Deep Learning

Lecture 5: Energy-based models

Chris G. Willcocks

Durham University

Lecture overview

1. **Manifolds**
2. **Energy-based models**
   - definition
   - GANs as energy-based models
   - clustering as an energy-based model
   - softmax and softmin
   - exact likelihood
3. **Contrastive-divergence approaches**
4. **Boltzmann machines**
   - definition
   - restricted and deep Boltzmann machines
5. **Score-based approaches**
   - Langevin dynamics
   - score-matching and denoising diffusion

Manifold definition

Definition: manifold

A manifold is a topological space that locally resembles Euclidean space

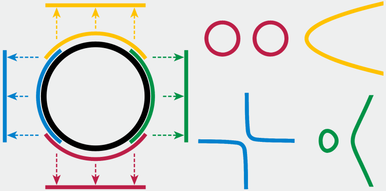
- topological manifold
- differentiable manifold
- Riemannian manifold

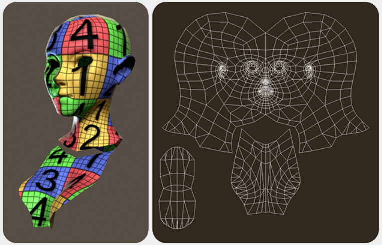
Definition: embedding

An embedding is a function ϕ that maps a manifold $\mathcal{M}$ to a new manifold $\mathcal{N}$ in an injective way that preserves its structure:

$$\phi : \mathcal{M} \rightarrow \mathcal{N}$$

Example: manifolds





Manifold charts by KSmrq, distributed under CC. UV mesh example by Antony Ward.

Energy-based models definition

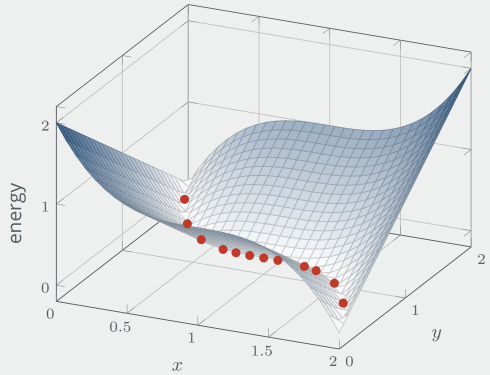
Definition: energy-based models

These are just any function that is happy when you input something that looks like data, and is not happy when you input something that doesn't look like data.

$$E(\mathbf{x}) = 0 \ ✓$$
$$E(\tilde{\mathbf{x}}) > 0 \ ✗$$

This generic definition fits a large majority of machine learning models. For example $\mathcal{L}(E(\mathbf{x}), \mathbf{y})$ (a classifier)

Energy increases off manifold



Energy-based models GANs as energy-based models

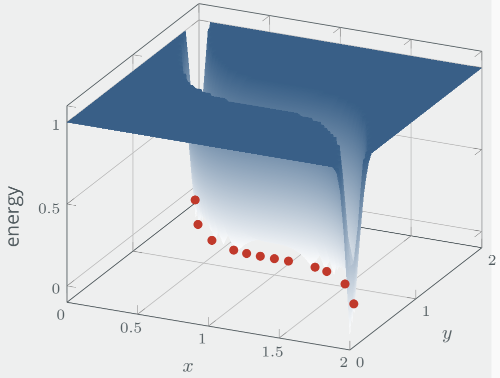
Definition: energy-based models

GANs are also energy models. The generator G generates samples off the manifold, then the descriminator D says these should be one everywhere, whereas it says real samples should be zero everywhere.

The generator also has to get good at sampling points on the data manifold. So it has to learn to generate points in the valley regions.

Is this smooth? What does a 1-Lipschitz discriminator do to the energy landscape?

GAN energy



Energy-based models clustering as an energy-based model

Definition: clustering algorithm

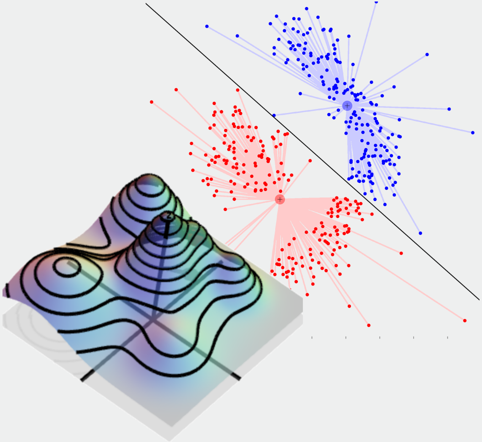
A cluster is a **connected-component** of a **level-set** of the **unknown PDF** over our data observations.

Traditionally:

- We don't know the PDF (the energy landscape)
- We don't necessarily know the level set
  - although 0.5 is appropriate for BCE
- This can be expensive (deep learning)

Click to watch a video that visually explains from the definition ▶

Example: clustering by its definition



Energy-based models softmax and softmin

Definition: softmax and softmin

Softmax and softmin functions rescale elements to be in the range [0, 1] and such that they sum to 1. So they create a probability mass function, e.g.:

$$\begin{bmatrix} 1.3 \\ 7.2 \\ 2.4 \\ 0.5 \\ 1.1 \end{bmatrix} \rightarrow \frac{e^{\mathbf{z}_i}}{\sum_{j=1}^{K} e^{\mathbf{z}_j}} \rightarrow \begin{bmatrix} 0.0027 \\ 0.9858 \\ 0.0081 \\ 0.0012 \\ 0.0022 \end{bmatrix}$$

Softmax functions are widely used (not just for EBMs) where a distribution is needed, such as the last layer of a classifier.

Energy-based models exact likelihood

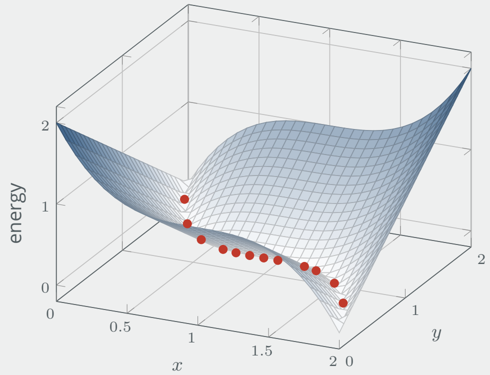
Challenges: energy-based models

EBMs are based on the observation that any probability density function $p(\mathbf{x})$ for $\mathbf{x} \in \mathbb{R}^n$ can be expressed as:

$$p(\mathbf{x}) = \frac{e^{-E(\mathbf{x})}}{\int_{\tilde{\mathbf{x}} \in \mathcal{X}} e^{-E(\tilde{\mathbf{x}})}},$$

where $E(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ is the energy function. However computation of the integral is intractable [1] for most models.

Energy increases off manifold



Contrastive-divergence approaches definition

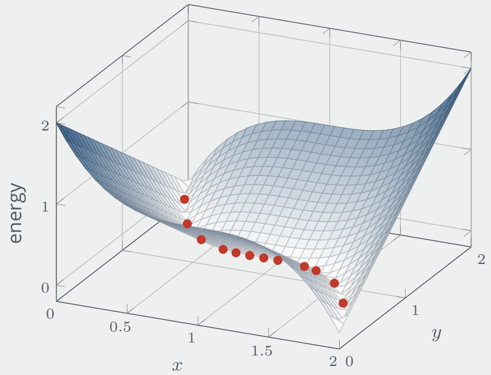
Definition: contrastive-divergence

The gradient of the negative log-likelihood loss $\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}\sim p_d}[-\ln p_\theta(\mathbf{x})]$ has been shown to demonstrate the following property:

$$\nabla_\theta \mathcal{L} = \mathbb{E}_{\mathbf{x}^+\sim p_d}[\nabla_\theta E_\theta(\mathbf{x}^+)] - \mathbb{E}_{\mathbf{x}^-\sim p_\theta}[\nabla_\theta E_\theta(\mathbf{x}^-)]$$

where $\mathbf{x}^- \sim p_\theta$ is a sample from the energy model found through a Monte Carlo Markov Chain (MCMC) generating procedure.

Energy increases off manifold



Boltzmann machines definition

Definition: Boltzmann machine

Boltzmann machines [2] are one of the earliest neural networks for modeling binary data. They can associate the probability of the visible vectors $\mathbf{v}$ using finite summations:

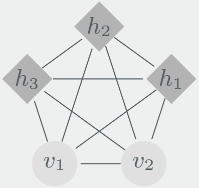
$$p_\theta(\mathbf{v}) = \frac{\sum_{\mathbf{h}} e^{-\beta E_\theta(\mathbf{v},\mathbf{h})}}{\sum_{\tilde{\mathbf{v}}} \sum_{\mathbf{h}} e^{-\beta E_\theta(\tilde{\mathbf{v}},\mathbf{h})}}$$

They are typically trained via negative log-likelihood through contrastive divergence, where the weights are updated:

$$\sum_{\mathbf{x}\in\mathcal{X}} \frac{\partial \ln p(\mathbf{x})}{\partial w_{i,j}} = \mathbb{E}_{p_{\mathsf{d}}}[\mathbf{v}\mathbf{h}^T] - \mathbb{E}_{p_{\mathsf{model}}}[\mathbf{v}\mathbf{h}^T]$$

Example: Boltzmann machine

They are an energy model which just have visible layers $v_1, v_2, ..., v_n$ (inputs) and hidden layers $h_1, h_2, ..., h_n$ (no outputs):



This example Boltzmann Machine has 2 visible units and 3 hidden units.

Boltzmann machines restricted and deep Boltzmann machines

Definition: RBMs and DBMs

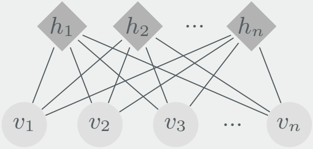
Restricted Boltzmann Machines (RBMs) and Deep Boltzmann Machines (DBMs) are Boltzmann machines with a more restricted (bipartite) graph structure [3]. DBMs have additional hidden layers.

That means that the visible units conditional on the hidden units become independant, which makes training these straightforward in practice.

Link to Colab ↗ **Good YouTube talk** ▶

Example: RBM

RBMs have a restricted architecture architecture so that there are no connections between hidden units:



DBMs are like the above, but with multiple hidden layers between.

Score-based approaches Langevin dynamics

Definition: score-based GMs

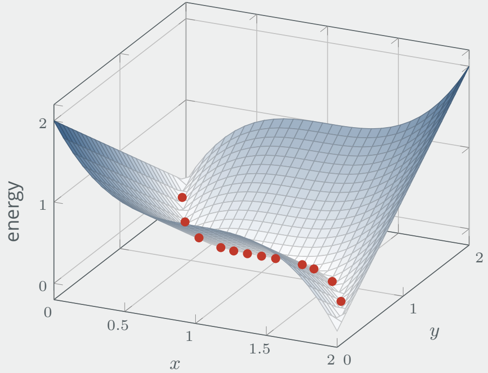
Score-based generative modeling [4] also eliminates the intractable second term (sampling from the model). For the PDF $p(\mathbf{x})$ the score function is:

$$s(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

When the score function is known, we can use Langevin dynamics to sample the model. Given a step size $\alpha > 0$, a total number of iterations T, and an initial sample $x0$ from any prior distribution $\pi(\mathbf{x})$, Langevin dynamics iteratively updates:

$$\mathbf{x}_t \leftarrow \mathbf{x}_{t-1} + \alpha \nabla_{\mathbf{x}} \log p(\mathbf{x}_{t-1}) + \sqrt{2\alpha}\, \mathbf{z}_t$$

Energy increases off manifold



Score-based approaches score-matching and denoising diffusion

Definition: score-matching

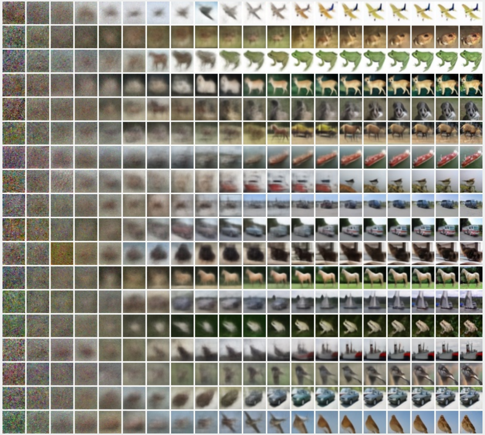
Score matching minimises the Fisher divergence between p_d and p_θ:

$$\mathcal{L} = \frac{1}{2}\mathbb{E}_{p_d(\mathbf{x})}[\,\|s_\theta(\mathbf{x}) - s_{\text{data}}(\mathbf{x})\|_2^2],$$

however, the score function is inaccurate in regions without training data. Instead, perturb the data with noise [5] giving corrupted data samples $q(\tilde{\mathbf{x}}|\mathbf{x})$. In particular, when $q = \mathcal{L}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2 \boldsymbol{I})$, we get:

$$\mathcal{L} = \frac{1}{2}\mathbb{E}_{p_d(\mathbf{x})}\mathbb{E}_{\tilde{\mathbf{x}}\sim\mathcal{N}(\mathbf{x},\sigma^2\boldsymbol{I})}\left[\left\|s_\theta(\tilde{\mathbf{x}}) + \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2}\right\|_2^2\right].$$

Example: CIFAR10 samples from [6]



References I

[1] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. “A tutorial on energy-based learning”. In: Predicting structured data 1.0 (2006).

[2] Geoffrey E Hinton and Terrence J Sejnowski. “Optimal perceptual inference”. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. Vol. 448. Citeseer. 1983.

[3] Geoffrey E Hinton. “Training products of experts by minimizing contrastive divergence”. In: Neural computation 14.8 (2002), pp. 1771–1800.

[4] Yang Song and Stefano Ermon. “Improved techniques for training score-based generative models”. In: arXiv preprint arXiv:2006.09011 (2020).

[5] Pascal Vincent. “A connection between score matching and denoising autoencoders”. In: Neural computation 23.7 (2011), pp. 1661–1674.

[6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: arXiv preprint arXiv:2006.11239 (2020).