Introduction to Score-matching

Justin T Chiu

July 6, 2023

Goals

1. What is an energy-based model and why are they hard to train?
2. What is score-matching, and how can it be used to train an EBM?
3. How does score-matching relate to diffusion models?

Energy-Based Models (EBM)

Problem setup: Density estimation

- Observations from true model $x \sim p^*(x)$
- Ideally: Learn a model $p(x)$ that's close to $p^*(x)$
  - Capture uncertainty / variability over x
- Participation: Give examples of an x we model, and how $p(x)$ is parameterized
  - Ex: Language modeling uses Transformers for $p(x) = \prod_t p(x_t | x_{<t})$

Running example: Image generation

- "Solved": Finite-class density estimation
  - Softmax assigns a score to each $E(x)$ then normalizes

$$softmax(x) = \frac{\exp(E(x))}{\sum_x \exp(E(x))}$$

- Image generation
  - Every change in a single pixel is a new class
  - Size: 1024 x 1024, each pixel has 256 * 3 values

Image generation models

- Autoregressive: Break down generation from left-to-right

$$p(x) = \prod_t p(x_{ij} | x_{<i,j}, x_{\bullet,<j})$$

- Latent variable model: Specify break down more flexibly

$$p(x) = \sum_z p(x|z)p(z)$$

- Energy-based model: Don't force breakdown of decision process

$$p(x) = \frac{E(x)}{\int_x E(x)}$$

EBM drawing

- Example:

$$E(x) = \sum_i E_i(x)$$

What is an EBM?

- Globally normalized over images x

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

- Computation of the partition function Z is hard
  - Integrate $E(x)$ over all possible images
- Goal of training: maximize likelihood (minimize KL div)
  - Need to compute $p(x)$ and therefore Z
  - Next: How to avoid computing partition function Z

Score-matching: Training an EBM

KL divergence to Fisher divergence

- Standard: Minimize KL divergence

$$E_{p^*(x)} \log \frac{p^*(x)}{p(x)} = E_{p^*(x)} \log p^*(x) - E_{p^*(x)} \log p(x)$$

- Issue: Can't compute $p(x)$ because of Z
- Instead: Give up on exact likelihood computation. Use Fisher divergence

Approximation lemma

- Two continuous functions are equal iff their derivatives are equal

Fisher divergence intuition

- If two density fns are equal, have the same Stein score $s(x) = \nabla_x \log p(x)$
- Can use the Stein score to get good samples / find likely x
  - Langevin dynamics: follow score + noise
- Lose ability to compute likelihoods, can only use score model for sampling

Minimize Fisher divergence = Score matching

- Minimize Fisher divergence to avoid computing Z

$$E_{p^*(x)} \left\| \nabla_x \log \frac{p^*(x)}{p(x)} \right\|_2^2 = E_{p^*(x)} \left\| \nabla_x \log p^*(x) - \nabla_x \log p(x) \right\|_2^2$$

- Notation: Introduce Stein score $s(x) = \nabla_x \log p(x)$

$$E_{p^*(x)} \left\| \nabla_x \log p^*(x) - \nabla_x \log p(x) \right\|_2^2 = E_{p^*(x)} \left\| \nabla_x \log p^*(x) - s(x) \right\|_2^2$$

- Parameterize $s(x)$ directly instead of $p(x)$, avoid computing Z

Issues in training an EBM

$$E_{p^*(x)} \|\nabla_x \log p^*(x) - s(x)\|_2^2$$

1) Solved: Cant compute $p(x)$ b/c of Z => model Stein score $s(x) = \nabla_x \log p(x)$

2) Unknown p^*: Dont know $p^*(x)$ or its score

3) Covariate shift: $E_{p^*(x)}$ is problematic because of covariate shift

Avoiding p^*: Implicit score matching

- Can rewrite the explicit score matching objective to avoid p^*

$$E_{p^*(x)}\left[\|\nabla_x \log p^*(x) - s(x)\|_2^2\right] \approx E_{p^*(x)}\left[\frac{1}{2}\|s(x)\|_2^2 + tr(\nabla_x s(x))\right]$$

- Second term is nasty: $s(x) \in R^d$, $\nabla_x s(x) \in R^{d\times d}$
- Solution: Use Hutchinson's trace estimator

$$E_{p^*(x)}\left[\frac{1}{2}\|s(x)\|_2^2 + tr(\nabla_x s(x))\right] = E_{v\sim N(0,I_d)}E_{p^*(x)}\left[\frac{1}{2}\|s(x)\|_2^2 + v^T\nabla_x s(x))v\right]$$

- Easy to implement with pytorch

Covariate shift

- Sample via Langevin dynamics := Start with random point and follow score + noise
  - Score is trained on examples drawn from $p^*(x)$
  - Score is bad on regions of low $p^*(x)$, eg random points
  - Slow mixing and bad samples

Solution to cov shift

- Solution: sample perturbed $x \sim p^*(x)$ with multiple noise scales $\{\sigma_i\}$
  - Interpretation: Data augmention + smooth out samples
  - Need to have score model condition on noise $s(x; \sigma_i)$

Summary

- Intractable partition function => Model score
  - Lose ability to compute likelihoods, can only use score model for sampling
  - Sample via Langevin dynamics (follow grad+noise)
- Don't know data score: Rewrite objective to avoid $\nabla_x p^*(x)$
  - Results in some nasty expressions => Estimate with Hutchinson trace estimator
- Add multiple noise scales to help learning score at random points

Connection to diffusion models

Diffusion models

- Hierarchical VAE perspective: forward / reverse process vs noised marginals + score model

- SDE: continuous-time extension of score matching (time = the noise scale)

Credits

- Ayan Das' blog post
- Lyu 2009
- Vincent 2011
- Song 2019