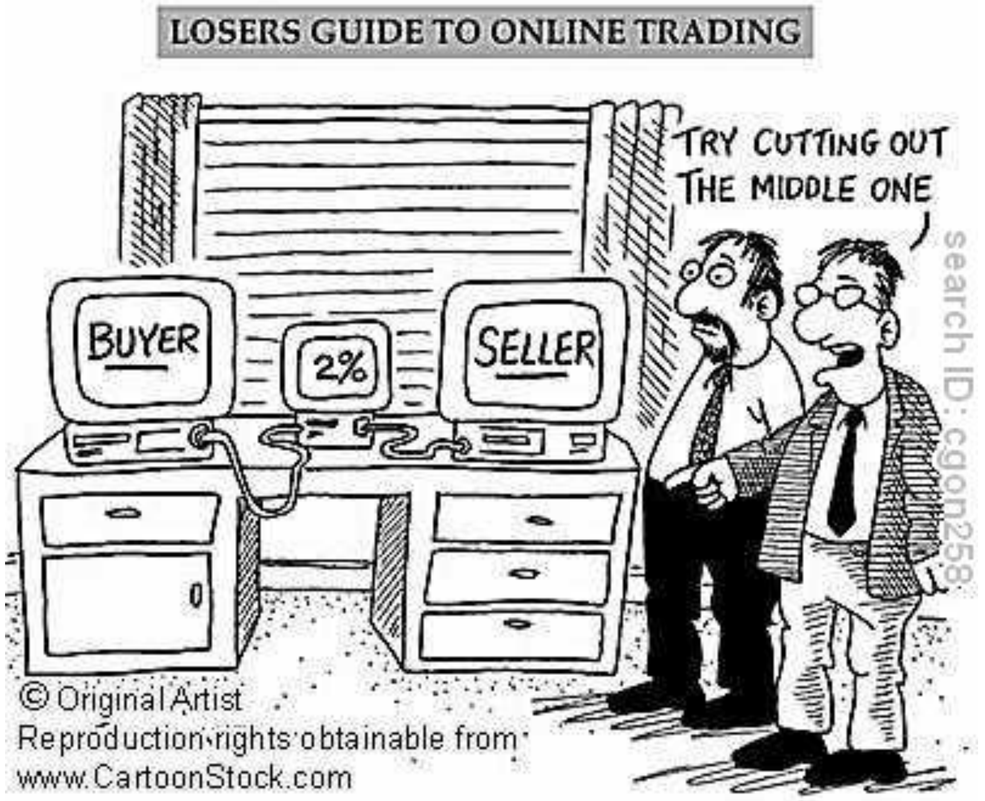
Cutting out the Middle-Man: Training and Evaluating Energy-Based Models without Sampling

Will Grathwohl, Kuan-Chieh Wang, Jorn-Henrik Jacobsen, David Duvenaud, Richard Zemel

https://arxiv.org/abs/2002.05616

Paper discussion



Alex Fedorov

May 29, 2020

Agenda

- Introduction
- Stein Discrepancy
- Model Evaluation
- Model Training

Introduction

Energy-Based Models

Density

$$p(x) = \frac{\exp(-E(x))}{Z}$$

density

$E : \mathbb{R}^D \to \mathbb{R}$

energy function

$$Z = \int_x \exp(-E(x))$$

normalizing constant

Normalizing Constant

Main Challenge

$$Z = \int_{x} \exp(-E(x))$$

1. Can not compute efficiently
2. Can not compute likelihoods under our model
3. Making training and evaluation difficult

How to train EBM?

Sampling techniques.

- MCMC
  - To estimate likelihood (for evaluation)
  - To estimate its gradients (for training)

Surrogate objectives.

- Score matching (Hyvärinen, 2005))
- NCE (Gutmann & Hyvärinen, 2010)

when MC is seeded from D_{tr}

$$\frac{\partial \log p_\theta(x)}{\partial \theta} = E_{x' \sim p_\theta(x)}\left[\frac{\partial E_\theta(x')}{\partial \theta}\right] - \frac{\partial E_\theta(x)}{\partial \theta}$$

Alternative models.

- VAE (Kingma & Welling, 2013)
  - Posterior collapse (Lucas et al., 2019)
- NF (Rezende & Mohamed, 2015)
  - inability of NFs to model distributions with certain topological structures (Falorsi et al., 2018)
- Offer more easily scalable training, evaluation, and sampling
- More restrictive model parameterization

Recent advances

- Noise-Contrastive approaches (Gao et al., 2019)
- Score Matching (Song & Ermon, 2019)
- MCMC-based training (Nijkamp et al., 2019a)

protein structure prediction (Ingraham et al., 2019; Du et al., 2020)

adversarial robustness, calibration, out-of- distribution detection (Grathwohl et al., 2019; Du & Mor- datch, 2019), and semi-supervised learning (Song & Ou, 2018).

Nijkamp, E., Zhu, S.-C., and Wu, Y. N. On learning nonconvergent short-run mcmc toward energy-based model. arXiv preprint arXiv:1904.09770, 2019b.

However

"Biased by the sampler"

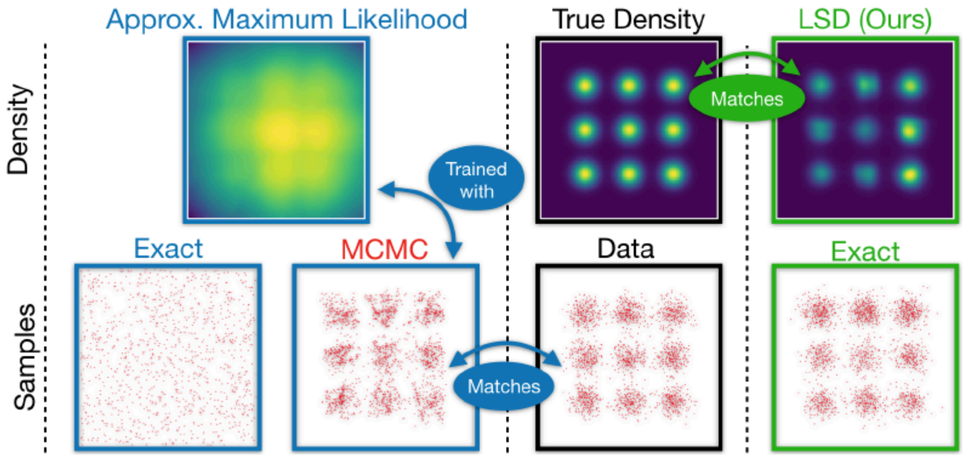


Figure 1. Density models trained with approximate MCMC samplers can fail to match the data density while still generating high-quality samples. Samples from approximate MCMC samplers follow a *different* distribution than the density they are applied to. It is this induced distribution which is trained to match the data. In contrast, our approach **LSD** directly matches the model density to the data density without reliance on a sampler.

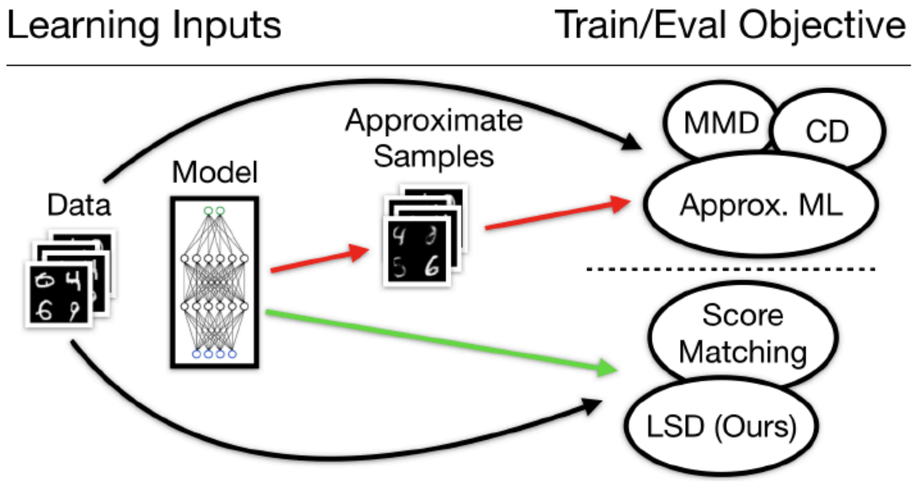


Figure 2. Cutting out the "middle-man" of approximate sampling can lead to simpler training and evaluation that is tied directly to the quality of our model and is not obfuscated by the parameters of an MCMC sampler.

Also

- Little insight into the quality of the models we are learning
  - Solution 1: to indirectly measure quality of the model by evaluating performance on downstream discriminative tasks as advocated for in Theis et al. (2015), and recently applied to EBMs by Grathwohl et al. (2019)
  - Solution 2: to use metrics that rely on samples generated by expensive MCMC algorithms (Nijkamp et al., 2019a; Song & Ermon, 2019; Du & Mordatch, 2019).

Stein Discrepancy

Stein's Identity (Stein et al., 1972)

$$\mathbb{E}_{p(x)}\left[\nabla_x \log q(x)^T f(x) + \mathrm{Tr}\left(\nabla_x f(x)\right)\right] = 0 \qquad (\star)$$

don't need normalizing constant

$f(x) : \mathbb{R}^D \to \mathbb{R}^D$ is any s.t. $\lim_{\|x\| \to \infty} p(x) f(x) = 0$

↑ critic

$(\star) = 0$ iff $p = q$

$$S(p, q) = \sup_{f \in \mathcal{F}} \mathbb{E}_{p(x)}\left[\nabla_x \log q(x)^T f(x) + \mathrm{Tr}\left(\nabla_x f(x)\right)\right]$$

↑ Stein Discrepancy

Kernelized Stein Discrepancy (KSD)

if $\mathcal{F}$ is a ball in RKHS

$$\Rightarrow \quad KSD(p, q) = \mathbb{E}_{x, x' \sim p(x)} \Big[\nabla_x \log q(x)^T k(x, x') \nabla_{x'} \log q(x') + \nabla_x \log q(x)^T \nabla_{x'} k(x, x') + \nabla_x k(x, x') \nabla_{x'} \log q(x') + \mathrm{Tr}\left(\nabla_{x, x'} k(x, x') \right) \Big]$$

Learning the Stein Discrepancy

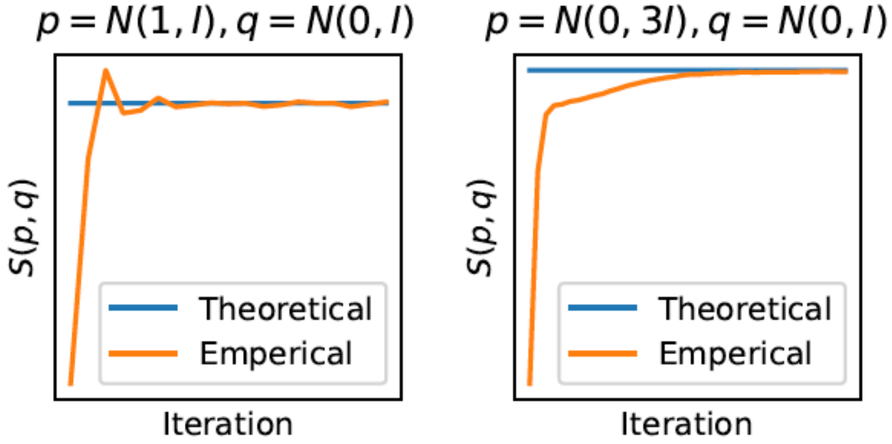
- Methods based on distances and kernels quickly degrade in performance as dimension increases (Ramdas et al., 2015).
- The power of tests based on these kernel methods is closely tied to their asymptotic run-time;
  - the quadratic time test of Liu et al. (2016) significantly outperforms their linear-time variant, which prevents using the best-performing approach on large datasets.

$\Longrightarrow$ Using NN as f_ϕ and Stein Discrepancy

$$\text{Learned} \rightarrow \boxed{\text{LSD}}(f_\phi, p, q) = \mathbb{E}_{p(x)}\left[\nabla_x \log q(x)^T f_\phi(x) + \text{Tr}(\nabla_x f(x))\right]$$

Choosing F

- To estimate a Stein Discrepancy, we must optimize the critic over a bounded space of functions F.
- Unconstrained neural networks do not fall into this category
  - This can be accomplished in many ways such as with **weight-clipping**, or **spectral normalization** (Miyato et al., 2018).



$$\Rightarrow \mathcal{F} = \{ f : \mathbb{E}_{p(x)}[f(x)^T f(x)] < \infty \}$$

$$\Rightarrow L_2 \text{ regularizer} \Rightarrow R_\lambda(f_\phi) = \lambda \mathbb{E}_{p(x)}[f_\phi(x)^T f_\phi(x)]$$

$$\Rightarrow LSD(f_\phi, p, q) - R_\lambda(f_\phi) \Rightarrow \text{Hu et al. (2018) optimal critic:}$$

$$f_\phi(x) = \frac{1}{2\lambda}(\nabla_x \log q(x) - \nabla_x \log p(x))$$

Efficient Estimation

$\mathrm{Tr}\left(\nabla_x f(x)\right)$ is expensive !!!

$O(D)$ vector-Jacobian products ("backward passes")

$\Rightarrow$ Hutchinson's estimator (Hutchinson, 1990)

$$\mathrm{Tr}\left(\nabla_x f(x)\right) = \mathbb{E}_{N(\varepsilon|0,1)}\left[\varepsilon^T \nabla_x f(x) \varepsilon\right]$$ ← single sample MC estimator

$$\Rightarrow \underbrace{\mathrm{LSD}E}_{\text{efficient}}\left(f_\phi, p, q\right) = \mathbb{E}_{p(x)N(\varepsilon|0,1)}\left[\nabla_x \log q(x)^T f_\phi(x) + \varepsilon^T \nabla_x f_\phi(x) \varepsilon\right]$$

Model Evaluation

Model Evaluation with LSD

Model Comparison

model $q_1(x)$ model $q_2(x)$

$\{x_i\}_{i=1}^n \sim p(x)$

Choose model with max

$\mu - 6$

Algorithm 1 LSD Model Comparison

Input: Critic architecture f_ϕ, models q_1, q_2, data $\mathbf{x} = \{x_i\}_{i=1}^n$, $\mathcal{L}_2$ regularization hyperparameter λ

Output: Estimated Stein Discrepancies $\mathbf{S}(p, q_1)$, $\mathbf{S}(p, q_2)$

Split $\mathbf{x}$ into $\mathbf{x}_{\text{train}}$, $\mathbf{x}_{\text{val}}$, and $\mathbf{x}_{\text{test}}$

for model q_j in (q_1, q_2) **do**

find $\phi_j = \text{argmax}_\phi \text{LSDE}(f_\phi, \mathbf{x}_{\text{train}}, q_j) - \mathcal{R}_\lambda(f_\phi)$

(using $\mathbf{x}_{\text{val}}$ for model selection)

$s_j = \text{LSD}(f_{\phi_j}, \mathbf{x}_{\text{test}}, q_j)$

end for

Rank models in increasing order of s_j.

Model Evaluation with LSD

Goodness-Of-Fit Testing

$H_0: p = q \qquad H_1: p \neq q$

$$s_f^q(x) = \nabla_x \log q(x)^T f(x) + \mathrm{Tr}\left(\nabla_x f(x)\right)$$

$H_0: E_{p(x)}\left[s_f^q(x)\right] = 0$

$H_1: E_{p(x)}\left[s_f^q(x)\right] \neq 0$

one sample location test

Test Power

$$P\left(f_\phi, p, q\right) = \frac{E_{p(x)}\left[s_{f_\phi}^q(x)\right]}{\sigma_{p(x)}\left[s_{f_\phi}^q(x)\right]}$$

Algorithm 2 LSD Goodness of Fit Test

Input: Critic architecture f_ϕ, model q, $\mathbf{x} = \{x_i\}_{i=1}^n$, $\mathcal{L}_2$ regularization hyperparameter λ, confidence α

Output: Decision whether $p = q$

Split $\mathbf{x}$ into $\mathbf{x}_{\text{train}}$, $\mathbf{x}_{\text{val}}$, and $\mathbf{x}_{\text{test}}$

maximize $\phi = \text{argmax}_\phi \mathcal{P}(f_\phi, \mathbf{x}_{\text{train}}, q)$ (using $\mathbf{x}_{\text{val}}$ for model selection)

Compute $t = \sqrt{n}\frac{\mathbf{E}[s(\mathbf{x}_{\text{test}})]}{\sigma[s(\mathbf{x}_{\text{test}})]}$

If $t > \Phi(1-\alpha)$ reject H_0, else accept H_0

Model Evaluation
Hypothesis Testing

Gaussian-Bernoulli Restricted Boltzmann Machine (Cho et al., 2013).

- the quadratic-time Kernelized Stein Discrepancy (KSD),
- its linear-time variant (LKSD) (Liu et al., 2016),
- the lineartime Finite Set Stein Discrepancy (FSSD) (Jitkrittum et al., 2017).

$$p(x,h) = \frac{1}{Z}\exp\left(\frac{1}{2}x^T Bh + b^T x + c^T h - \frac{1}{2}||x||^2\right)$$

$$\nabla_x \log p(x) = b - x + B \cdot \tanh\left(B^T x + c\right)$$

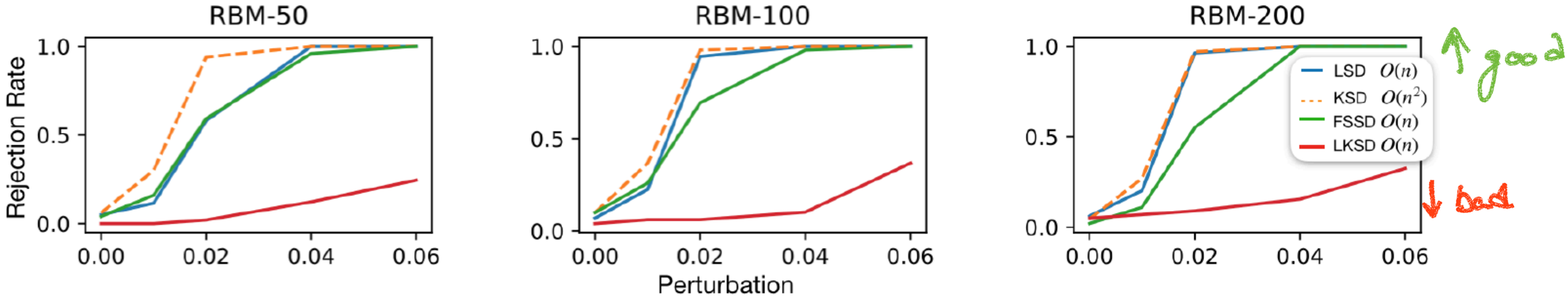


Figure 5. Hypothesis testing results. Test confidence 0.05. Perturbed RBMs of increasing data dimension. Perturbation magnitude on the x-axis, rejection rate on the y-axis. Number of datapoints $n = 1000$. Ideal behavior is a 5% rejection when perturbation is 0 and close to 100% rejection otherwise. In high dimensions our linear-time **LSD** matches the performance of the quadratic-time KSD.

RBM Evaluation

Ability to rank and evaluate the fit of unnormalized models on fixed test data.

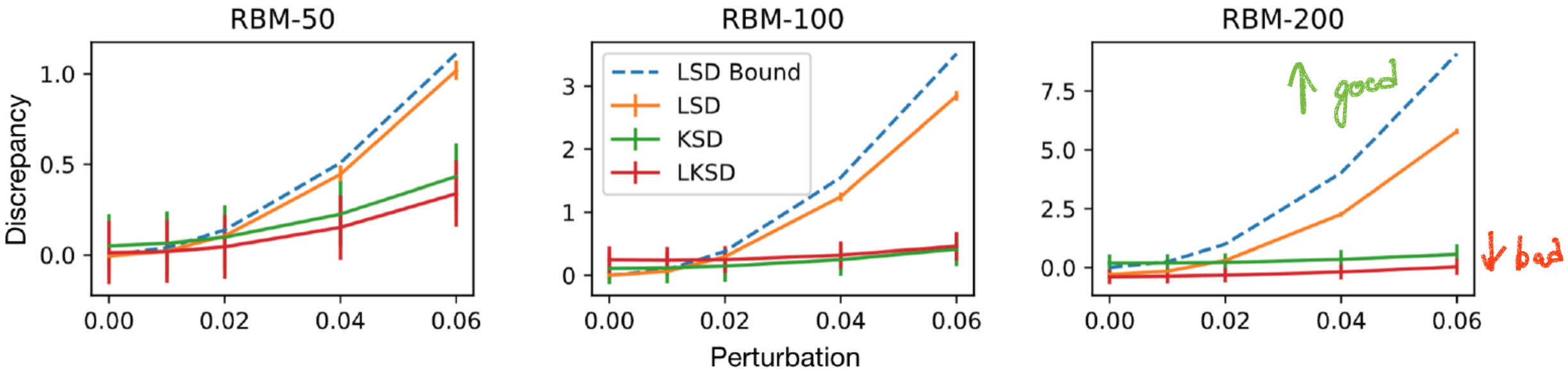


Figure 6. Model evaluation results for Gaussian-Bernoulli RBMs. Predicted discrepancy should increase as perturbation increases. **LSD** reliably increases, approaching the theoretical ground-truth value. **LSD** provides much more certain results than both linear-time and quadratic-time kernel-based approaches.

Model Evaluation

Normalizing Flow Evaluation

Architecture from ↓

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In Advances in Neural Information Processing Systems, pp. 10215–10224, 2018.

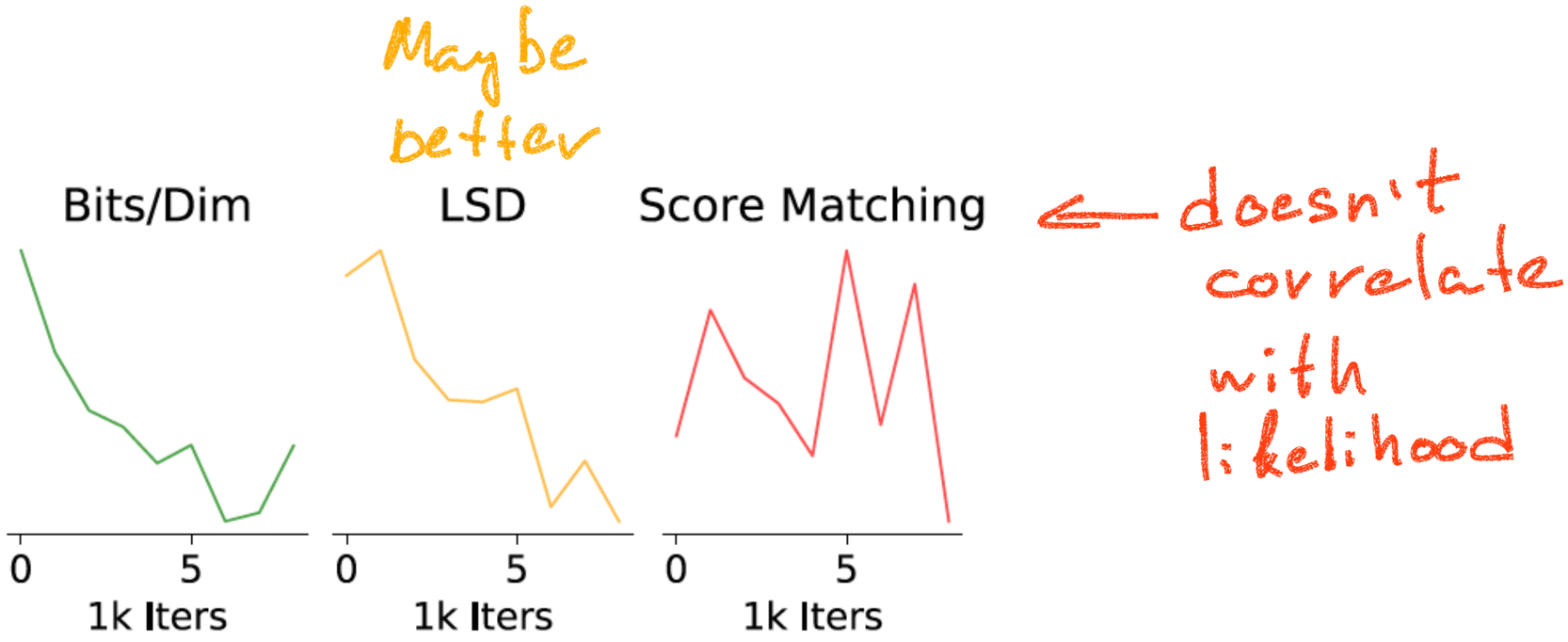


Figure 7. Flow model evaluation. Left to right: Bits/Dim, LSD, Score Matching. Y-axis is removed because scales are not comparable.

Model Training

Training Unnormalized Models with LSD

Algorithm 3 LSD Training

Input: Critic architecture f_ϕ, models q_θ, data $\mathbf{x} = \{x_i\}_{i=1}^n \sim p(x)$, $\mathcal{L}_2$ regularization hyperparameter λ, training iterations T, critic training iterations C

Output: Parameters θ such that $q_\theta \approx p$

for T iterations **do**
  for C iterations **do**
    Sample mini-batch $\mathbf{x}'$
    Update ϕ with $\nabla_\phi \left(\text{LSDE}(f_\phi, \mathbf{x}', q_\theta) - \mathcal{R}_\lambda(f_\phi)\right)$
  end for
  Sample mini-batch $\mathbf{x}'$
  Update θ with $-\nabla_\theta \text{LSD}(f_\phi, \mathbf{x}', q_\theta)$
end for
Return resulting model q_θ

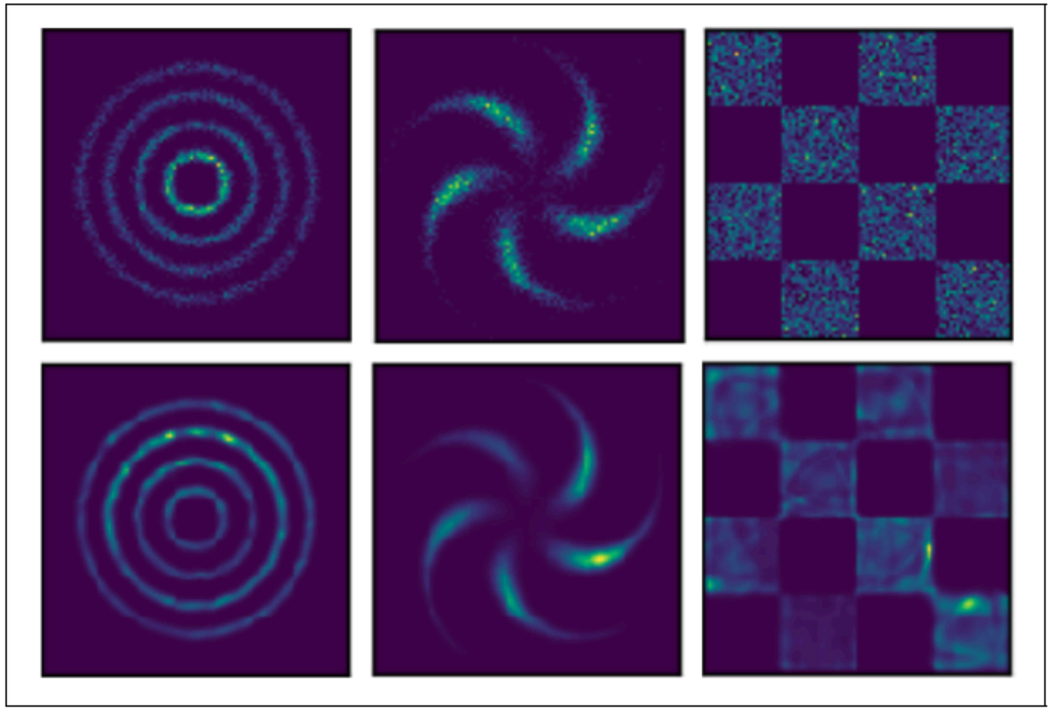


Figure 4. Density models trained using LSD. Top: Data. Bottom: Learned densities.

Model Training

Linear ICA

$$\text{ICA} \quad z \sim \text{Laplace}(0,1)$$
$$x = Wz$$
$$\log p(x;W) = \log p_z(W^{-1}x) - \log|W|$$

- Noise-Contrastive Estimation (NCE) (Gutmann & Hyv¨arinen, 2010)
- Conditional Noise-Constrastive Estimation (CNCE) (Ceylan & Gutmann, 2018)
- Score Matching (SM) (Hyv¨arinen, 2005)

Method	Data Dimension				
	10	20	30	40	50
Max. Likelihood	-10.98	-18.48	-21.49	-23.43	-25.53
LSD (Ours)	−10.95	**−18.37**	**−21.23**	**−25.14**	**−25.36**
Score Matching	-11.13	-27.20	-21.48	NaN	NaN
NCE	**−10.92**	-22.52	-30.33	-55.53	-73.62
CNCE	-11.00	-18.77	-24.47	-37.64	-36.31

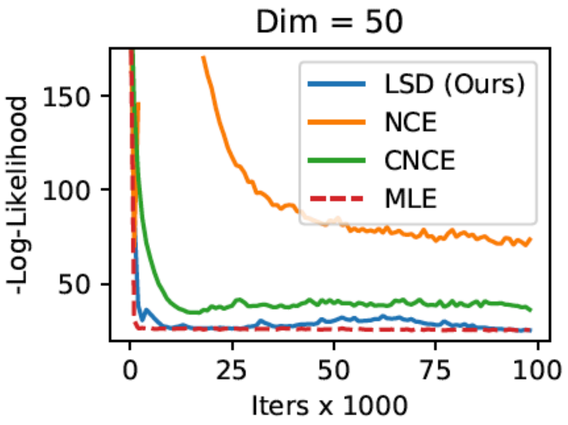


Figure 8. Linear ICA training results. Left: Table comparing test set log-likelihoods of linear ICA models with various training methods. **LSD** closely tracks the performance maximum likelihood training while other unnormalized methods fall behind or diverge. Right: Learning curves for 50-dimensional ICA. While slower to converge than maximum likelihood, **LSD** cleanly converges to the same result.

Model Training

Preliminary Image Modeling Results

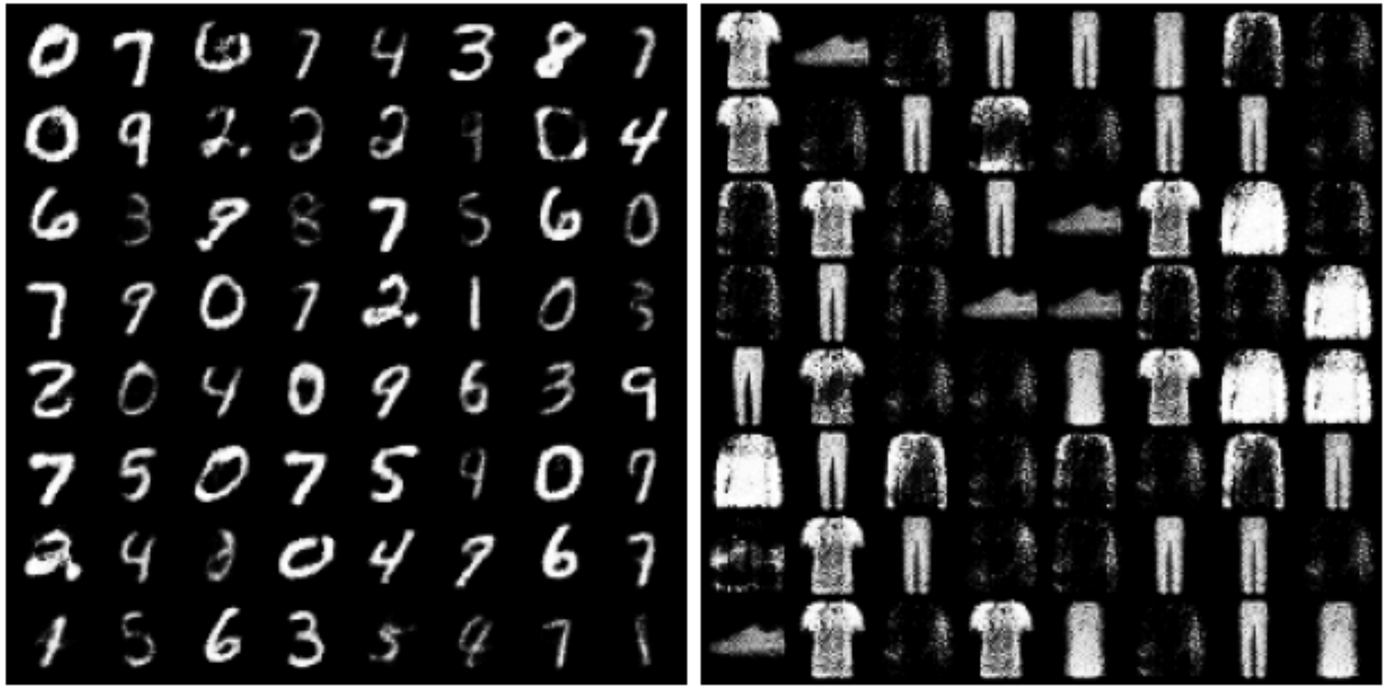


Figure 9. Samples from deep EBMs trained with LSD. Left: MNIST, Right: FashionMNIST.

Thank you