On Autoencoders and Score Matching for Energy Based Models

Kevin Swersky Marc'Aurelio Ranzato David Buchman
Benjamin M. Marlin Nando de Freitas

Toronto Machine Learning Group Meeting, 2011

Goal: Unsupervised Feature Learning

- Automatically learn features from data
- Useful for modeling images, speech, text, etc. without requiring lots of prior knowledge
- Many models and algorithms, we will focus on two commonly used ones

Motivation

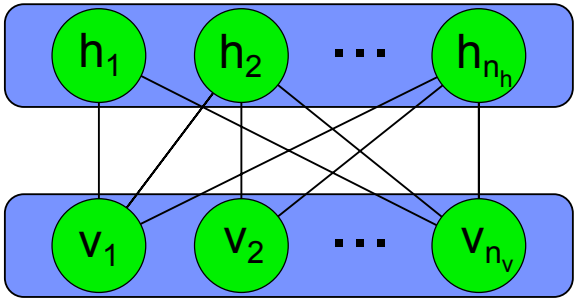
- Autoencoders and energy based models represent the state-of-the-art for feature learning in several domains
- We provide some unifying theory for these model families
- This will allow us to derive new autoencoder architectures from energy based models
- It will also give us an interesting way to regularize these autoencoders

Latent Energy Based Models

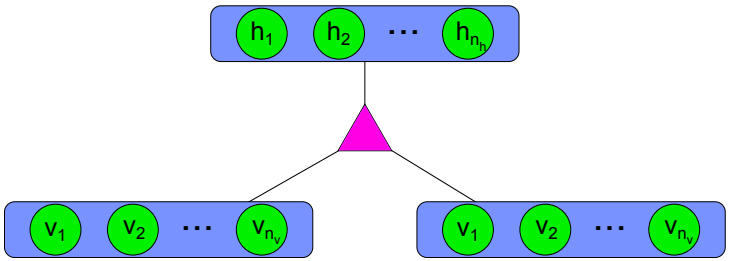
- A probability distribution over data vectors v, with latent variables h and parameters θ
- Defined using an *energy function* $E_\theta(v, h)$ as:

$$P_\theta(v) = \frac{\exp(-E_\theta(v,h))}{Z(\theta)}$$

- Normalized by the *partition function* $Z(\theta) = \int_v \int_h \exp(-E_\theta(v,h)) dh\, dv$
- $Z(\theta)$ is usually not analytical
- Features given by h



Restricted Boltzmann Machine



Factored EBM (e.g. mPoT)

Example: Gaussian-Bernoulli RBM

An Gaussian-Bernoulli restricted Boltzmann machine is an energy based model with visible (observed) variables $v \in \mathbb{R}^D$, hidden (latent) variables $h \in \{0,1\}^K$, and parameters $\theta = \{W\}$. The energy is given by:

$$E_\theta(v,h) = \frac{1}{2} v^T v - v^T W h$$

The free-energy is obtained by marginalizing over the hidden variables:

$$F_\theta(v) = \frac{1}{2} v^T v - \sum_{j=1}^{K} \log\left(1 + \exp\left(v^T W_j\right)\right)$$

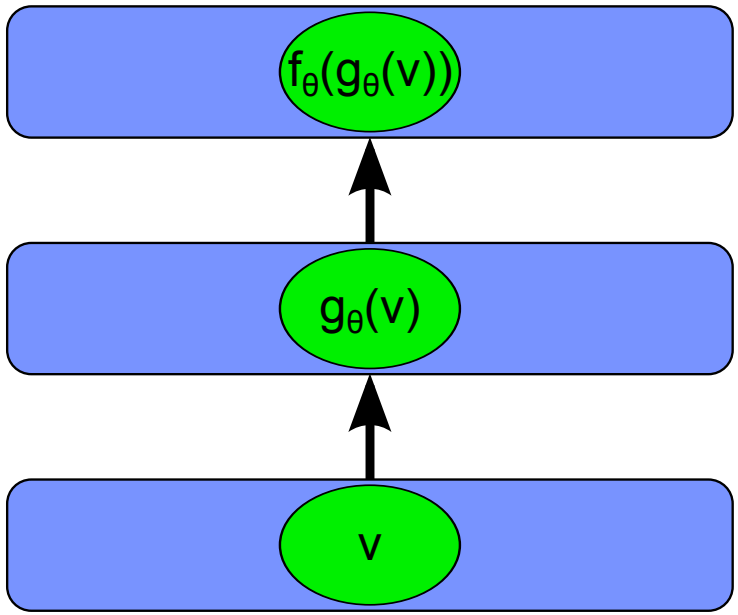
Where W_j represents the j^{th} column of W.

Autoencoders

- A feed-forward neural network designed to reconstruct its input
- Encoding function: $g_\theta(v)$
- Reconstruction function: $f_\theta(g_\theta(v))$
- Minimizes reconstruction error:

$$\hat{\theta} = \arg\min_\theta ||f_\theta(g_\theta(v)) - v||^2$$

- Features given by $g_\theta(v)$



Example: One-Layer Autoencoder

A commonly used autoencoder architecture maps inputs $v \in \mathbb{R}^D$ to outputs $\hat{v} \in \mathbb{R}^D$ through a deterministic hidden layer $\hat{h} \in [0,1]^K$ using parameters $\theta = \{W\}$ by the following system:

$$\hat{v} = f_\theta(g_\theta(v)) = W\hat{h}$$
$$\hat{h} = g_\theta(v) = \sigma(W^T v)$$

Where $\sigma(x) = \frac{1}{1+\exp(-x)}$.
We train this by minimizing the reconstruction error:

$$\ell(\theta) = \sum_{i=1}^{D} \frac{1}{2}(v_i - \hat{v}_i)^2$$

Comparison

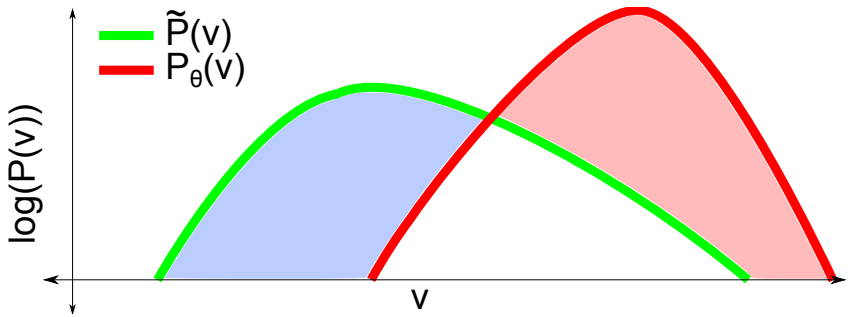
Energy based models	Autoencoders
• Undirected • Probabilistic • Slow inference • Intractable objective function	• Directed • Deterministic • Fast inference • Tractable objective function

- Unifying these model families lets us leverage the advantages of each

Maximum Likelihood for EBMs

- Minimize KL divergence between empirical data distribution $\widetilde{P}(v)$ and model distribution $P_\theta(v)$:

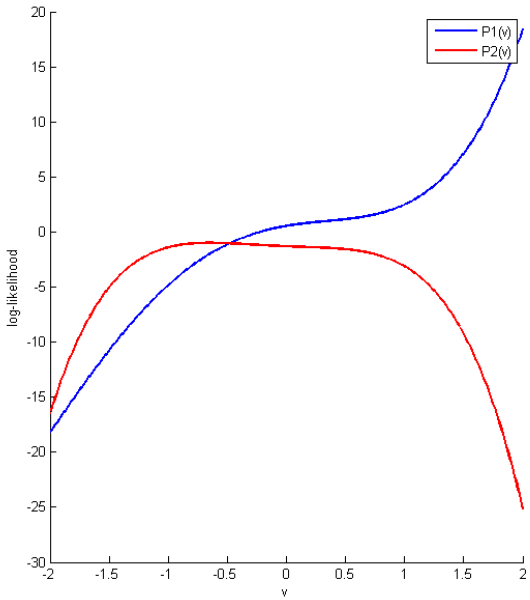
$$\hat{\theta} = \arg\min_\theta \mathbb{E}_{\widetilde{p}(v)} \left[\log(\widetilde{P}(v)) - \log(P_\theta(v)) \right]$$

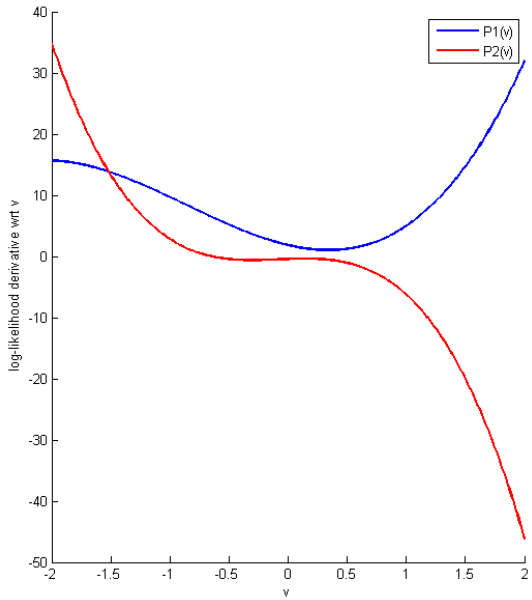


- Requires that we know $Z(\theta)$
- Idea: Approximate $Z(\theta)$ with samples from $P_\theta(v)$
  - Uses MCMC
  - Contrastive divergence (CD), persistent contrastive divergence (PCD), fast PCD (FPCD)

Score Matching (Hyvarinen, 2006)

- Intuition: If two functions have the same derivatives everywhere, then they are the same functions up to a constant
- We match $\frac{\partial}{\partial v_i} \log\left(\widetilde{P}(v)\right)$ and $\frac{\partial}{\partial v_i} \log\left(P_\theta(v)\right)$
- The constraint $\int_v P(v)dv = 1$ means that the distributions must be equal if their derivatives are perfectly matched





Score Matching (2)

- Minimize distance between the *score functions* $\nabla_v \log(\widetilde{P}(v))$ and $\nabla_v \log(P_\theta(v))$:

$$\hat{\theta} = \arg\min_\theta J(\theta) = \mathbb{E}_{\widetilde{p}(v)} \left[||\nabla_v \log(\widetilde{P}(v)) - \nabla_v \log(P_\theta(v))||^2 \right]$$

- Equivalent to (Hyvarinen, 2006):

$$J(\theta) = \mathbb{E}_{\widetilde{p}(v)} \left[\sum_{i=1}^{D} \frac{1}{2} \left(\frac{\partial \log(P_\theta(v))}{\partial v_i} \right)^2 + \frac{\partial^2 \log(P_\theta(v))}{\partial v_i^2} \right] + const$$

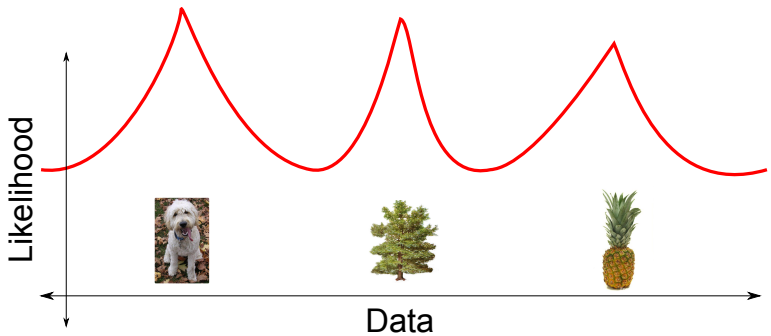
- Does not depend on $Z(\theta)$ since $\nabla_v Z(\theta) = 0$

Score Matching (3)

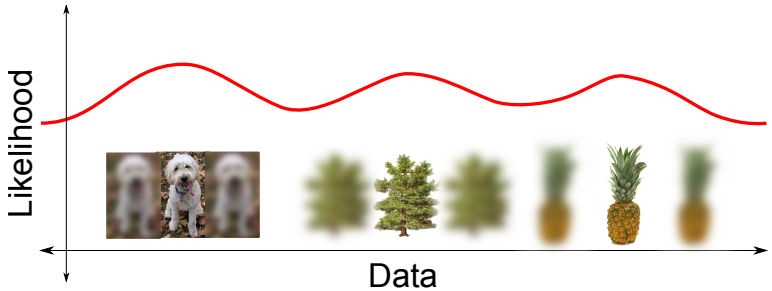
- Score matching is not as statistically efficient as maximum likelihood
  - It is in fact an approximation to pseudolikelihood (Hyvarinen, 2006)
- Statistical efficiency is only beneficial if you can actually find good parameters
- We trade efficiency for the ability to find better parameters with more powerful optimization

Denoising Score Matching (Vincent, 2011)

- Rough idea:
  1. Apply a kernel $q_\beta(v|v')$ to $\widetilde{P}(v)$ get a smoothed distribution $q_\beta(v) = \int q_\beta(v|v')\widetilde{P}(v')dv'$
  2. Apply score matching to $q_\beta(v)$
- Faster than ordinary score matching (no 2nd derivatives)
- Includes denoising autoencoders (for continuous data) as a special case



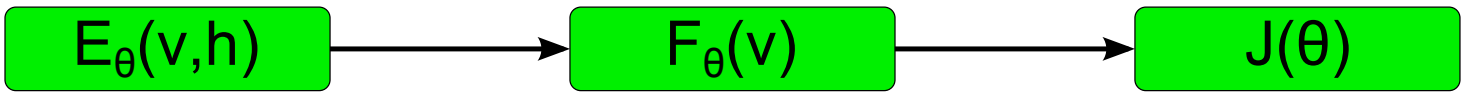
$\widetilde{P}_\theta(v)$



$q_\beta(v)$

Score Matching Recipe

1. Write down energy function $E_\theta(v, h)$
2. Marginalize over latent variables to form $F_\theta(v)$
3. Derive score matching objective $J(\theta)$
   - In practice, use Theano to do this step!



Example: RBM with All Gaussian Units

- If both v and h are Gaussian-distributed, we have the following energy and free-energy:

$$E_\theta(v,h) = \frac{1}{2}v^T v + \frac{1}{2}h^T h - v^T W h$$

$$F_\theta(v) = \frac{1}{2}v^T \left(\mathrm{I} - WW^T\right) v$$

$$J(\theta) = \sum_{i=1}^{D}\left[\frac{1}{2}\left(v_i - \sum_{j=1}^{K} W_{ij}\left(W_j^T v\right)\right)^2 + \sum_{j=1}^{K} W_{ij}^2\right]$$

- This is a linear autoencoder with weight-decay regularization

Example: RBM with All Gaussian Units (2)

- With denoising score matching:

$$J(\theta) = \sum_{i=1}^{D} \frac{1}{2} \left(v_i - \sum_{j=1}^{K} W_{ij} \left(W_j^T x \right) \right)^2$$

$$x \sim N(v, \sigma^2 \mathrm{I})$$

- Adding noise to the input is roughly equivalent to weight-decay (Bishop, 1994)

Example: Gaussian-Bernoulli RBMs

- We apply score matching to the GBRBM model:

$$J(\theta) = const + \sum_{i=1}^{D} \left[\frac{1}{2} (v_i - \hat{v}_i)^2 + \sum_{j=1}^{K} W_{ij}^2 \hat{h}_j \left(1 - \hat{h}_j\right) \right]$$

- This is also a regularized autoencoder
- The regularizer is a sum of weighted variances of the Bernoulli random variables h conditioned on v

Example: mPoT (Warning: lots of equations!)

- The GBRBM is bad at modelling heavy-tailed distributions, and capturing covariance structure in v
- We can use more sophisticated energy-based models to overcome these limitations

The mPoT is an energy based model with Gaussian-distributed visible units, Gamma-distributed hidden units $h^c \in \mathbb{R}^{+K}$, and Bernoulli-distributed hidden units $h^m \in \{0,1\}^M$ with parameters $\theta = \{C, W\}$.

$$E_\theta(v, h^m, h^c) = \sum_{k=1}^{K} \left[h_k^c (1 + \frac{1}{2}(C_k^T v)^2) + (1-\gamma)\log(h_k^c) \right] + \frac{1}{2} v^T v - v^T W h^m$$

$$F_\theta(v) = \sum_{k=1}^{K} \gamma \log(1 + \frac{1}{2}(\phi_k^c)^2) - \sum_{j=1}^{M} \log(1 + e^{\phi_j^m}) + \frac{1}{2} v^T v$$

$$\phi_k^c = C_k^T v$$

$$\phi_j^m = W_j^T v$$

Example: mPoT (2) (Warning: lots of equations!)

$$J(\theta) = \sum_{i=1}^{D} \frac{1}{2} \psi_i(p_\theta(v))^2 + \sum_{k=1}^{K} \left(\rho(\hat{h}_k^c) D_{ik}^2 + \hat{h}_k^c K_{ik} \right) + \sum_{j=1}^{M} \left(\hat{h}_j^m (1 - \hat{h}_j^m) W_{ij}^2 \right) - 1$$

$$\psi_i(p_\theta(v)) = \sum_{k=1}^{K} \hat{h}_k^c D_{ik} + \sum_{j=1}^{M} \hat{h}_j^m W_{ij} - v_i$$

$$K_{ik} = \sum_{k=1}^{K} -C_{ik}^2$$

$$D_{ik} = \sum_{k=1}^{K} \left(- \left(C_k^T v \right) C_{ik} \right)$$

$$\hat{h}_k^c = \frac{\gamma}{1 + \frac{1}{2}(\phi_k^c)^2}$$

$$\hat{h}_j^m = \text{sigm}(\phi_j^m)$$

$$\rho(x) = x^2$$

Score Matching for General Latent EBMs

- Score matching can produce complicated and difficult to interpret objective functions
- Our solution is to apply score matching to latent EBMs without directly specifying $E_\theta(v, h)$

Score Matching for General Latent EBMs (2)

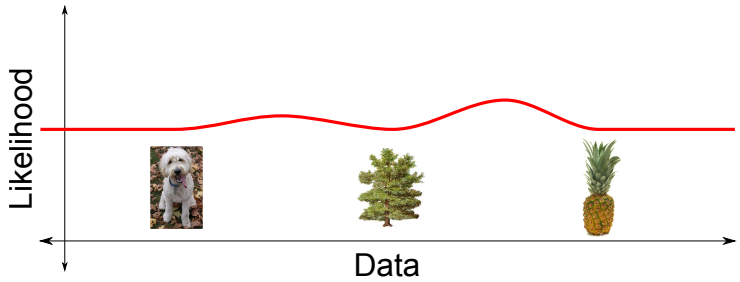
Theorem

The score matching objective for a latent EBM can be expressed succinctly in terms of expectations of the energy with respect to the conditional distribution $P_\theta(h|v)$.

$$J(\theta) = \mathbb{E}_{\tilde{p}(v)} \left[\sum_{i=1}^{n_v} \frac{1}{2} \left(\mathbb{E}_{p_\theta(h|v)} \left[\frac{\partial E_\theta(v,h)}{\partial v_i} \right] \right)^2 + var_{p_\theta(h|v)} \left[\frac{\partial E_\theta(v,h)}{\partial v_i} \right] - \mathbb{E}_{p_\theta(h|v)} \left[\frac{\partial^2 E_\theta(v,h)}{\partial v_i^2} \right] \right].$$

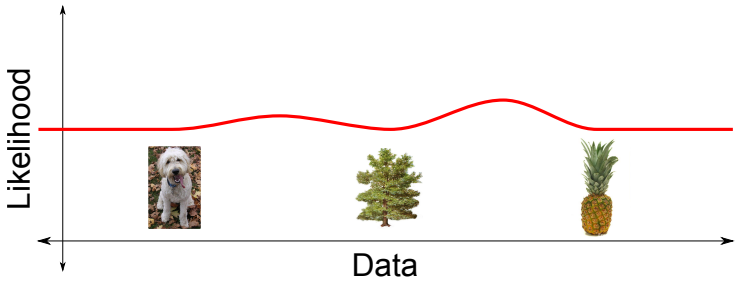
- The score matching objective for a latent EBM always consists of 3 terms:
  1. Squared error
  2. Variance of error
  3. Curvature
- Variance and curvature can be seen as regularizers

Visualization

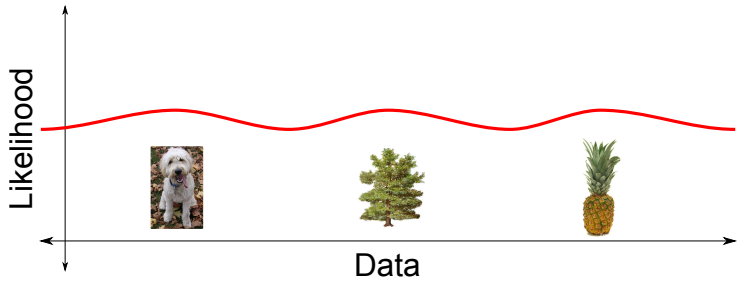


Model probability before learning

Visualization

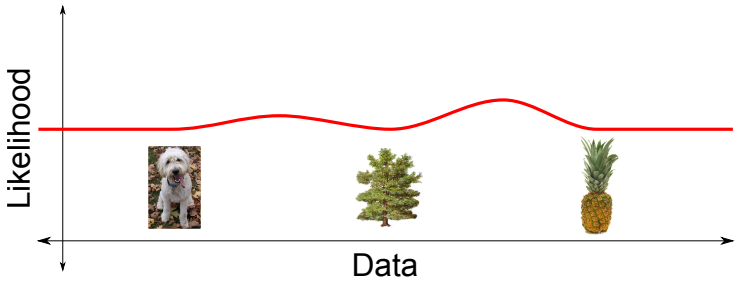


Model probability before learning

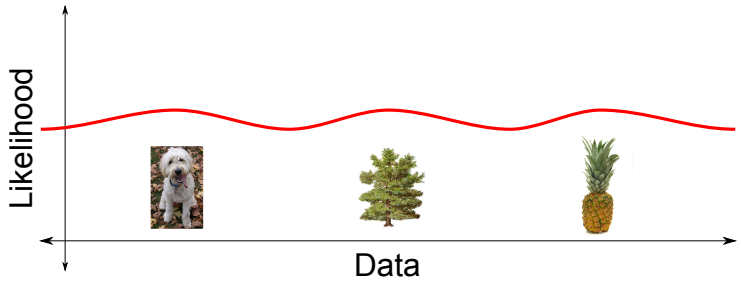


Squared error centers mass on data

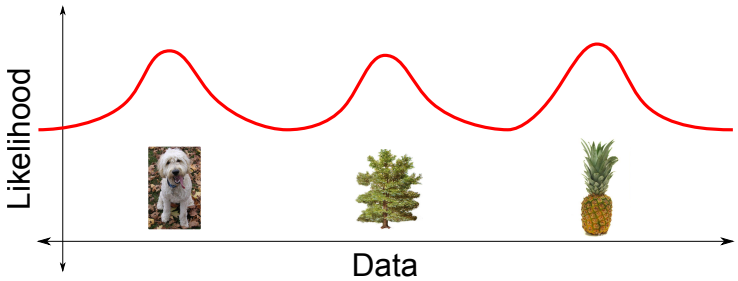
Visualization



Model probability before learning



Squared error centers mass on data



Variance and Curvature shape the distribution

Gradient Steps on the Free Energy

- Consider taking a gradient step along the free-energy with step size ε:

$$v_i^{(t+1)} = v_i^{(t)} - \varepsilon \frac{\partial F_\theta(v)}{\partial v_i}$$

- When $F_\theta(v) = \tilde{F}_\theta(v) + \frac{1}{2} v^T v$ term, we get:

$$v_i^{(t+1)} = v_i^{(t)} - \varepsilon \left(v_i^{(t)} + \frac{\partial \tilde{F}\theta(v)}{\partial v_i} \right)$$

- Setting ε to 1:

$$v_i^{(t+1)} = \frac{\partial \tilde{F}\theta(v)}{\partial v_i}$$

- An autoencoder can be thought of as performing a reconstruction by taking a gradient step along its free energy with a step-size of 1

Gaussian Energy Functions

- If the energy function forms a Gaussian distribution in v:

$$E_\theta(v,h) = \tfrac{1}{2}(v-\mu_h)^T \Omega_h (v-\mu_h) + \text{const}$$

- Then the squared error will take the form:

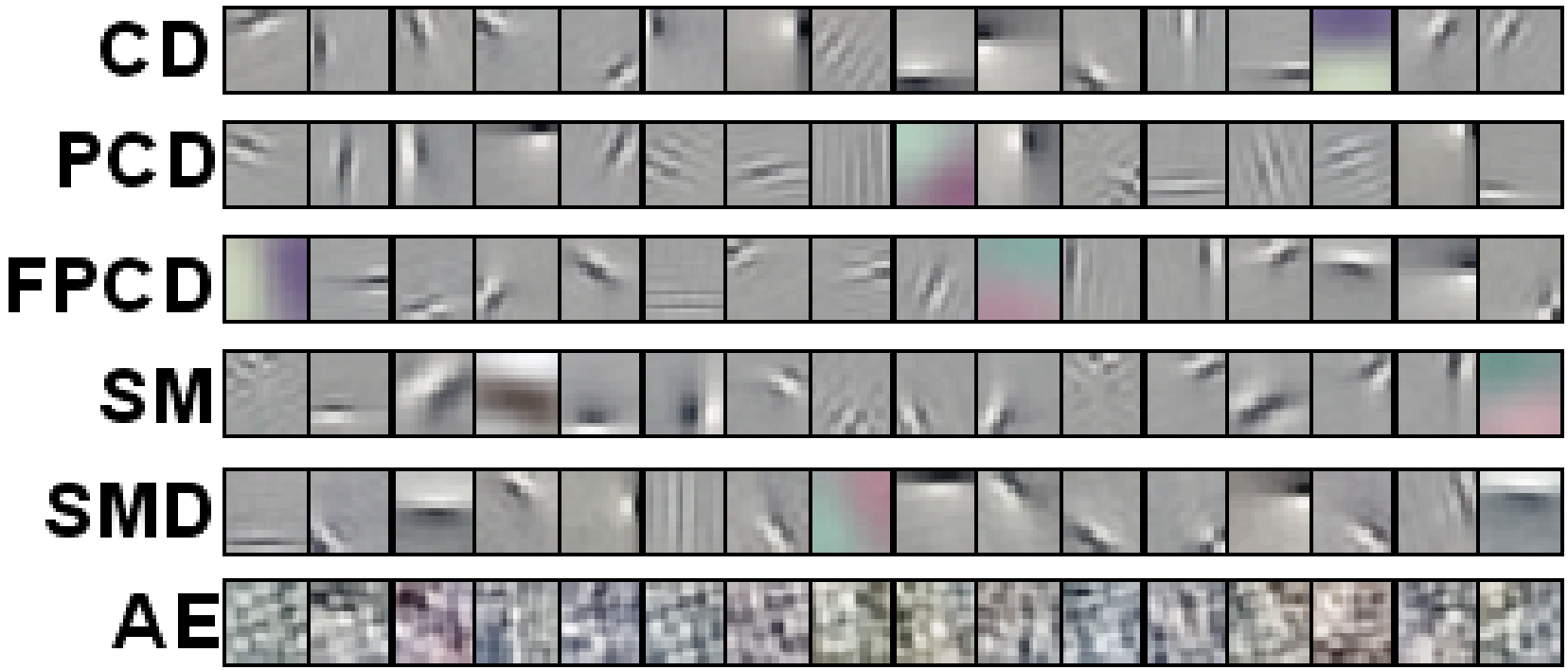
$$\tfrac{1}{2}\left(\mathbb{E}_{p_\theta(h|v)}\left[\Omega_h(v-\mu_h)\right]\right)^2$$

- This corresponds to a more general autoencoder reconstruction error
- For a GBRBM: $\Omega_h = \mathrm{I}$
- If $\Omega_h \neq \mathrm{I}$ then the autoencoder will also model covariance in v

Objectives

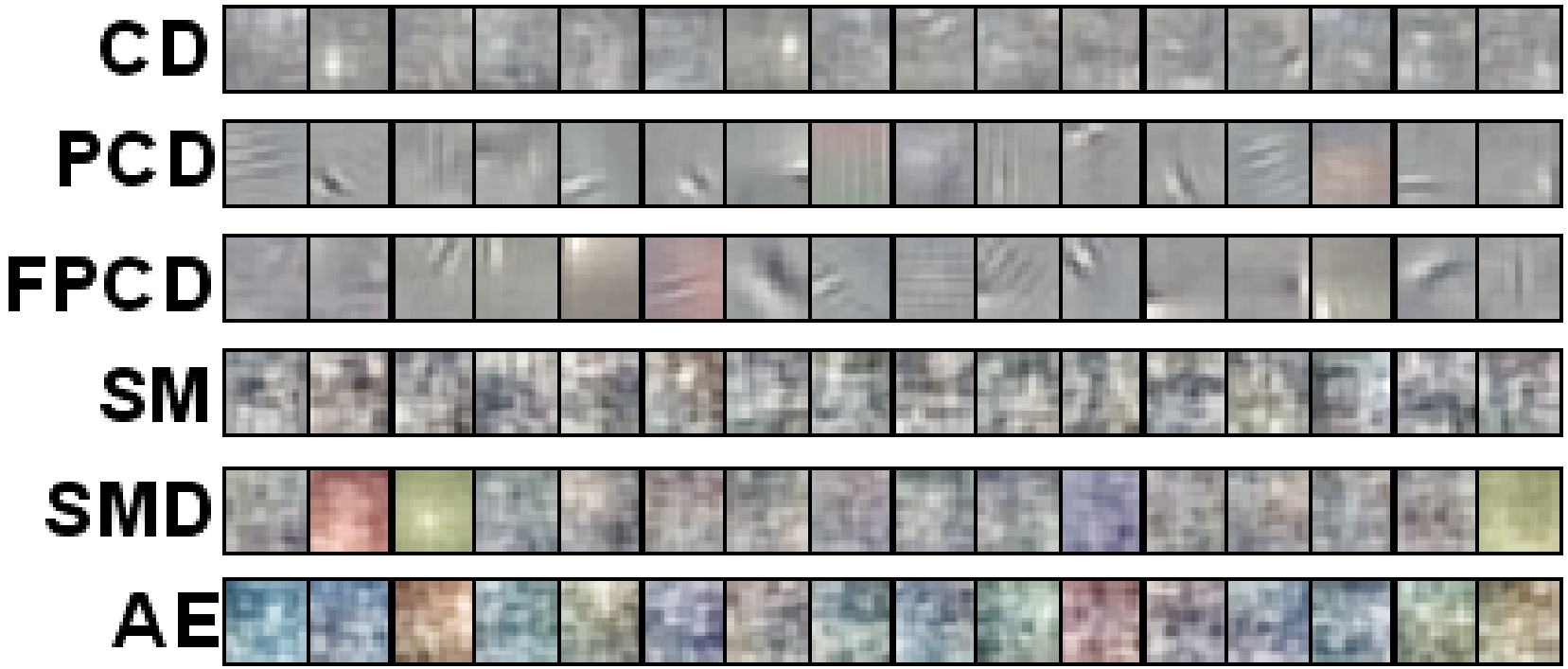
- Can we learn good features with richer autoencoder models?
- How do score matching estimators compare to maximum likelihood estimators?
- What are the effects of regularization in a score matching-based autoencoder?

Learned Features



Covariance filters

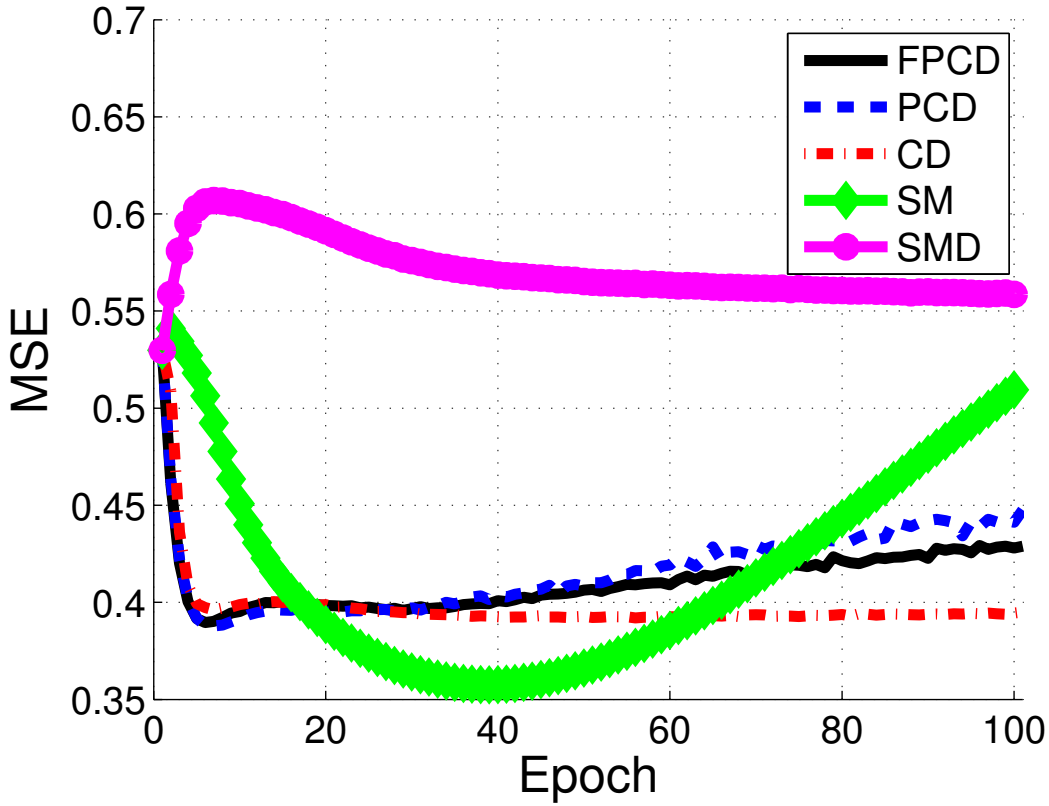
Learned Features (2)



Mean filters

Denoising

- Reconstruct an image from a blurry version
  - Finds a nearby image that has a high likelihood

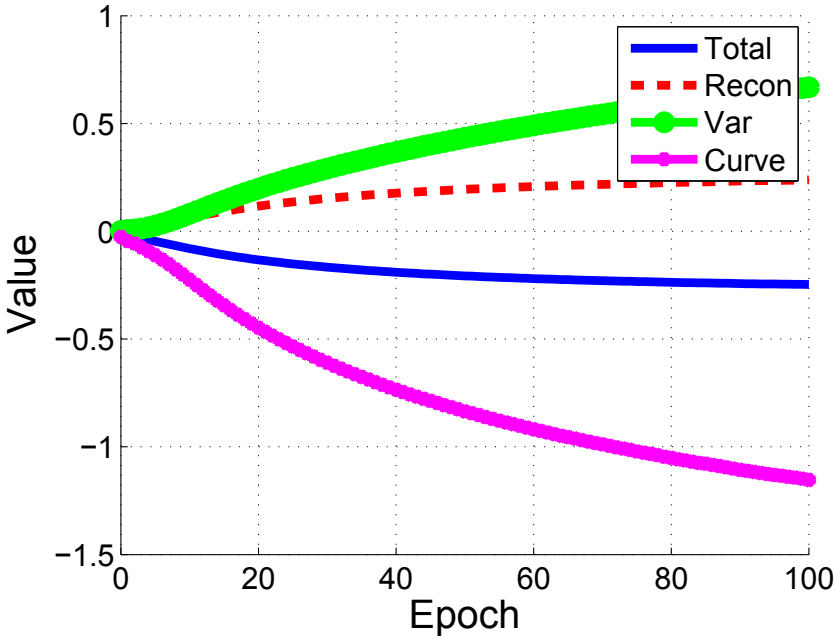


Classification on CIFAR-10

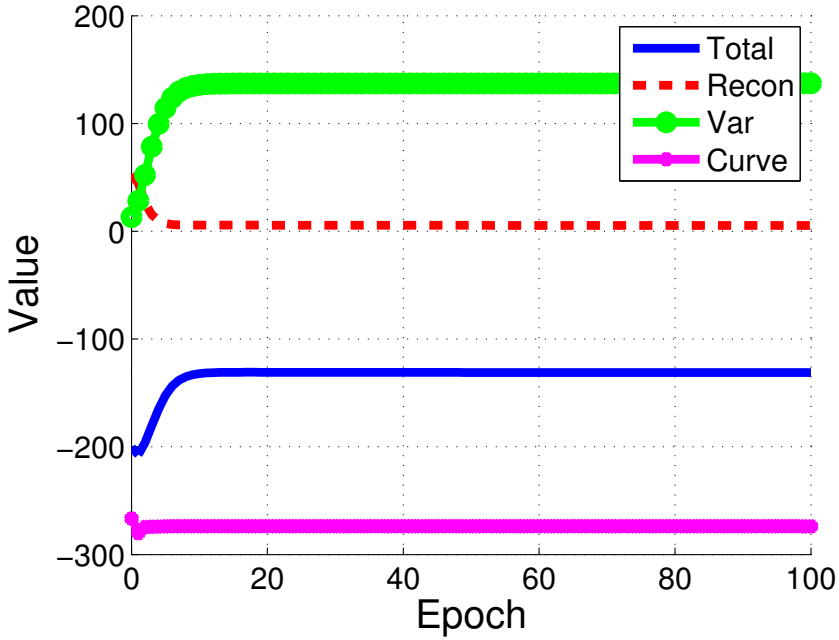
CD	PCD	FPCD	SM	SMD	AE
64.6%	64.7%	65.5%	65.0%	64.7%	57.6%

- Most methods are comparable
- mPoT autoencoder without regularization does significantly worse

Effect of Regularizers on mPoT Autoencoder



Score matching mPoT autoencoder ($\times 10^4$)



Autoencoder with no regularization

- Total (blue curve) is the sum of the other curves
- Other curves represent the 3 score matching terms

Quick Mention: Memisevic, 2011

- Roland Memisevic, a recent U of T grad published a paper in this year's ICCV
- He derived the same models through the idea of relating two different inputs using an autoencoder

Conclusion

- Score matching provides a unifying theory for autoencoders and latent energy based models
- It reveals novel regularizers for some models
- We can derive novel autoencoders that model richer structure
- We provide an interpretation of the score matching objective for all latent EBMs
- Properly shaping the implied energy of an autoencoder is one key to learning good features
  - This is done implicitly when noise is added to the input
  - Perhaps also done implicity with sparsity penalties, etc.

Future Work

- We can model other kinds of data with higher-order features by simply changing the loss function
- Extend higher-order models to deeper layers
- Apply Hessian-free optimization to EBMs and related autoencoders