Energy-based models (EBM), part 1

Aapo Hyvärinen

for the NNDL course, 11 April 2025 (second half)

- Reminder: estimation of statistical models
- Definition of energy-based models (= unnormalized statistical models)
- How to estimate EBMs? Why maximum likelihood is difficult.
- Score matching, one solution to EBM estimation
- Denoising score matching, a computationally simpler alternative

Literature: Murphy's book Chapter 24

Reminder: Estimation of (ordinary) statistical models

- Assume a random variable/vector $\mathbf{x}$ comes from the family of probability densities $p(\mathbf{x};\boldsymbol{\theta})$

$$\mathbf{x} \sim p(\mathbf{x};\boldsymbol{\theta}) \tag{1}$$

  but we don't know the true value of $\boldsymbol{\theta}$, denoted by $\boldsymbol{\theta}^*$
- For example, p could be normal distribution, $\boldsymbol{\theta}$ is mean and/or variance parameters
- We observe a *sample* of $\mathbf{x}$, i.e. N observations from $p(\mathbf{x};\boldsymbol{\theta}^*)$

$$[\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(N)] \tag{2}$$

- Fundamental problem: estimate $\boldsymbol{\theta}$ based on the sample
  - Find a good "guess" or "approximation" of the generating $\boldsymbol{\theta}^*$
- Classic method: maximum likelihood estimation (MLE)

New problem: Estimation of energy-based models (EBM)

- Also called estimation of *unnormalized* statistical models
- As above, we want to estimate the parameters of a model
- As usual, the data is a multivariate random vector $\mathbf{x} \in \mathbb{R}^n$
- But here: pdf is known only up to a multiplicative constant

$$p_{\text{norm}}(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} p_{\text{un}}(\mathbf{x}; \boldsymbol{\theta})$$

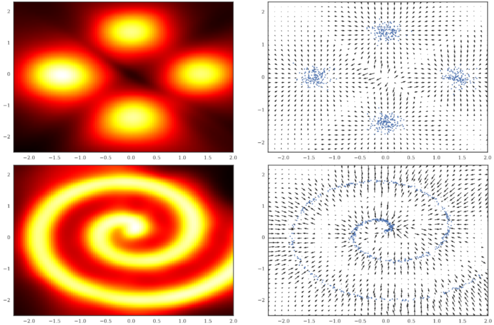
  with p_{norm}: actual pdf, and p_{un}: unnormalized version of pdf
- Functional form of p_{un} is "known" (can be easily computed)
- Normalization constant (= "partition function") $Z(\boldsymbol{\theta})$ "unknown", i.e. *cannot be easily computed*
- By definition of a pdf, must integrate to unity, and thus:

$$Z(\boldsymbol{\theta}) = \int_{\boldsymbol{\xi} \in \mathbb{R}^n} p_{\text{un}}(\boldsymbol{\xi}; \boldsymbol{\theta}) \, d\boldsymbol{\xi} \tag{3}$$

  but computing this is *very* difficult in high dimensions
- Thus, p_{norm} cannot be easily computed either

Motivation in unsupervised deep learning

- We want use a NN as a general model of data pdf
- $p_{\text{un}}(\mathbf{x}; \boldsymbol{\theta})$ is given by the output of a neural network with scalar output; $\boldsymbol{\theta}$ is network parameters
- Extremely difficult to calculate Z by integration (Eq. 3)
- But using special methods, we can learn $\boldsymbol{\theta}$ and pdf:



Right: Data points in blue (and stuff explained later).
Left: Pdf learned by methods explained below.

In general, what is the utility of EBM estimation?

1. If we have a simple parametric model as p_{un}, the parameters often have interesting *interpretations*:
   - Parameters with physical, biological etc. meaning, in case of a scientific model
   - $\rightarrow$ This utility is same as in classical statistical modelling
2. If we train a NN as p_{un}, we get a very general model of the probability density. Useful for:
   - Data generation (later lecture)
   - Visualization of density (in small dimensions)
   - Bayesian inference, using EBM as prior (not in this course)
   - Many more applications...
   - $\rightarrow$ This utility is more specific to deep learning

Why MLE fails for EBM

- Consider Gaussian model with zero mean, variance σ^2

$$p_{\text{norm}}(x; \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2\sigma^2} x^2) \tag{4}$$

- Here, normalization constant $Z(\sigma^2) = \sqrt{2\pi\sigma^2}$
- Suppose *we don't know* Z and want to estimate an EBM

$$p_{\text{un}}(x; \sigma^2) = \exp(-\frac{1}{2\sigma^2} x^2) \tag{5}$$

- Given sample $x_1, \ldots, x_N$, let us (foolishly) try MLE using p_{un}

$$\log L_{\text{un}}(\sigma^2) = -\sum_{i=1}^{N} \frac{1}{2\sigma^2} x_i^2 = -\frac{1}{2\sigma^2} \sum_{i=1}^{N} x_i^2 \tag{6}$$

- Maximized by $\sigma^2 \to \infty$ for any sample $\Rightarrow$ MLE fails

How can we approach EBM estimation?

- Denote the true distribution of data by $p_{\mathbf{x}}$.
- A general theoretical approach is to define some kind of "distance" $\mathcal{D}$ and minimize it:

$$\min_{\boldsymbol{\theta}} \mathcal{D}(p_{\mathrm{un}}(.;\boldsymbol{\theta}), p_{\mathbf{x}}) \tag{7}$$

- Such $\mathcal{D}$ is often called *divergence* in this context, since it may not be a true distance in mathematical sense
- Important point is that for the divergence it should hold:

$$\mathcal{D}(p, q) = 0 \Leftrightarrow p = q \tag{8}$$

- Many such divergences exist for normalized densities
  - Even MLE can be formulated as minimization of a divergence called "Kullback-Leibler"
- The trick is to find one that is easy to compute but still statistically good...
- ... and in our context, even works without normalization !

Definition of "score function" (in this context)

- Define model score function $\mathbb{R}^n \to \mathbb{R}^n$ as

$$\phi(\boldsymbol{\xi};\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial \log p_{\text{norm}}(\boldsymbol{\xi};\boldsymbol{\theta})}{\partial \xi_1} \\ \vdots \\ \frac{\partial \log p_{\text{norm}}(\boldsymbol{\xi};\boldsymbol{\theta})}{\partial \xi_n} \end{pmatrix} = \nabla_{\boldsymbol{\xi}} \log p_{\text{norm}}(\boldsymbol{\xi};\boldsymbol{\theta})$$

  where p_{norm} is normalized model density.
- This is more precisely called the *Stein score*;
  Cf. Fisher score which is derivative wrt. parameter $\boldsymbol{\theta}$
- $\phi(\boldsymbol{\xi};\boldsymbol{\theta})$ does not depend on $Z(\boldsymbol{\theta})$ because

$$\phi(\boldsymbol{\xi};\boldsymbol{\theta}) = \nabla_{\boldsymbol{\xi}} \log \frac{1}{Z(\boldsymbol{\theta})} p_{\text{un}}(\boldsymbol{\xi};\boldsymbol{\theta})$$
$$= \nabla_{\boldsymbol{\xi}} \log p_{\text{un}}(\boldsymbol{\xi};\boldsymbol{\theta}) - \nabla_{\boldsymbol{\xi}} \log Z(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\xi}} \log p_{\text{un}}(\boldsymbol{\xi};\boldsymbol{\theta}) - 0 \quad (9)$$

- As far as score function is concerned, *no need* to compute normalization constant Z, non-normalized pdf p_{un} is enough!

Score matching: definition of objective function

- How to use the score function for estimating parameters?
- Define data score function as

$$\phi_{\mathbf{x}}(\boldsymbol{\xi}) = \nabla_{\boldsymbol{\xi}} \log p_{\mathbf{x}}(\boldsymbol{\xi})$$

  where observed data is assumed to follow pdf $p_{\mathbf{x}}(.)$.
- Estimate parameters by minimizing distance between model score function $\phi(.;\boldsymbol{\theta})$ and score of observed data $\phi_{\mathbf{x}}(.)$:

$$J(\boldsymbol{\theta}) = \frac{1}{2} \int_{\boldsymbol{\xi} \in \mathbb{R}^n} p_{\mathbf{x}}(\boldsymbol{\xi}) \| \phi(\boldsymbol{\xi};\boldsymbol{\theta}) - \phi_{\mathbf{x}}(\boldsymbol{\xi}) \|^2 d\boldsymbol{\xi} \qquad (10)$$

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- This gives a consistent estimator almost by construction
- (Some call this "score-based modelling" instead of EBM)
- But: computation of this is not straightforward...

A computational trick: central theorem of score matching

- In objective function we have score of data distribution $\phi_{\mathbf{x}}(.)$... Not easy to compute. But there is a trick:
- Write out squared norm in objective function $J(\boldsymbol{\theta})$:

$$\frac{1}{2}\int p_{\mathbf{x}}(\boldsymbol{\xi})\|\phi(\boldsymbol{\xi};\boldsymbol{\theta})\|^2 d\boldsymbol{\xi} - \int p_{\mathbf{x}}(\boldsymbol{\xi})\phi_{\mathbf{x}}(\boldsymbol{\xi})^T\phi(\boldsymbol{\xi};\boldsymbol{\theta})d\boldsymbol{\xi} + \text{const.}$$

- Constant does not depend on $\boldsymbol{\theta}$. First term easy to compute.
- The trick is to use *integration by parts* on second term. In one dimension:

$$\int p_x(x)(\log p_x)'(x)\phi(x;\theta)dx = \int p_x(x)\frac{p_x'(x)}{p_x(x)}\phi(x;\theta)dx$$
$$= \int p_x'(x)\phi(x;\theta)dx = 0 - \int p_x(x)\phi'(x;\theta)dx$$

- We got rid of score function of data distribution $p_{\mathbf{x}}(x)$!

Theorem on score matching

- We have now (kind of) proven:

Theorem

Assume some regularity conditions, and smooth densities. Then, the score matching objective function J can be expressed as

$$J(\boldsymbol{\theta}) = \int_{\boldsymbol{\xi}\in\mathbb{R}^n} p_{\mathbf{x}}(\boldsymbol{\xi}) \sum_{j=1}^{n} \left[\partial_j \phi_j(\boldsymbol{\xi};\boldsymbol{\theta}) + \frac{1}{2}\phi_j(\boldsymbol{\xi};\boldsymbol{\theta})^2 \right] d\boldsymbol{\xi} + const. \quad (10)$$

where the constant does not depend on $\boldsymbol{\theta}$, and

$$\phi_j(\boldsymbol{\xi};\boldsymbol{\theta}) = \frac{\partial \log p_{un}(\boldsymbol{\xi};\boldsymbol{\theta})}{\partial \xi_j}, \quad \partial_j \phi_j(\boldsymbol{\xi};\boldsymbol{\theta}) = \frac{\partial^2 \log p_{un}(\boldsymbol{\xi};\boldsymbol{\theta})}{\partial \xi_j^2}$$

Final method of score matching

- Replace integration over data density $p_{\mathbf{x}}(.)$ by sample average
  - transform a theoretical distance to a concrete learning objective
- Given N observations $\mathbf{x}(1), \ldots, \mathbf{x}(N)$, we have objective

$$\tilde{J}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{n} \left[\partial_j \phi_j(\mathbf{x}(i); \boldsymbol{\theta}) + \frac{1}{2} \phi_j(\mathbf{x}(i); \boldsymbol{\theta})^2 \right] \tag{10}$$

  where ϕ_j is j-th entry in score function (partial derivative of non-normalized model log-density $\log p_{\mathrm{un}}$), and $\partial_j \phi_j$ a second partial derivative. *All derivatives are with respect to x_j!*
- No difficult integrals; no $\boldsymbol{\phi}_{\mathbf{x}}$ or $p_{\mathbf{x}}$ left!
- Only needs evaluation of some derivatives of p_{un}
- Thus: a computationally relatively simple and statistically consistent method for parameter estimation in EBM
- But: while no integration, needs higher derivatives...

Example of Gaussian multivariate precision

- Precision = inverse of covariance, denote by $\mathbf{M}$
- Let's pretend we cannot normalize Gaussian pdf (zero-mean):

$$p_{\text{un}}(\mathbf{x}; \mathbf{M}) = \frac{1}{Z(\mathbf{M})} \exp(-\frac{1}{2}\mathbf{x}^T \mathbf{M}\mathbf{x}) \tag{11}$$

We calculate (in exercises)

$$\phi(\mathbf{x}; \mathbf{M}) = -\mathbf{M}\mathbf{x}, \qquad \partial_j \phi_j(\mathbf{x}; \mathbf{M}) = -m_{jj}$$

and obtain

$$\tilde{J}(\mathbf{M}) = \frac{1}{N} \sum_{i=1}^{N} [\sum_j -m_{jj} + \frac{1}{2}\mathbf{x}(i)^T \mathbf{M}\mathbf{M}\mathbf{x}(i)] \tag{12}$$

- Quadratic function! Easy to optimize.
- In 1D: $J(\sigma^2) = -\sigma^{-2} + \frac{1}{2}\sigma^{-4} \sum_i x_i^2 / N$, cf. with the foolish MLE attempt a few slides ago
- Turns our to be equal to MLE: optimizing $\hat{\mathbf{M}}$ is inverse of sample covariance

Denoising score matching (DSM)

- Second derivatives in basic SM expensive to compute in a NN
- DSM: using denoising autoencoders to estimate score
- Easy to implement in a NN, no need for higher derivatives
- Denoising autoencoders (reminder):
  1. Add (small, Gaussian) noise $\mathbf{n}$ to the data to get $\tilde{\mathbf{x}} = \mathbf{x} + \mathbf{n}$
  2. Train NN to learn a denoising function to reconstruct $\mathbf{x}$ from $\tilde{\mathbf{x}}$
- Objective function formulated as

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{N} \|\mathbf{x}(i) - [\tilde{\mathbf{x}}(i) + \sigma^2 \phi(\tilde{\mathbf{x}}(i); \boldsymbol{\theta})]\|^2 \tag{13}$$

  where noise is Gaussian of covariance $\sigma^2 \mathbf{I}$,
  ϕ is NN with parameters $\boldsymbol{\theta}$; nonlinearity "split" in special way
- Deep theorem: Minimizing this will lead to ϕ which is the score function of $\tilde{\mathbf{x}}$ (note: not original $\mathbf{x}$)

Connection between score function and denoising

- How come learning to denoise gives us the score function?
- Assume as above

$$\tilde{\mathbf{x}} = \mathbf{x} + \mathbf{n} \tag{14}$$

  - $\mathbf{n}$ is Gaussian with covariance $\mathbf{C_n}$.
- The optimal prediction of $\mathbf{x}$ from $\tilde{\mathbf{x}}$ is given by

Theorem (Tweedie-Miyasawa)

$$E\{\mathbf{x}|\tilde{\mathbf{x}}\} = \tilde{\mathbf{x}} + \mathbf{C_n}\phi_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}) \tag{15}$$

- Add to this a well-known property of conditional expectation: it minimizes the least-squares error in prediction
- So, predicting $\mathbf{x}$ from $\tilde{\mathbf{x}}$ as in the the right-hand-side of (15) by least-squares regression is optimized by the score function: $\phi(\tilde{\mathbf{x}};\boldsymbol{\theta}) = \phi_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}) \Rightarrow$ DSM works.

Proof of Tweedie-Miyasawa formula (optional)

Denote the pdf of $\mathbf{n}$ as $\psi(\mathbf{n}) = \frac{1}{(2\pi)^{d/2}|\mathbf{C_n}|^{1/2}} \exp(-\frac{1}{2}\mathbf{n}^T \mathbf{C_n}^{-1}\mathbf{n})$.
To prove the theorem, we start by the simple identity

$$p_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}) = \int p(\tilde{\mathbf{x}}|\mathbf{x}) p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} = \int \psi(\tilde{\mathbf{x}} - \mathbf{x}) p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \tag{16}$$

based first, on definition of marginal pdf, and second, the noise model saying $p(\tilde{\mathbf{x}}|\mathbf{x}) = \psi(\tilde{\mathbf{x}} - \mathbf{x})$. Now, we take derivatives of both sides (ignoring the middle term)

$$\nabla_{\tilde{\mathbf{x}}} p_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}) = \int \mathbf{C_n}^{-1}(\mathbf{x} - \tilde{\mathbf{x}}) \psi(\tilde{\mathbf{x}} - \mathbf{x}) p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \tag{17}$$

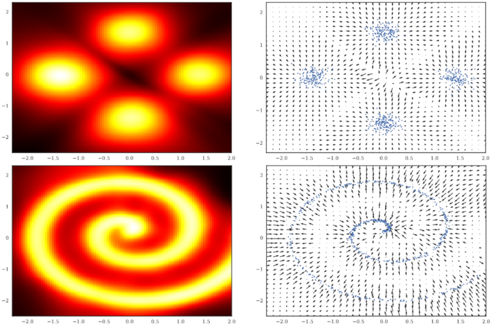
We divide both sides by $p_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}})$, multiply by $\mathbf{C_n}$, and rearrange:

$$\begin{aligned} \mathbf{C_n} \phi_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}) &= \int (\mathbf{x} - \tilde{\mathbf{x}}) \psi(\tilde{\mathbf{x}} - \mathbf{x}) p_{\mathbf{x}}(\mathbf{x}) / p_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}) d\mathbf{x} \\ &= \int \mathbf{x} p(\mathbf{x}|\tilde{\mathbf{x}}) d\mathbf{x} - \tilde{\mathbf{x}} \int p(\mathbf{x}|\tilde{\mathbf{x}}) d\mathbf{x} \end{aligned} \tag{18}$$

The last integral is equal to unity, which proves Eq. (15). $\square$

Earlier example revisited

- $\log p_{\text{un}}(\mathbf{x}; \boldsymbol{\theta})$ is given by the output of a neural network with scalar output; $\boldsymbol{\theta}$ is network parameters
- A neural network is here estimated by DSM



Left: Pdf learned by DSM.
Right: Data points in blue, learned score function as arrows

Summary: EBM part 1

- Definition of EBM: an *unnormalized* statistical model
- Lack of normalization makes estimation much more difficult
- Estimation *not* possible with maximum likelihood
- Score matching is one approach
  - Completely ignores normalization constant
- Basic SM still has annoying higher derivatives
- Denoising SM gets rid of higher derivatives
  - Equivalent to a denoising autoencoder
- Very general method for approximating data pdf
  - but small problem: DSM approximates pdf of *noisy* data only