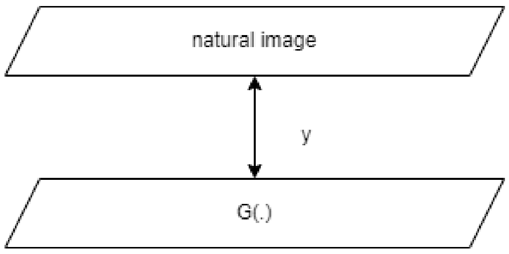


Figure 6.1: Visualization of the residual y closing the gap between the manifold of the generator $G(.)$ and the natural image distribution. The goal of the HAT-EBM is to move synthesized samples into the natural image distribution because that's where realism occurs.

6.2 Introduction

One challenge of generative models is obtaining an explicit representation of the probability distribution defined by the output of the network after transforming the latent space. For Generative Adversarial Networks (GANs) [GPM14, RMC15] and Variational Autoencoders (VAEs) [KW13, RMW14] where the latent states that correspond to realistic images follow a naive distribution (e.g. isotropig Gaussian), it's difficult to obtain image space probabilities since we must calculate the log determinant of the Jacobian of the generator transform for density change-of-variables. For other models such as the deterministic autoencoder, the latent space that corresponds to realistic images is much smaller than the image space in terms of dimensionality which makes it even more difficult to obtain probabilistic samples.

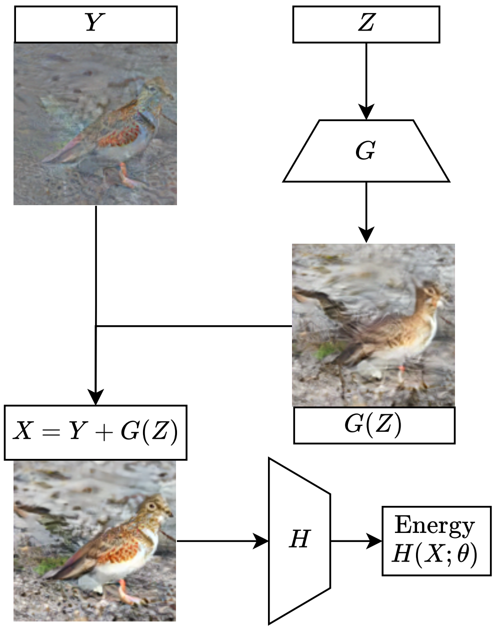
The *Hat EBM* [MZ23] proposed a method for using a generator network as the foundation of an Energy-Based Model (EBM). The generator network is concatenated with a *hat network* that takes in an image as input and outputs a scalar. A residual image is added to the generated output before it is then fed into the hat network. This residual image spans the gap between the generator output and the image manifold as shown in fig 6.1. The complete function, including the generator, and the residual image, and the hat network is called the *Hat EBM.* Therefore, we define an EBM which encapsulated a generator latent space as part of its internal MCMC sampling process. In figure 6.2 we show a diagram of the Hat EBM and selected Hat EBM samples for unconditional ImageNet. There already exist some methods for converting generators to EBMs but those only apply to specific generator models like GAN [CZS20] or VAE [XKK21]. The HAT-EBM is able to convert any generator-based model into an EBM. Moreover, our learning method doesn't require any inference of the latent states of the observed data which is required in related works such as [KW13, XLG18, HNF19, PHN20].

The Hat EBM is a tool to be used in conjunction with pretrained generators. It can refine samples from an existing probabilistic generator (GANs) and synthesize samples using a non-probabilistic generator (Autoencoders). Moreover, we propose a self-contained learning strategy that extends cooperative learning [XLG18] of EBM and generator networks to achieve great synthesis. The main contributions are:

- We introduce a method for defining a Hat EBM that incorporates a generator network as part of its forward pass. This EBM uses the generator latent space as part of Langevin sampling.
- We show that our method can refine samples from pretrained GAN generators and sample from the latent space of deterministic autoencoders which were originally incompatible with sampling.
- We propose a self-contained Hat EBM learning method that trains both a generator and energy network from scratch. This enables us to achieve an FID score of 29.2 on unconditional ImageNet at resolution 128×128, demonstrating that EBMs are competitive with state-of-the-art generative models on complex and high resolution datasets.

6.3 Related Work

EBM. An EBM defines an unnormalized density or a Gibbs distribution. The prototypes include exponential family distributions, Boltzmann machines [AHS85, SH09], and the FRAME (Filters, Random field, And Maximum Entropy) model [ZWM98]. Recent work has introduced the EBM with a ConvNet potential [XZW17, XZG18]. This dramatically increases the model capacity and demonstrates strong image synthesis performance [NHZ19a, DM19] and adversarial robustness [HMZ21]. Several works investigate training and EBM in tandem



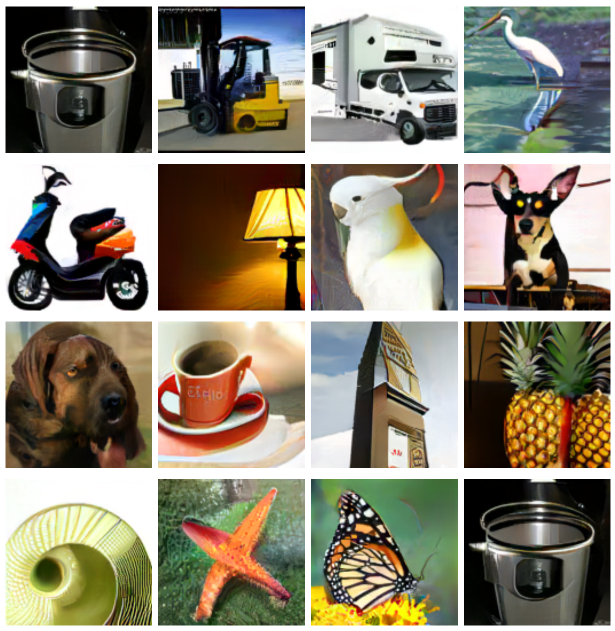


Figure 6.2: *Left:* The model takes a joint input (Y, Z) where Y is a residual image and Z is a latent vector. An image is generated using $X = Y + G(Z)$ for a generator $G(Z)$, and the image is passed to the hat network $H(X;\theta)$ to obtain the energy of the pair (Y, Z). This allows for principled probabilistic learning which can incorporate the latent space of any generator. *Right:* Unconditional ImageNet 128×128 samples generated by a Hat EBM.

with an auxiliary model. The work [KB16] jointly trains an EBM and generator without MCMC by using samples from the generator as direct approximations of the EBM density and training the generator using a variational objective. A similar approach is explored by [GKH21]. Cooperative learning [XZF21] trains the EBM and generator by using the generator to initialize samples needed to train the EBM and uses reconstruction loss between generator and EBM samples to learn the generator. The work [GNK20] learns an EBM using Noise Contrastive Estimation with an auxiliary flow model.

Latent Space EBM. EBMs in the data space can be highly multi-modal, and MCMC sampling can be difficult [XLZ16, NHZ19a, DM19]. Recent works [PHN20, PW21] explore learning an EBM in latent space, which is then mapped to the data space with a learned generator. The energy landscape in the latent space is smoother and less multi-modal because it lives on a much lower dimensional manifold than the data space. These works define a prior EBM in the latent space as a correction of the non-informative uniform prior or isotropic Gaussian prior. To learn the model, one needs to infer the posterior of the latent variables. Posterior inference given such a complicated model is non-trivial. One needs to either design a sophisticated amortized inference network or run expensive MCMC. Our model also defines an EBM in the latent space, while its learning does not need posterior inference, making the learning much simpler and more scalable. [CZS20] leverages a pretrained GAN to define an EBM in the latent space of the generator with a correction based on the discriminator, and shows improved synthesis quality over the pretrained GAN. The work [XKK21] uses a pretrained VAE to facilitate EBM learning. Our model is similar in that it can be utilized to improve pretrained GAN or VAE generators. Our method is however more general since it can be used to improve any pretrained generator, even the non-probabilistic ones like deterministic autoencoders.

6.4 Formulation of Hat EBM

This section presents the formulation of the Hat EBM energy function and the proposed learning procedure. We first review the fundamental equations of EBM learning. Then we introduce two variants of the Hat EBM: one for sampling from the latent space directly, and one for sampling a residual image conditioned on the generator output. Finally, we propose a method for learning the hat network and generator network of a Hat EBM at the same time so that our model can be used for self-contained image generation without the need for a pretrained generator.

6.4.1 Review of EBM Learning

Here we provide a very small review of EBM learning which is also provided in previous chapter. Our EBM learning follows the standard model defined in [Hin02, ZWM98, XLZ16]. The Deep Frame EBM follows the gibbs boltzman distribution:

$$p(x;\theta) = \frac{1}{\mathcal{Z}(\theta)} \exp\{-U(x;\theta)\} \tag{6.1}$$

where $U(x;\theta)$ is a deep neural network with weights θ and $\mathcal{Z}(\theta)$ is the intractable normalizing constant. Given a true but unknown data density $q(x)$, Maximum Likelihood (ML) learning uses the objective $\arg\min_\theta D_{KL}(q(x)||p(x;\theta))$, which can be minimized using the stochastic gradient

$$\nabla\mathcal{L}(\theta) \approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta U(X_i^+;\theta) - \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta U(X_i^-;\theta) \tag{6.2}$$

where the positive samples $\{X_i^+\}_{i=1}^n$ are a set of data samples which are also commonly referred to as q throughout this body of work. The negative samples $\{X_i^-\}_{i=1}^n$ are samples from the current EBM model $p(x;\theta)$. To obtain the negative samples for the EBM, we use a form of MCMC sampling known as Langevin Sampling with K steps.

$$X^{(k+1)} = X^{(k)} - \frac{\varepsilon^2}{2}\nabla_{X^{(k)}} U(X^{(k)};\theta) + \varepsilon V_k, \tag{6.3}$$

where ε is the step size and $V_k \sim N(0, I)$. The Langevin trajectories are initialized from a set of states $\{X_{i,0}^-\}_{i=1}^n$ obtained from a certain initialization strategy.

6.4.2 Hat EBM: Joint Distribution of Latent and Residual Image

The *Hat EBM* adapts a fixed pretrained generator network $G(Z)$ into an EBM. The Hat EBM defines the joint distribution of the random variable $Z \in \mathbb{R}^m$ in the m-dimensional latent space of the generator network and a random variable $Y \in \mathbb{R}^d$ in the d-dimensional image space. The joint energy has the form

$$U(Y, Z; \theta) = H(G(Z) + Y; \theta) \tag{6.4}$$

where $H(x; \theta)$ is a neural network that takes an image $x \in \mathbb{R}^d$ as input and returns a scalar output. The weights of H are given by θ. We call H the *hat network* because it sits atop the generator G to incorporate the generator latent space directly into the probabilistic model.

The random variable Y is meant to accommodate the gap between the output of $G(Z)$ and the image manifold (see fig 6.1). We expect that $G(Z)$ contains an approximate but imperfect representation of the image distribution which can further be refined by the residual state Y. The majority of the appearance of $G(Z) + Y$ comes from the generator network and not the residual image Y and therefore the majority of the sampling dynamics of our model are determined by movement in the latent space of $G(Z)$.

The Hat EBM formulation allows us to learn a model without calculating the log determinant of $G(Z)$ which would typically be required for an energy $U(G(Z); \theta)$ or for inferring the Z latent vectors associated with observed image samples X [PHN20]. This is true because we define the distribution of observed images X by $X = G(Z) + Y$ where each pair (Y, Z) is drawn from a true unknown density $q(y, z)$. Thus we can use the Maximum Likelihood learning framework (in Section 6.4.1) to learn the weights θ of the hat network $H(x; \theta)$ by minimizing $\arg\min_\theta D_{KL}(q(y, z) || p(y, z; \theta))$ where

$$p(y, z; \theta) = \frac{1}{\mathcal{Z}(\theta)} \exp\{-H(G(z) + y; \theta)\}. \tag{6.5}$$

One can obtain negative samples using alternating Langevin updates

$$Y^{(k+1)} = Y^{(k)} - \frac{\varepsilon_1^2}{2} \nabla_{Y^{(k)}} H(G(Z^{(k)}) + Y^{(k)}; \theta) + \varepsilon_1 V_{k,1} \tag{6.6}$$

$$Z^{(k+1)} = Z^{(k)} - \frac{\varepsilon_2^2}{2} \nabla_{Z^{(k)}} H(G(Z^{(k)}) + Y^{(k+1)}; \theta) + \varepsilon_2 V_{k,2} \tag{6.7}$$

which switches off between updates with respect to z and updates with respect to y. The algorithm is essentially Metropolis-within-Gibbs. This is because the Langevin update can be written as a Metropolis-Hastings step with Gaussian proposal. This shows that (6.6) and (6.7) define a valid sampler for $p(y, z; \theta)$. Finally, updating θ can be accomplished using

$$\nabla \mathcal{L}(\theta) \approx \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta H(X_i^+; \theta) - \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta H(G(Z_i^-) + Y_i^-; \theta) \tag{6.8}$$

where X_i^+ are observed image samples and the pairs (Y_i^-, Z_i^-) are obtained via MCMC. In our formulation the observed data are sufficient statistics for $H(x; \theta)$ so there's no need to infer the (Y, Z) pairs for the positive samples when learning the weights of the hat network.

6.4.3 Conditional Hat EBM: Residual Image Conditional on Latent Sample

The previous version of the Hat EBM is ammendable to any generator network $G(Z)$. In this section we define a conditional version of the Hat EBM tailored toward generator networks that map a trivial latent distribution to complex high dimensional manifolds. For these generators, we utilize the latent distribution as an ancestral distribution and learn a conditional distribution on the residual image Y given the latent sample z.

Suppose we use a trivial marginal distribution $p_0(z)$ for Z and a generator trained using this latent distribution. We can now define a conditional Hat EBM $p(y|z; \theta) = \frac{1}{\mathcal{Z}_z(\theta)} \exp\{-H(G(z) + y; \theta)\}$ and a joint density

$$p(y, z; \theta) = \frac{1}{\mathcal{Z}_z(\theta)} p_0(z) \exp\{-H(G(z) + y; \theta)\}. \tag{6.9}$$

In this case, we posit that observed images X are generated according to $X = G(Z) + Y$ for some distribution $q(y, z) = p_0(z)q(y|z)$. Extracting the negative samples (Z_i^-, Y_i^-) is achieves by first drawing Z_i^- from $p_0(z)$ and then obtaining $Y_i^- | Z_i^-$ using Langevin updates on the conditional probability $p(y|z; \theta)$. Note that $p(y|z; \theta) = p(y|G(z; \phi); \theta)$ because of the form of (6.9). We update θ using the same equation (6.8) as the joint Hat EBM because X_i^+ remains a sufficient statistic for learning $H(x; \theta)$ and because we do not need to infer the Z_i^+ for X_i^+ since the prior $p_0(z)$ does not contain any model parameters.

6.4.4 Learning the Hat Network and Generator in Tandem for Conditional Hat EBM

In the two previous formulations we assumed access to a pretrained generator network $G(z)$. Here we describe a self-contained method for learning the Hat EBM which trains both the hat network $H(x;\theta)$ parametrized by θ and the generator $G(z;\phi)$ parameterized by ϕ simultaneously. This method is based on coop-nets [XLG18]. First, we review the Coop-Nets learning framework and then present the learning strategy for the Hat EBM. A stark difference between the two is that the origin coop learning strategy requires MCMC inference of the latent space $\hat{Z}$ associated with an MCMC sample X to train $G(Z;\phi)$ whereas the conditional Hat EMB formulation does not because the formulation of the latent variables Z is well defined and does not need to be inferred. This lends a large computational advantage to the Hat EBM learning method since we don't need to compute MCMC inference on the hat $\hat{Z}$.

In cooperative learning [XLG18], the generator output $G(z;\phi)$ is trained to match the appearance of a Langevin chain X_K sampled from the potential $U(X;\theta)$ and initialized from the state $X_0 = G(Z_0;\phi)$ where $Z_0 \sim N(0,I)$. This model defines the conditional density of images X given latents $Z \sim N(0,I)$ as $X|Z \sim N(G(Z;\phi), \eta^2 I)$ for some sufficiently small η. Given a sampled state X_t, updating $G(z;\phi)$ requires inferring $Z|X_K$ using the latent Langevin equation

$$Z_{k+1} = Z_k - \frac{\varepsilon^2}{2}\left(Z_k + \frac{1}{2\eta^2}\nabla_{Z_k}\|G(Z_k;\phi) - X_K\|_2^2\right) + \varepsilon V_k \tag{6.10}$$

before updating ϕ using the Maximum Likelihood objective function

$$\mathcal{L}_G(\phi) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2\eta^2}\|G(Z_{K,i};\phi) - X_{K,i}\|_2^2 \tag{6.11}$$

where the i index denotes a member i of a batch with size n. In the code released with the coop learning paper, the latent variable is not inferred at all and $Z_0 \sim N(0,I)$ is used in place of Z_K in the objective function (6.11). Inferring Z_K often hurts model performance and leads to additional complications. Both the difficulty of inferring $Z|X_K$ and omission of this step leads to a gap in the cooperating learning formulation which can be bypassed using

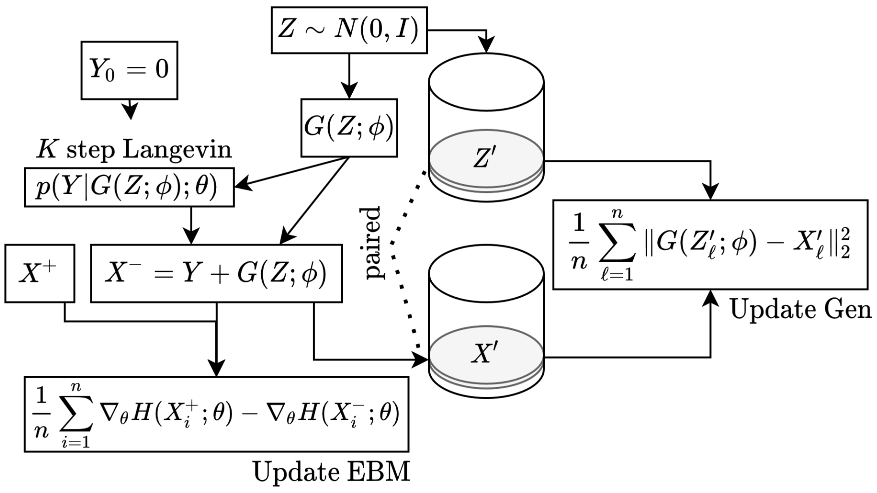

$Z \sim N(0, I)$
$Y_0 = 0$
$G(Z;\phi)$
K step Langevin
$p(Y|G(Z;\phi);\theta)$
Z'
$\frac{1}{n}\sum_{\ell=1}^{n} \|G(Z'_\ell;\phi) - X'_\ell\|_2^2$
paired
X^+
$X^- = Y + G(Z;\phi)$
Update Gen
X'
$\frac{1}{n}\sum_{i=1}^{n} \nabla_\theta H(X_i^+;\theta) - \nabla_\theta H(X_i^-;\theta)$
Update EBM

Figure 6.3: Visualization of tandem training method for hat network and generator. The left side illustrates training for the hat network $H(x;\theta)$. Z is drawn from a latent distribution, Y is initialized from the 0 image and updated according to $p(Y|Z;\theta) = p(Y|G(Z;\phi);\theta)$. Then data samples X^+ and negative samples $X^- = Y + G(Z;\phi)$ are used to update the weight θ of the hat network. On the right, pairs (X', Z') from past hat network updates will be drawn randomly from a bank of states to update the weight ϕ of the generator. The bank memory (X', Z') will then be overwritten by new pairs (X, Z) from the current model.

the Hat EBM generator update framework since we do not need to infer Z_k in the Hat EBM formulation.

To learn a Hat EBM generator, we initialize Langevin sampling from $X_0 = G(Z;\phi) + Y_0$ where $Y_0 = 0$ and $Z \sim N(0, I)$ to draw residual sample Y_K. The Langevin update is only used to update Y_k while Z remains fixed, as in the method from Section 6.4.3. We once more define our model as $X|Z \sim N(G(Z;\phi), \eta^2 I)$ except we further define the data distribution of (X, Z) as $Z \sim N(0, I)$ and $X = Y + G(Z;\phi_t)$ where $Y|Z$ is drawn from the Hat EBM density (6.9) using the generator $G(Z;\phi_t)$. Then $G(z;\phi)$ can be trained using the Maximum Likelihood objective

$$\phi_{t+1} = \arg\min_{\phi} \mathcal{L}_G(\phi;\phi_t) = \arg\min_{\phi} \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2\eta^2} \|G(Z_i;\phi) - (Y_{K,i} + G(Z_i;\phi_t))\|_2^2. \tag{6.12}$$

Conceptually, this loss function should allow $G(Z;\phi)$ to match the appearance of samples $X = Y + G(Z;\phi_t)$ for a fixed generator $G(Z;\phi_t)$ and the current hat network $H(x;\theta_t)$.

Rigorous training of H and G involves using the gradient of (6.12) until convergence and then updating the hat network according to (6.8), and continuing this cycle until the EBM converges. In practice, we implement (6.12) to obtain ϕ_{t+1} using only one update initialized from $\phi = \phi_t$ rather than training until full convergence. This is done to increase training efficiency and to avoid the need of maintaining a separate copy of generator weights for the fixed network $G(z; \phi_t)$.

We observe that the objective (6.12) has some limitations because the generator output can become too tethered to biases of the hat network which is a problem stemming from the original cooperative learning objective (6.11). For an illustration please see sec 6.4.5. To overcome these problems, we choose to train $G(z; \phi)$ at time $t + 1$ to match the historical distribution of hat $H(x; \theta_{t_\ell})$ and generator $G(z; \phi_{t_\ell})$ for a selection of past epochs $t_1, t_2, \ldots t_L \leq t$ instead of training the generator to match the distribution of the current networks $H(x; \theta_t)$ and generator $G(z; \phi_t)$. This involves redefining the data distribution (X, Z) by first sampling t_ℓ from $\{t_1, \ldots, t_L\}$ and then generating Z and $X_K = Y_K + G(Z; \phi_{t_\ell})$ where $Y|Z$ follows the energy $H(G(Z; \phi_{t_\ell}) + Y; \theta_{t_\ell})$. In this case, the gradient (6.12), replacing ϕ_t with ϕ_{t_ℓ}, is a stochastic approximation of the Maximum Likelihood gradient defined by the joint distribution (t_ℓ, Z, X_K). See section 6.4.5 for details. In practice, we implement this procedure by keeping a persistent bank of 10,000 pairs (X, Z) created from past hat network updates. When updating the generator, we draw $n = 128$ pairs from the bank and replace it with a newly generated batch of $n = 128$ pairs from the current hat EBM. This ensures that the selection $\{t_1, \ldots, t_L\}$ of past epochs always remains within a certain range of the current epoch t. Saving the generated images X at each EBM update allows us to learn from past generator weights ϕ_{t_ℓ} without maintaining a copy of the weights. See Figure 6.3 for an illustration.

6.4.5 Historical Generator Update to Improve EBM Memory and Stability

The Conditional Hat EBM makes use of historical EBM samples to update the generator instead of samples from the current EBM in memory to improve synthesis. If we use current EBM samples and not historical samples, the generator will fail because our MCMC trajectories would be too short and cannot have enough influence on samples coming out of the generator. Therefore, if the generator-derived samples lack diversity, our EBM will also lack diversity because we cannot move far enough away from the generator's manifold with a small number of MCMC steps (see fig 6.1). The idea for this was unrolled in the [HMC22a] work and we continue to implement this each time we use cooperative learning.

A Lack of sample diversity leads to EBM instability because the EBM will try to urgently shift its distribution to cover modes it thinks it never learned. This leads the EBM to forget previously learned modes in favor of new modes. Then once it learns new modes it forgets the old ones and the loop continues. If the EBM was a computer, using historical updates gives it more RAM.

We visualize the importance of our historical update in Figure 6.4. This figure compares cooperative learning [XLG18] with and without batch normalization to Hat EBM synthesis for CIFAR-10. One Hat EBM experiment uses only the current EBM to update the generator, while the other uses the historical approach we outline in the text. Neither of our Hat EBM experiments use batch norm. We see that the Hat EBM using historical updates is the most successful synthesis method.

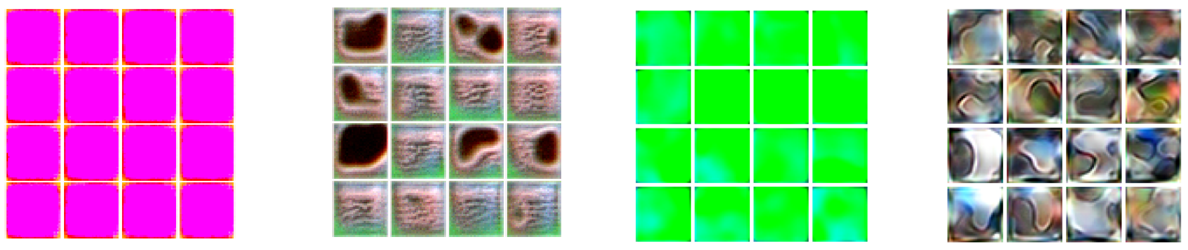
This historical update can be viewed as performing maximum likelihood where the data distribution (X, Z) is defined as the marginal of the joint distribution of (t_ℓ, X, Z) where $X = Y_K + G(Z; \phi_{t_\ell})$ and $Y_K|Z$ is sampled from

$$p(Y|Z, t_\ell) \propto -\exp\{H(Y + G(Z; \phi_{t_\ell}); \theta_{t_\ell})\}.$$

This is true because

$$\begin{aligned} E_{(t_\ell,Z,X)}[-\log p_G(X, Z; \phi)] &= E_{(t_\ell,Z,X)}\left[\frac{1}{2\eta^2}\|G(Z;\phi) - X\|_2^2\right] + C \\ &\approx \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2\eta^2}\|G(Z_i;\phi) - (Y_{K,i} + G(Z_i; \phi_{t_{\ell_i}}))\|_2^2 + C \end{aligned}$$

for n samples $\{(t_{\ell_i}, Z_i, X_i)\}_{i=1}^n$.



Coop. (no batch norm) Coop. (batch norm) Hat EBM (current) Hat EBM (historical)

Figure 6.4: Samples after 500 weight updates for different EBM learning methods. Cooperative learning fails without batch norm because the tether between the EBM and generator leads to lack of diversity and instability. Including batch norm in cooperative learning helps add some diversity, but samples can still remain very visually similar. The Hat EBM has similar problem as cooperative learning when current EBM samples are used to update the generator. This problem is greatly alleviated by instead using historical samples to update the generator, because several past updates of the EBM have much higher diversity than a single snapshot.

6.5 Experiments

In the subsequent empirical evaluations, we will address the following questions:

1. **Refinement:** To what extent can our method refine samples from a pretrained generator model with known prior distribution?

2. **Retrofit:** Is our method capable of turning a generator model pretrained as a deterministic auto-encoder into a generative model that samples realistic images?

3. **Synthesis:** Can we learn a generator from scratch for competitive image synthesis? Can our method be scaled up to challenging high dimensional datasets such as ImageNet with competitive synthesis for unconditional sampling?

4. **Out-of-Distribution:** Can the Hat EBM be used for Out-of-Distribution (OOD) detection to distinguish between samples from the training distribution and samples from dissimilar distributions?

Figure 6.5 visualizes some representative sampling paths for the models trained for questions 1, 2, and 3 on the CIFAR-10 dataset. For details related to pseudocode, training parameters, and model architectures please see the appendix.

6.5.1 Refinement

In this section we take a pretrained SNGAN with Generator network G and refine the samples. This situation was previously explored in [CZS20] and we build on top of this work from the lens of the joint Hat EBM from Section 6.4.2. We learn a Hat EBM that includes G in order to refine samples coming out of the generator. The architecture of the Hat EBM is the same as the SNGAN discriminator with the exclusion of spectral norm layers. We fix the parameters of the batch norm layers of the generator and our energy is determistic. We perform experiments for the CIFAR-10 dataset at $32x32$ resolution and CelebA at $64x64$ resolution. During evaluation we compare with Discriminator Driven Latent Sampling (DDLS) [CZS20], which uses a pretrained Discriminator D trained with G. DDLS samples form the potential $U(z) = D(G(z)) + \frac{1}{2}\|z\|_2^2$. We trained D and G according to the Mimicry [LT20] repository for reproducible GAN experiments which provides stronger baseline scores for $G(z)$ than those in [CZS20].

Table 6.1 shows our results. The Hat EBM is capable of refining an image better than DDLS on both CIFAR-10 and CelebA. We use the joint version of the Hat EBM so that both Y and Z are updated during sampling. We initialize $Y_0 = 0$ and $Z_0 \sim N(0, I)$. The sampling will learn to tilt Z away from its initial normal distribution to find a nearby latent vector with a more realistic appearance.

Table 6.1: Improvement in FID by refinement of samples from a fixed generator learned by SNGAN.

Model	CIFAR-10	CelebA
SNGAN (baseline) [MKK18]	18.58 ± 0.08	6.13 ± 0.03
DDLS [CZS20]	14.59 ± 0.07	6.06 ± 0.01
Hat EBM (*Ours*)	**14.04 ± 0.11**	**5.98 ± 0.02**

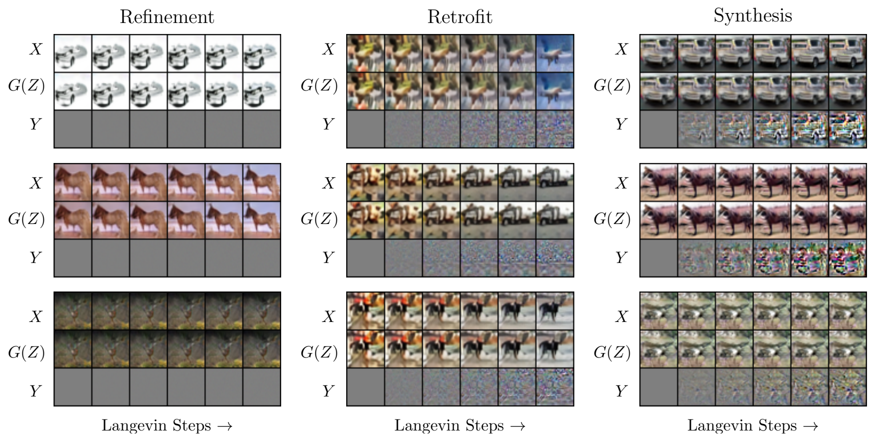


Figure 6.5: Visualization of shortrun Langevin sampling paths used for training and evaluating each Hat EBM type. In each grouping, the three rows correspond to image X, generator output $G(Z)$, and residual image Y. All paths are initialized with residual $Y_0 = 0$ which corresponds to a grey image. For visualization purposes, all Y images have been magnified by a factor of 5. The Refinement paths (*left*) use the joint Hat EBM to refine the appearance of a pre-trained SNGAN generator. The residual Y is barely noticeable, but still is essential for stable learning. The Retrofit paths (*center*) use the joint Hat EBM to sample using a non-probabilistic autoencoder generator. The residual Y is somewhat noticeable after magnification but most of the image appearance comes from $G(Z)$. The Synthesis paths (*right*) use the conditional Hat EBM to sample refine a generator learned in tandem. $G(Z)$ is fixed during sampling. The residual Y plays a significant role in refining the generator appearance. See section 6.7.1 for a broader explanation on why Y is important

We observe that the majority of the refinement occurs in the latent space, and that the residual image Y is essentially imperceptible. See Figure 6.5. We notice that training quickly becomes unstable when Y is removed from the Hat EBM. It's essential to incorporate the residual for stable learning. A possible reason for this phenomenon is that the hat network can learn to discriminate between generator images and images not from the generator, whether they are realistic or not. If so, the hat network can assign increasingly high energy to generator samples in the absence of the residual Y. Even an imperceptible Y appears enough to prevent the hat network from easily distinguishing positive and negative samples so that learning becomes stable.

6.5.2 Retrofit

In this section, we incorporate a non-probabilistic generator network $G(z)$ into a probabilistic Hat EBM model. This essentially allows us to sample from the latent space of $G(z)$ to find latent samples whose mapping corresponds to a realistic image. Similarly to results in Section 6.5.1, the residual image has small norm in the image space and most of the appearance of the sampled images comes from $G(z)$. This happens naturally without the need to coerce Y to be close to 0 by including a prior term such as $p_0(y) = \|y\|_2^2$, although including a prior term can further limit growth of Y.

The autoencoder generator $G(z)$ is pre-trained as the second half of a standard inference network and generator network pairing. An image X is fed into the inference network I and converted into a latent state $Z = I(X)$, which is then decoded with $\hat{X} = G(Z)$. The inference network and generator are learned jointly using the MSE loss $\|\hat{X} - X\|_2^2$. To keep the latent space mapping of $I(X)$ numerically stable, we project the raw output of the inference network to the sphere around the origin with radius $\sqrt{m}$ so that $\|I(X)\|_2 = \sqrt{m}$. More sophisticated methods such as perceptual and adversarial loss could have been used to train the autoencoder, but we use MSE loss to keep our implementation very simple. We observe that when Z is a vector-shaped latent state, it can be extremely difficult to achieve reconstructions $\hat{X}$ with sharp appearance even for simple datasets like CIFAR-10. To obtain better reconstructions and therefore a latent space with more realistic mappings to the image space, we use image-shaped latent states Z. The details of our autoencoder networks can be found in the appendix. When Z is an image shaped latent, we treat it exactly the same as a vector latent in the learning and sampling algorithms.

We experiment with assimilating an autoencoder into a Hat EBM potential for the CIFAR-10 dataset. Our results are presented in Figure 6.5 with additional results in the supplementary appendix. We train the Hat EBM using shortrun learning in the latent and image space by initializing $Y_0 = 0$ and $Z_0 \sim N(0, I)$ and using $K = 100$ MCMC steps of (6.6) and (6.7) from initialization during both training and testing evaluation to generate samples. During the Langevin dynamics, image appearance is refined mostly in the latent space. Our

Table 6.2: Comparison of FID scores among representative generative models. (*=EBM)

CIFAR-10 (32 × 32)

Model	FID
Hat EBM (*Ours*)*	**19.30 ± 0.15**
Improved CD EBM [DLT20]*	25.1
VERA [GKH21]*	27.5
Cooperative EBM[XLG18]*	33.6
Multigrid EBM [GNK20]*	37.3
JEM [GWJ20]*	38.4
IGEBM [DM19]*	40.6
DDPM [HJA20]	**3.2**
NCSNv2[SE20]	10.9
BigGAN [BDS19]	14.7
SNGAN [MKK18]	18.6

CelebA (64 × 64)

Model	FID
Hat EBM (*Ours*)*	**11.57 ± 0.04**
Divergence Triangle [HNF19]*	31.9
SNGAN [MKK18]	**6.1**
NCSNv2[SE20]	10.2

ImageNet (128 × 128)

Model	FID
Hat EBM (*Ours*)*	**40.24 ± 0.18**
SNGAN [MKK18]	65.7
SSGAN [CZR19]	43.9
InfoMax GAN [LTC21]	58.9
Hat EBM, scaled (*Ours*)*	29.37 ± 0.15
SSGAN, scaled [CZR19]	**23.4**

best model achieves a solid FID score of 26.01 ± 0.09. This demonstrates that Hat EBM can learn a probabilistic model over a non-probabilistic latent space.

6.5.3 Synthesis

In this section, we use the conditional Hat EBM formulation from Section 6.4.4 to learn a hat network and generator network from scratch for self-contained synthesis. We explore synthesis for CIFAR-10, CelebA at resolution 64×64, and unconditional ImageNet at resolution 128×128. While recent generative models show promising results for class-label conditional sampling, unconditional sampling with high quality synthesis remains a significant challenge. We find especially strong results for ImageNet synthesis using the conditional Hat EBM. This demonstrates strong potential for our synthesis method for learning with highly diverse unstructured datasets.

We use SNGAN architectures for sizes 32 × 32, 64 × 64, and 128 × 128, where the discriminator architecture is used for the hat network. During learning, we keep the generator batch norm parameters fixed to mean 0 and variance 1. We remove all spectral norm layers from the hat network. Training parameters can be found in the supplementary material. For ImageNet models, we found that annealing the generator and hat network learning rate by a factor of 10 after 250K weight updates for each network further improved the FID score significantly. Our results are shown in Table 6.2. See the supplementary material for uncurated samples from each model.

Results show strong performance compared to a selection of representative generative models across all datasets, with an especially strong performance for ImageNet. Our method significantly outperforms other EBM learning methods on CIFAR-10. The Hat EBM synthesis results are on par with the SNGAN baseline for CIFAR-10 and CelebA. The Hat EBM results for ImageNet significantly outperform SNGAN. At the budget of ~8 GPUs, our Hat EBM achieves a score of 40.0, outperforming the small-scale SSGAN.

To our knowledge, the current state-of-the-art score for unconditional ImageNet 128×128 is the SSGAN [CZR19] with a score of 23.4 trained using a BigGAN network and 128-core TPUv3 pods. To scale up our Hat EBM, we doubled the number of channel dimensions for both the hat network and generator network from the original SNGAN architecture and trained on 32-core TPU-v3 pods. Our best FID score for unconditional ImageNet 128×128 was 29.2, which comes within a competitive range of state-of-the-art. We believe that further scaling in future work could enable Hat EBM to match or surpass state-of-the-art. Our results decisively demonstrate the potential of EBM learning well beyond the scale investigated in any prior EBM work.

6.5.4 OOD

We assess our model performance on Out-Of-Distribution detection. We use the conditional Hat EBM model trained on CIFAR-10 from Section 6.5.3 and calculate the energy $H(X; \theta)$ on in-distribution images from the CIFAR-10 test set and the OOD datasets which include CIFAR-100, CelebA, and SVHN. We follow standard OOD evaluation from works such as [NMT19] which use the AUROC metric. This metric measures the ability of the Hat EBM to distinguish between in-distribution samples not seen during training and OOD samples.

Table 6.3: Comparison of OOD scores (AUROC) among representative models. We separate scores for fully unsupervised models (above the line) and models which used supervised data (below the line).

Model	SVHN	CIFAR-100	CelebA
Ours	**0.92**	**0.87**	**0.94**
IGEBM [DM19]	0.43	0.54	0.69
VAEBM [XKK21]	0.83	0.62	0.77
Improved CD EBM [DLT20]	0.91	0.83	–
JEM [GWJ20]	0.67	0.87	0.77
HDGE [LA21]	0.96	0.91	0.80
OOD EBM [LWO20]	0.91	0.87	0.78
OOD EBM (fine-tuned) [LWO20]	**0.99**	**0.94**	**1.00**

Following [GWJ20, XKK21], we expect that the energy of the OOD datasets be higher than the energy of in-distribution test images.

Our results are shown in Table 6.3. The Hat EBM shows strong performance as an OOD detection method. Among methods that are fully unsupervised, our model has the top performance across all three OOD datasets. Our method approaches the results of methods that are trained with labelled data such as HDGE [LA21] and the fine-tuned OOD EBM [LWO20], although we do not yet match these scores. Overall, there is strong evidence at the Hat EBM is naturally an effective method for OOD detection, especially when supervised label information is unavailable.

6.6 Conclusion

Maximum-likelihood based learning of EBMs is challenging since we must draw negative samples from the current density model, which is often highly multi-modal. Prior art addresses this challenge by recruiting approximations of the EBM in the form of an ancestral sampling from a generator model, truncated Langevin chains, flow-based models, or lifting the EBM into the induced latent space of generator models. In contrast, the Hat EBM work proposes a method for absorbing any generator as a backbone of an EBM. The formulation assumes that observed images are the sum of unobserved latent variables pushed forward through the generator and a residual random variable which closes the gap between generator samples and image manifold (see fig 6.1). Thus, the *Hat EBM* sits atop the generator. The generator allows for efficient sampling but may only capture the coarse structure of the images, while the residuals can capture fine-grained or even imperceptible details.

The Hat EBM formulation is presented in three variations: (1) joint learning of latent and residual image for adapting any fixed generator, (2) conditional learning for generators with known prior distribution, (3) self-contained learning of both EBM and generator from scratch. Notably, the training doesn't require computing the log determinant of the generator Jacobian or inference of latent variables which makes learning much more scalable and simpe to do.

Empirical evaluations demonstrate the various capabilities of the Hat EBM: (1) Strong performance for the ImageNet synthesis at 128×128 resolution with self-contained learning, (2) Significant refinement of the quality of synthesis of pre-trained generators on CIFAR-10 and CelebA with conditional learning, (3) Retrofitting pre-trained auto-encoder generators with a means of sampling, and (4) Out-of-Distribution detection with state-of-the-art performance for unsupervised models.

6.7 Appendix

6.7.1 Importance of Residual Image for Stability

Throughout our experiments with different versions of the Hat EBM, we find the inclusion of the residual image Y essential for stability. In particular, one could consider an alternate version of the Hat EBM where

$$U(z;\theta) = H(G(z);\theta) \tag{6.13}$$

without a residual state. As long as the training data is of the form $X^+ = G(Z^+)$ for a latent state Z^+, one could learn the hat network using the same procedure as the Hat EBM without the residual Y. In practice, it is usually not possible to exactly invert the generator. In other words, real images X^+ never lie exactly on the generator output manifold, although the might be close by. Nonetheless, one could bend the rules and use X^+ to train the potential (6.14) with the justification that there is some Z^+ such that $X^+ \approx G(Z^+)$. In practice, this leads to instability as shown in Figure 6.7. Even when it is nearly invisible, the residual state Y is still required for the hat network to balance the energy of positive and negative samples and achieve stable learning.

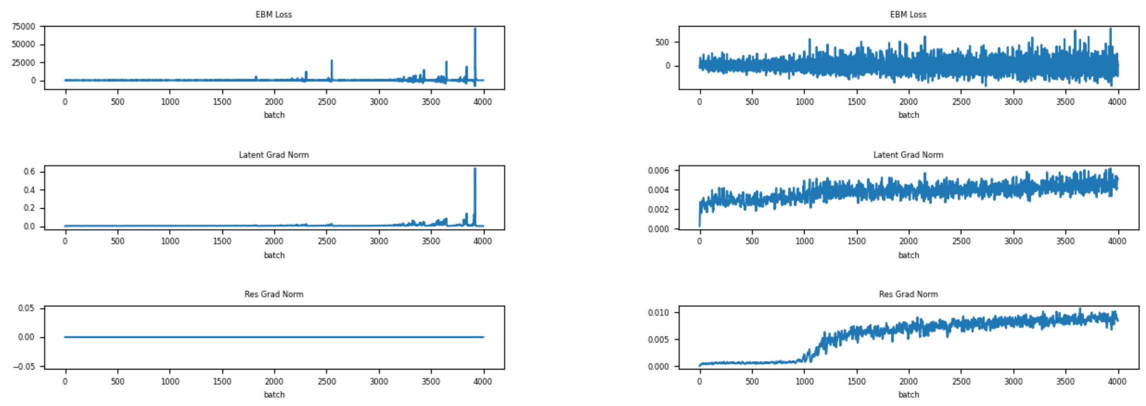


Figure 6.6: Unstable learning using the energy (6.14) *(left)* and stable learning using the joint Hat EBM *(right)*. Both settings replicate the refinement experiment using CIFAR-10 with a pretrained SNGAN generator. Even though the appearance of the residual image Y is nearly invisible, including the residual is essential both from a theoretical perspective and for practical stability. This is explicitly apparent when logging the gradient magnitude of the residual during learning.

6.7.2 Algorithm for Joint Training of Conditional Hat EBM

A code sketch of training our Conditional Hat EBM for image synthesis is presented below. Note that the equation numbers in the Algorithm refer to equations in the main paper and not equations in the appendix.

Algorithm 5 Training a Conditional Hat EBM for Image Synthesis

```
Require: Natural images $\{x_m^+\}_{m=1}^M$, EBM $U(x;\theta)$, generator $G(z;\phi)$ Langevin noise $\varepsilon$, number of shortrun steps $K$, EBM
    optimizer $h_U$, generator optimizer $h_G$, random initial weights $\theta_0$ and $\phi_0$, number of training iterations $T$, bank size $N$.
Ensure: Learned weights $\theta_T$ for EBM and $\phi_T$ for generator.
  Initialize bank of random latent states $\{Z_i\}_{i=1}^N$ i.i.d. from the Gaussian $N(0, I)$.
  Initialize image bank $\{X_i^-\}_{i=1}^N$ from generator using $X_i^- = g(Z_i; \phi_0)$
  for $1 \leq t \leq T$ do
      Steps to Update EBM
      Select batch $\{\tilde{X}_b^+\}_{b=1}^B$ from data samples $\{x_m^+\}_{m=1}^M$.
      Draw latent samples $\{\tilde{Z}_b\}_{b=1}^B$ i.i.d. from the Gaussian $N(0, I)$.
      Initialize residual images $\{\tilde{Y}_{b,0}^-\}_{b=1}^B$ from the image with all pixels set to 0.
      Update residual images $\{\tilde{Y}_{b,0}^-\}_{b=1}^B$ with $K$ Langevin steps of Equation 6 to obtain $\{\tilde{Y}_{b,K}^-\}_{b=1}^B$. Keep $\tilde{Z}_b$ fixed.
      Sum generated image and residual image using $\tilde{X}_b^- = G(\tilde{Z}_b, \phi_{t-1}) + \tilde{Y}_{b,K}^-$ to obtain negative samples $\{\tilde{X}_b^-\}_{b=1}^B$.
      Get learning gradient $\Delta_U^{(t)}$ using Equation 8 with samples $\{\tilde{X}_b^+\}_{b=1}^B$ and $\{\tilde{X}_b^-\}_{b=1}^B$.
      Update $\theta_t$ using gradient $\Delta_U^{(t)}$ and optimizer $h_U$.

      Steps to Update Generator
      Randomly choose unique indices $\{i_1, \ldots, i_B\} \subset \{1, \ldots, N\}$.
      Get paired batches $\{Z_{i_b}\}_{b=1}^B$ and $\{X_{i_b}^-\}_{b=1}^B$ from $\{Z_i\}_{i=1}^N$ and $\{X_i^-\}_{i=1}^N$.
      Get learning gradient $\Delta_G^{(t)}$ using Equation 11 with samples $\{Z_{i_b}\}_{b=1}^B$ and $\{X_{i_b}^-\}_{b=1}^B$.
      Update $\phi_t$ using gradient $\Delta_G^{(t)}$ and optimizer $h_G$.
      Overwrite old states $\{Z_{i_b}\}_{b=1}^B$ and $\{X_{i_b}^-\}_{b=1}^B$ in bank with update $Z_{i_b} \leftarrow \tilde{Z}_b$ and $X_{i_b}^- \leftarrow \tilde{X}_b^-$.
  end for
```

6.7.3 Discussion of EBM Synthesis Methods

This appendix briefly introduces the EBM methods, including methods presented in Table 2 of the main paper, and draws relevant comparisons between the Hat EBM and other EBM models.

One branch of EBM works uses MCMC-based Maximum Likelihood with persistent initialization of MCMC states. Persistent initialization uses samples of prior short run EBM trajectories to initialize the current sampling trajectory. This approach is introduced by Persistent Contrastive Divergence (PCD) [Tie08]. The IGEBM [DM19] is trained using a bank with 10,000 images to hold persistent states. States are rejuvenated from a Gaussian or uniform noise image with of between 0.5% and 5% probability before being returned to the image bank. The Improved CD EBM [DLT20] builds on these results by including an approximate KL divergence term in EBM learning to minimize the difference between the data distribution and the sampled distribution, and by rejuvenating MCMC trajectories using data augmentation instead of resetting states with noise. The Joint Energy Model (JEM) [GWJ20] trains an unconditional EBM and a classifier model simultaneously with the same network using persistent initialization with noise rejuvenation. The use of persistent states in our work differs from prior work because we use persistent states to update only the generator while the EBM is updated by states generated from scratch in the current iteration. This is done to increase the diversity of samples used to update the generator, which is essential for enabling the generator to create distinct appearances for different Z early in training (see Appendix 6.4.5).

Another branch of EBM works trains a generator network in tandem with the energy network. Most works use the standard EBM update or a close variant to train the energy network, as we do. In some works, the generators produce the final samples and no MCMC is used, while other works use the generator to initialize samples and then refine the samples with MCMC driven by the energy network. Our work adopts the second strategy. To our knowledge, the first work that explores the idea jointly training an energy network and generator network is by Kim & Bengio [KB16]. This work suggests using the generator samples directly as negative samples without use of MCMC, and updating the generator network to decrease the energy of the generator samples. The EGAN [DAB17] builds on [KB16] by introducing a entropy maximization term which is needed for a valid Maximum Likelihood objective and which prevents generator collapse. The entropy term is estimated by neighborhood methods and variational methods. MEG [KGC19] and VERA [GKH21] build on [DAB17] by introducing more sophisticated methods of entropy maximization. The GEBM [AZG21] uses an approach similar to [DAB17], with the major differences being use of a generalized log likelihood objective that bridges the gap between the support of the generator output and the full image space distribution of the data, and a novel approximate KL bound for learning the generator. Like the Hat EBM, none of these methods require the log determinant of the generator Jacobian or inference of latent states for data. Unlike the Hat EBM, the probability models from these methods lie in the latent space (or the restricted image space given by the generator outputs) instead of the full image space. The methods are also incompatible with non-probabilistic generators, unlike Hat EBM. None of the works above use MCMC during training, although some use MCMC during synthesis [GKH21, AZG21]. Cooperative learning [XLG18] uses Maximum Likelihood learning described in Section 3.4. This requires MCMC sampling for both image and latent states. The conditional Hat EBM for synthesis requires sampling for image states but not latent states.

A third branch of EBM methods initialize MCMC sampling from a noise distribution and use a fixed length MCMC trajectory to generate states without a generator network. This branch differs from persistent methods because no persistent bank is used and negative samples to update the EBM are created from scratch each time the EBM weights are updated. It differs from generator methods because realistic synthesis is achieved through pure MCMC without initial realistic states from the generator. The Multigrid EBM [GLZ18] has a MCMC-based training method where images are synthesized and sampled at multiple resolutions. Multiple EBMs are learned in parallel at different resolutions, and generated images from low resolution EBMs are passed to high resolution EBMs to initialize MCMC sampling. Generation can be performed by trivial sampling (uniform, Gaussian Mixture, KDE, etc.) at a single-pixel resolution and passing the generated MCMC states along from the single-pixel EBM to the full-size EBM. The short run initialization method [NHZ19a] starts sampling from a uniform image distribution and runs 50 to 100 MCMC steps to generate images during each EBM update, bypassing the need for persistent banks. Our retrofit Hat EBM training is a variation of the short run method where both the Y and Z are initialized from uniform noise. Since the generator is non-probabilistic, the short run trajectories of Z must move from uniform latent samples that represent noisy images to tuned latent samples whose generated images match the data appearance.

6.7.4 Importance of Residual Image for Stability

Throughout our experiments with different versions of the Hat EBM, we find the inclusion of the residual image Y essential for stability. In particular, one could consider an alternate version of the Hat EBM where

$$U(z;\theta) = H(G(z);\theta) \tag{6.14}$$

without a residual state. As long as the training data is of the form $X^+ = G(Z^+)$ for a latent state Z^+, one could learn the hat network using the same procedure as the Hat EBM without the residual Y. In practice, it is usually not possible to exactly invert the generator. In other words, real images X^+ never lie exactly on the generator output manifold, although the might be close by. Nonetheless, one could bend the rules and use X^+ to train the potential (6.14) with the justification that there is some Z^+ such that $X^+ \approx G(Z^+)$. In practice, this leads to instability as shown in Figure 6.7. Even when it is nearly invisible, the residual state Y is still needed for the hat network to balance the energy of positive and negative samples and achieve stable learning.

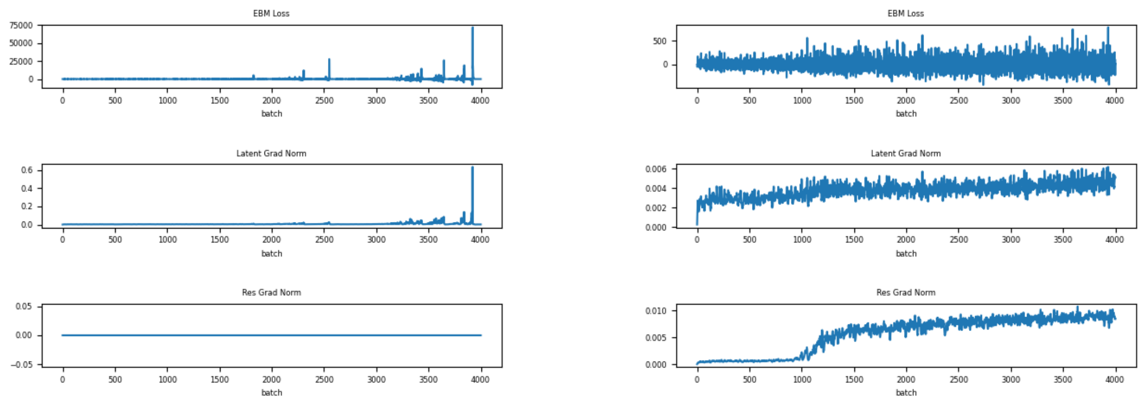


Figure 6.7: Unstable learning using the energy (6.14) *(left)* and stable learning using the joint Hat EBM *(right)*. Both settings replicate the refinement experiment using CIFAR-10 with a pretrained SNGAN generator. Even though the appearance of the residual image Y is nearly invisible, including the residual is essential both from a theoretical perspective and for practical stability.

6.7.5 Hyperparameters

Synthesis Training

Dataset	Celeb-A	CIFAR-10	ImageNet
Training Steps	100000	100000	300000
EBM LR	1e-4	1e-4	1e-4
EBM Optimizer	Adam	Adam	Adam
EBM Gradient Clip	0	0	50
Langevin Epsilon	1e-4	1e-4	1e-4
MCMC Steps	50	50	50
MCMC Temperature	1e-8	1e-3	1e-8
Persistent Bank Size	10000	10000	10000
Generator LR	1e-4	1e-4	5e-5
Generator Optimizer	Adam	Adam	Adam

Retrofit Training

Dataset	CIFAR-10
Training Steps	100000
Data Epsilon	1e-3
EBM LR	1e-4
EBM Optimizer	Adam
Image Space Epsilon	5e-3
Latent Epsilon	5e-3
MCMC Steps	100
MCMC Temperature	1e-3

Refinement Training

Dataset	Celeb-A	CIFAR-10
Training Steps	150000	150000
Data Epsilon	2e-2	2e-2
EBM LR	1e-4	1e-4
EBM Optimizer	Adam	Adam
Image Space Epsilon	1e-4	1e-4
Latent Epsilon	5e-3	5e-3
MCMC Steps	100	100
MCMC Temperature	1e-4	1e-4

6.7.6 Visualization of Synthesis Results

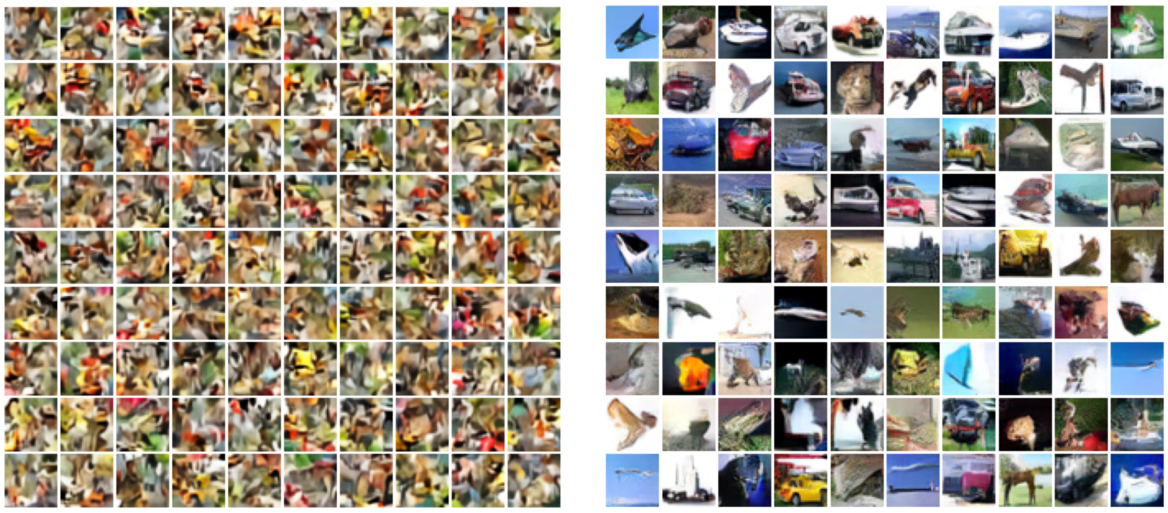


Figure 6.8: Initial image states *(left)* and sampled image states *(right)* for retrofit Hat EBM that uses a pretrained generator from a deterministic autoencoder. The training dataset is CIFAR-10.

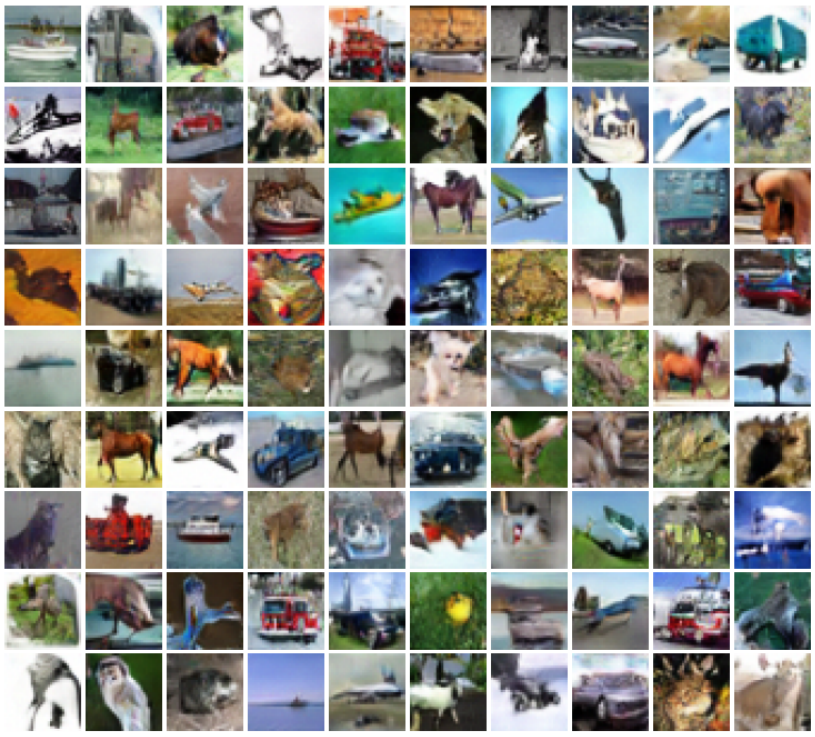


Figure 6.9: Uncurated Hat EBM samples for CIFAR-10 at resolution 32×32.

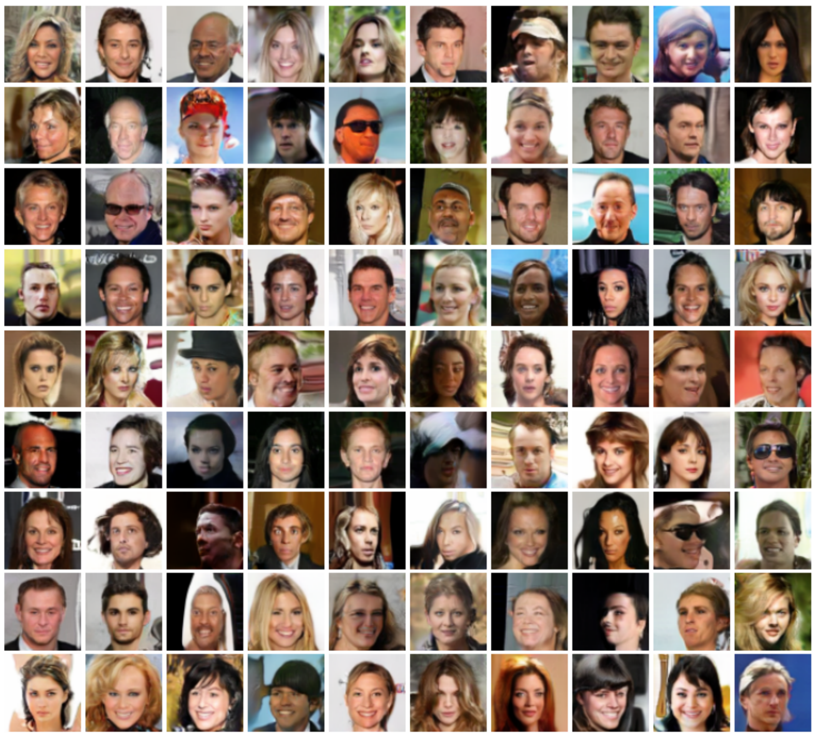


Figure 6.10: Uncurated Hat EBM samples for Celeb-A at resolution 64×64.

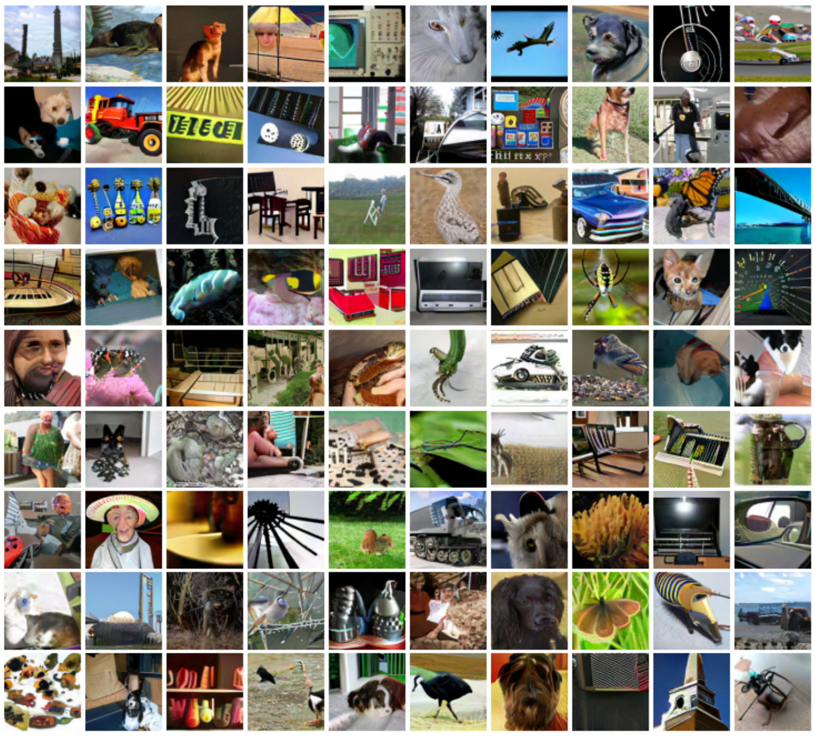


Figure 6.11: Uncurated Hat EBM samples for unconditional ImageNet at resolution 128×128.

CHAPTER 7

Conclusion

7.1 Themes

7.1.1 Compute and Capacity

A common theme in this body of work is that we keep going bigger and constructing larger models. As our models got bigger and better, I grew our infrastructure at the same time. MCMC is very compute intensive because you have to sample. I started out with my first lab issued GPU, a 1080. Later, in 2019 I built a 4 GPU server called "Odin" which we used to run all the experiments for the Stochastic Security paper. After that we received a generous TPU grant from Google for 5 TPUV3-8 and 5 TPUV2-8, we increased compute to train large scaled EBMs on ImageNet, CelebA, CIFAR10, Pets, Flowers, COCO, and other 4D medical datasets.

The architecture of our EBMs increased with our compute. At first, we had a very simple 4-layer Convolutional Neural Network EBM model. For the Lifecycle paper we started with a ResNet18-like architecture [ZK16], and eventually settled on an SNGAN64-like [MKK18] architecture. In the Hat EBM chapter we scaled this even further and utilized TPUV3-32 nodes with data parallelism to scale even higher.

7.1.2 Historical Updates and Paired Banks

Another theme is the idea of using the paired image and latent vector banks to use historical samples during learning. We found that by storing older samples we can mimic a very long MCMC trajectory without sacrificing stability. In the lifecycle and Hat EBM chapters we re-use the idea of paired bank storage and use of historical samples during learning to make the models incredibly stable and diverse. This is why our results were so strong. I believe going forward the wider community will adapt these strategies to drastically increase EBM model diversity during trainining.

7.1.3 Tradeoff between natural image accuracy and robustness

Another common theme is the trade-off between natural image accuracy and adversarial robustness. If we use a method, that has great robustness it must decrease the natural image accuracy because we are extending the inference capacity to encompass a much larger manifold then the natural image distribution. Any method, whether it be Adversarial Training (AT) or Purification-based defense will lower natural image accuracy. While the current state of the art for robustness against white-box attacks is still much lower then 80% there currently still room for growth. Eventually I believe that the natural image accuracy of a defense approach will asymptotically approach natural image classifier performance.

We can see in the "Fixer" chapter that by improving our upper bound on $F(x_{\text{nat}}))$ we also improve performance on $F(x_{\text{adv}})$.

7.1.4 Chaotic Dynamics

I didn't highlight this very much but the connection between chaotic dynamics and adversarial defense is incredibly fascinating. The fact that our defense only emerges at the border of chaotic and ordered lyapunov exponents is very interesting and we hope to explore this further in the future.

7.1.5 General Conclusion

As you have seen throughout this thesis, we are able to defend against a wide variety of datasets, and tasks using the Energy-based model [WXZ18] [NHH19] [HMZ21]. By harnessing metastable behavior of this model we can make off the shelf classifiers very robust towards the strongest-known adversarial attacks. Today we can see new models building on top of the same framework that Stochastic Security developed. [NGH22] [YHL21]. Adversarial attacks are one of the central limitations of deep learning models today and we hope that in the future our defense methods are so robust that they won't be a problem.

When developing adversarial defense research I also encountered some new ideas for unconditional image synthesis along the way. This work led to the Hat EBM and the short-run sampler in the Lifecycle paper.

I have described the progression of my research in adversarial defense from laying the foundational groundwork to scaling the models to finding new architectures and improvements along the way. We hope to continue this research in the future and incorporate new ideas and architectures as the field of Deep Learning evolves.

REFERENCES

[ACW18] Anish Athalye, Nicholas Carlini, and David Wagner. “Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples.” In *Proceedings of the 35th International Conference on Machine Learning*, pp. 274–283, 2018.

[AHS85] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. “A Learning Algorithm for Boltzmann Machines.” *Cognitive Science*, **9**(1):147–169, 1985.

[Ant19] Jason Antic. “Decrappification, deoldification, and Super Resolution.”, May 2019.

[AUH19] Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. “Are Labels Required for Improving Adversarial Robustness?” In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[AZG21] Michael Arbel, Liang Zhou, and Arthur Gretton. “Generalized Energy Based Models.” In *International Conference on Learning Representations*, 2021.

[BDS19] Andrew Brock, Jeff Donahue, and Karen Simonyan. “Large Scale GAN Training for High Fidelity Natural Image Synthesis.” In *International Conference on Learning Representations*, 2019.

[BGH19] Yogesh Balaji, Tom Goldstein, and Judy Hoffman. “Instance Adaptive Adversarial Training: Improved Accuracy Tradeoffs in Neural Nets.” *arXiv preprint arXiv:1910:08051*, 2019.

[BGS76] Giancarlo Benettin, Luigi Galgani, and Jean-Marie Streclyn. “Kolmogorov entropy and numerical experiments.” *Physical Review A*, **14**(6):2338–2345, 1976.

[BH06] Anton Bovier and Frank den Hollander. “Metastability: A potential theoretic approach.” *International Congress of Mathematicians*, **3**:499–518, 2006.

[BRR18] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. “Thermometer Encoding: One Hot Way To Resist Adversarial Examples.” In *International Conference on Learning Representations*, 2018.

[BZ19] Adrian Barbu and Song-Chun Zhu. *Monte Carlo Methods.* Springer, 2019.

[CFG14] Tianqi Chen, Emily B. Fox, and Carlos Guestrin. “Stochastic Gradient Hamiltonian Monte Carlo.”, 2014.

[CH20] Francesco Croce and Matthias Hein. “Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks.” In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[CRK19] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. "Certified Adversarial Robustness via Randomized Smoothing." In *Proceedings of the 36th International Conference on Machine Learning*, pp. 1310–1320, 2019.

[CRS19] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. "Unlabeled Data Improves Adversarial Robustness." In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[CZR19] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. "Self-supervised gans via auxiliary rotation loss." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12154–12163, 2019.

[CZS20] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. "Your GAN is secretly an energy-based model and you should use discriminator driven latent sampling." *arXiv preprint arXiv:2003.06060*, 2020.

[DAB17] Zihang Dai Dai, Amjad Almahairi, Philip Bachman, Eduard Hovy, and Aaron Courville. "Calibrating Energy-based Generative Adversarial Networks." In *International Conference on Learning Representations*, 2017.

[DAB18] Guneet S. Dhillon, Kamyar Azizzadenesheli, Jeremy D. Bernstein, Jean Kossaifi, Aran Khanna, Zachary C. Lipton, and Animashree Anandkumar. "Stochastic activation pruning for robust adversarial defense." In *International Conference on Learning Representations*, 2018.

[DDS09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database." In *CVPR09*, 2009.

[Dic45] Lee R. Dice. "Measures of the Amount of Ecologic Association Between Species." *Ecology*, **26**(3):297–302, 1945.

[DLT20] Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. "Improved contrastive divergence training of energy based models." *arXiv preprint arXiv:2012.01316*, 2020.

[DM19] Yilun Du and Igor Mordatch. "Implicit Generation and Modeling with Energy Based Models." In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[DPA21] Laura Daza, Juan C. Pérez, and Pablo Arbeláez. "Towards Robust General Medical Image Segmentation.", 2021.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[GKH21] Will Sussman Grathwohl, Jacob Jin Kelly, Milad Hashemi, Mohammad Norouzi, Kevin Swersky, and David Duvenaud. "No {MCMC} for me: Amortized sampling for fast and stable training of energy-based models." In *International Conference on Learning Representations*, 2021.

[GLZ18] Ruiqi Gao, Yang Lu, Junpei Zhou, Song-Chun Zhu, and Ying Nian Wu. "Learning generative convnets via multi-grid modeling and sampling." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9155–9164, 2018.

[GM06] Ulf Grenander and Michael I Miller. *Pattern Theory: From representation to inference.* Oxford University Press, 12 2006.

[GNK20] Ruiqi Gao, Erik Nijkamp, Diederik P Kingma, Zhen Xu, Andrew M Dai, and Ying Nian Wu. "Flow contrastive estimation of energy-based models." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7518–7528, 2020.

[GPM14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." *Advances in neural information processing systems*, **27**, 2014.

[GRC18] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. "Countering Adversarial Images using Input Transformations." In *International Conference on Learning Representations*, 2018.

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." In *International Conference on Learning Representations*, 2015.

[GWJ20] 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*, 2020.

[Hin02] Geoffrey E. Hinton. "Training Products of Experts by Minimizing Contrastive Divergence." *Neural Computation*, **14**(8):1771–1800, 2002.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models.", 2020.

[HMC22a] Mitch Hill, Jonathan Mitchell, Chu Chen, Yuan Du, Mubarak Shah, and Song-Chun Zhu. "EBM Life Cycle: MCMC Strategies for Synthesis, Defense, and Density Modeling.", 2022.

[HMC22b] Mitch Hill, Jonathan Mitchell, Chu Chen, Yuan Du, Mubarak Shah, and Song-Chun Zhu. "EBM Life Cycle: MCMC Strategies for Synthesis, Defense, and Density Modeling.", 2022.

[HMZ21] Mitch Hill, Jonathan Craig Mitchell, and Song-Chun Zhu. "Stochastic Security: Adversarial Defense Using Long-Run Dynamics of Energy-Based Models." In *International Conference on Learning Representations*, 2021.

[HNF19] Tian Han, Erik Nijkamp, Xiaolin Fang, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. "Divergence triangle for joint training of generator model, energy-based model, and inferential model." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8670–8679, 2019.

[HRF19] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. "Parametric Noise Injection: Trainable Randomness to Improve Deep Neural Network Robustness Against Adversarial Attack." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 588–597, 2019.

[HRU17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium." In *Advances in Neural Information Processing Systems*, volume 33, p. 6629–6640, 2017.

[HZR16] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition." *CVPR*, 2016.

[JAF16] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual Losses for Real-Time Style Transfer and Super-Resolution.", 2016.

[KB15] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization." In *International Conference on Learning Representations*, 2015.

[KB16] Taesup Kim and Yoshua Bengio. "Deep Directed Generative Models with Energy-Based Probability Estimation.", 2016.

[KD18] Diederik P Kingma and Prafulla Dhariwal. "Glow: Generative flow with invertible 1x1 convolutions." *arXiv preprint arXiv:1807.03039*, 2018.

[KGC19] Rithesh Kumar, Anirudh Goyal, Aaron C. Courville, and Yoshua Bengio. "Maximum Entropy Generators for Energy-Based Models." *ArXiv*, **abs/1901.08508**, 2019.

[KS17] Nitish Shirish Keskar and Richard Socher. "Improving Generalization by Switching from Adam to SGD." *arXiv preprint arXiv:1712.07628*, 2017.

[KW13] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114*, 2013.

[LA21] Hao Liu and Pieter Abbeel. "Hybrid Discriminative-Generative Training via Contrastive Learning.", 2021.

[LLB03] Ying-Cheng Lai, Zonghua Liu, Lora Billiings, and Ira B. Schartz. "Noise-induced unstable dimension variability and transition to chaos in random dynamical systems." *Physical Review E*, **67**, 2003.

[LO21] Kyungmin Lee and Seyoon Oh. "Efficient randomized smoothing by denoising with learned score function.", 2021.

[LT20] Kwot Sin Lee and Christopher Town. "Mimicry: Towards the reproducibility of gan research." *arXiv preprint arXiv:2005.02494*, 2020.

[LTC21] Kwot Sin Lee, Ngoc-Trung Tran, and Ngai-Man Cheung. "Infomax-gan: Improved adversarial image generation via information maximization and contrastive learning." In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3942–3952, 2021.

[LTH16] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network.", 2016.

[LWO20] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. "Energy-based Out-of-distribution Detection." *Advances in Neural Information Processing Systems*, 2020.

[LXF18] Kwonjoon Lee, Weijian Xu, Fan Fan, and Zhuowen Tu. "Wasserstein introspective neural networks." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3702–3711, 2018.

[MKK18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. "Spectral Normalization for Generative Adversarial Networks." In *International Conference on Learning Representations*, 2018.

[MMS18] 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*, 2018.

[MZ23] Bo Pang Jonathan Mitchell Mitch Hill, Erik Nijkamp and Song-Chun Zhu. "Learning Probabilistic Models from Generator Latent Spaces with Hat EBM." *NeurIPS (in review)*, 2023.

[Nea12] Radford M. Neal. "MCMC using Hamiltonian dynamics.", 2012.

[NGH22] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. "Diffusion Models for Adversarial Purification.", 2022.

[NHH19] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. "On the Anatomy of MCMC-Based Maximum Likelihood Learning of Energy-Based Models.", 2019.

[NHH20] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. "On the Anatomy of MCMC-based Maximum Likelihood Learning of Energy-Based Models." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 2020.

[NHZ19a] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. "Learning Non-Convergent Non-Persistent Short-Run MCMC Toward Energy-Based Model." In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[NHZ19b] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. “On Learning Non-Convergent Non-Persistent Short-Run MCMC Toward Energy-Based Model.”, 2019.

[NMT19] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. “Do Deep Generative Models Know What They Don’t Know?” In *International Conference on Learning Representations*, 2019.

[NZ08] M-E. Nilsback and A. Zisserman. “Automated Flower Classification over a Large Number of Classes.” In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

[OKV16] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. “Conditional image generation with pixelcnn decoders.” *arXiv preprint arXiv:1606.05328*, 2016.

[PHN20] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. “Learning latent space energy-based prior model.” *Advances in Neural Information Processing Systems*, **33**:21994–22008, 2020.

[PVZ12] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. “Cats and Dogs.” In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[PW21] Bo Pang and Ying Nian Wu. “Latent space energy-based model of symbol-vector coupling for text generation and classification.” In *International Conference on Machine Learning*, pp. 8359–8370. PMLR, 2021.

[QMG19] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. “Adversarial Robustness through Local Linearization.”, 2019.

[RDS14] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge.”, 2014.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation.” In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, Cham, 2015. Springer International Publishing.

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks.” *arXiv preprint arXiv:1511.06434*, 2015.

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models." In *International conference on machine learning*, pp. 1278–1286. PMLR, 2014.

[RSF19] Edward Raff, Jared Sylvester, Steven Forsyth, and Mark McLean. "Barrage of random transforms for adversarially robust defense." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6528–6537, 2019.

[SE19] Yang Song and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." *arXiv preprint arXiv:1907.05600*, 2019.

[SE20] Yang Song and Stefano Ermon. "Improved Techniques for Training Score-Based Generative Models.", 2020.

[SH09] Ruslan Salakhutdinov and Geoffrey E. Hinton. "Deep Boltzmann Machines." In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009*, pp. 448–455, 2009.

[SKC18] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. "Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models." In *International Conference on Learning Representations*, 2018.

[SKN18] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. "PixelDefend: Leveraging Generative Models to Understand and Defend against Adversarial Examples." In *International Conference on Learning Representations*, 2018.

[SLJ14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going Deeper with Convolutions.", 2014.

[SMM19] Vignesh Srinivasan, Arturo Marban, Klaus-Robert Muller, Wojciech Samek, and Shinichi Nakajima. "Defense Against Adversarial Attacks by Langevin Dynamics." *arxiv preprint arXiv:1805.12017*, 2019.

[SNG19] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. "Adversarial Training for Free!", 2019.

[SSY20] Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J. Zico Kolter. "Denoised Smoothing: A Provable Defense for Pretrained Classifiers." In *Advances in Neural Information Processing Systems*, volume 33, pp. 21945–21957, 2020.

[SZ14] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *arXiv 1409.1556*, 09 2014.

[TCB20] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. "On Adaptive Attacks to Adversarial Example Defenses." In *Advances in Neural Information Processing Systems*, volume 33, pp. 1633–1645, 2020.

[Tie08] Tijmen Tieleman. “Training restricted Boltzmann machines using approximations to the likelihood gradient.” In *Proceedings of the 25th international conference on Machine learning*, pp. 1064–1071, 2008.

[TL20] Mingxing Tan and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.”, 2020.

[WRK20] Eric Wong, Leslie Rice, and J. Zico Kolter. “Fast is better than free: Revisiting adversarial training.” In *International Conference on Learning Representations*, 2020.

[WXZ18] Yingnian Wu, Jianwen Xie, and Song-Chun Zhu. “Sparse and Deep Generalizations of the FRAME model.” *Annals of Mathematical Sciences and Applications*, 2018.

[XEQ18] Weilin Xu, David Evans, and Yanjun Qi. “Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks.” In *25th Annual Network and Distributed System Security Symposium, NDSS*, 2018.

[XKK21] Zhisheng Xiao, Karsten Kreis, Jan Kautz, and Arash Vahdat. “{VAEBM}: A Symbiosis between Variational Autoencoders and Energy-based Models.” In *International Conference on Learning Representations*, 2021.

[XLG18] Jianwen Xie, Yang Lu, Ruiqi Gao, and Ying Nian Wu. “Cooperative learning of energy-based model and latent variable model via mcmc teaching.” In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[XLZ16] Jianwen Xie, Yang Lu, Song-Chun Zhu, and Yingnian Wu. “A theory of generative convnet.” In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 2635–2644, 2016.

[XTG21] Cihang Xie, Mingxing Tan, Boqing Gong, Alan Yuille, and Quoc V. Le. “Smooth Adversarial Training.”, 2021.

[XWZ18] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. “Mitigating Adversarial Effects Through Randomization.” In *International Conference on Learning Representations*, 2018.

[XZF21] Jianwen Xie, Zilong Zheng, Xiaolin Fang, Song-Chun Zhu, and Ying Nian Wu. “Cooperative Training of Fast Thinking Initializer and Slow Thinking Solver for Conditional Learning.”, 2021.

[XZG18] Jianwen Xie, Zilong Zheng, Ruiqi Gao, Wenguan Wang, Song-Chun Zhu, and Ying Nian Wu. “Learning Descriptor Networks for 3D Shape Synthesis and Analysis.” In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 8629–8638, 2018.

[XZW17] Jianwen Xie, Song-Chun Zhu, and Ying Nian Wu. “Synthesizing Dynamic Patterns by Spatial-Temporal Generative ConvNet.” In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 1061–1069, 2017.

[YHL21] Jongmin Yoon, Sung Ju Hwang, and Juho Lee. "Adversarial purification with Score-based generative models." *arXiv preprint arXiv:2106.06041*, 2021.

[YSS20] Lantao Yu, Yang Song, Jiaming Song, and Stefano Ermon. "Training deep energy-based models with f-divergence minimization." In *International Conference on Machine Learning*, pp. 10957–10967. PMLR, 2020.

[YZK19] Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. "ME-Net: Towards Effective Adversarial Robustness with Matrix Estimation." In *Proceedings of the 36th International Conference on Machine Learning*, pp. 7025–7034, 2019.

[ZK16] Sergey Zagoruyko and Nikos Komodakis. "Wide Residual Networks." *arXiv preprint arXiv:1605.07146*, 2016.

[ZWM98] Song-Chun Zhu, Ying Nian Wu, and David Mumford. "Filters, Random Fields and Maximum Entropy (FRAME): Towards a Unified Theory for Texture Modeling." *International Journal of Computer Vision*, **27**(2):107–126, 1998.

[ZYJ19] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. "Theoretically principled trade-off between robustness and accuracy." In *Proceedings of the 36th International Conference on Machine Learning*, pp. 7472–7482, 2019.