COMPSCI 689
Lecture 19: Generalizing Autoencoders and RBMs

Benjamin M. Marlin

College of Information and Computer Sciences
University of Massachusetts Amherst

Slides by Benjamin M. Marlin (marlin@cs.umass.edu).

Factor Analysis to Autoencoders

- Last class, we saw how non-linear autoencoder neural network models generalize latent linear models (i.e., Factor Analysis).
- This allows autoencoders to model real-valued data defined on non-linear real manifolds.
- We also previously saw how factor analysis models can be generalized to deal with other inputs.
- In this class we'll provide a similar generalization for autoencoders, and introduce restricted Boltzmann machines.

Generalized Autoencoders

- We can use the same ideas we applied to generalize factor analysis to generalize autoencoders to different data types.
- In general, the input to first hidden layer structure remains the same (an exception is categorical variables, which need to be converted to a one-hot encoding).
- The hidden-to-output structure can be interpreted as computing the parameters of an exponential family distribution for each X_d using the last layer of hidden values $\mathbf{h}^L = [h_1^L, ..., h_{K_L}^L]$.

Example: Binary Autoencoders

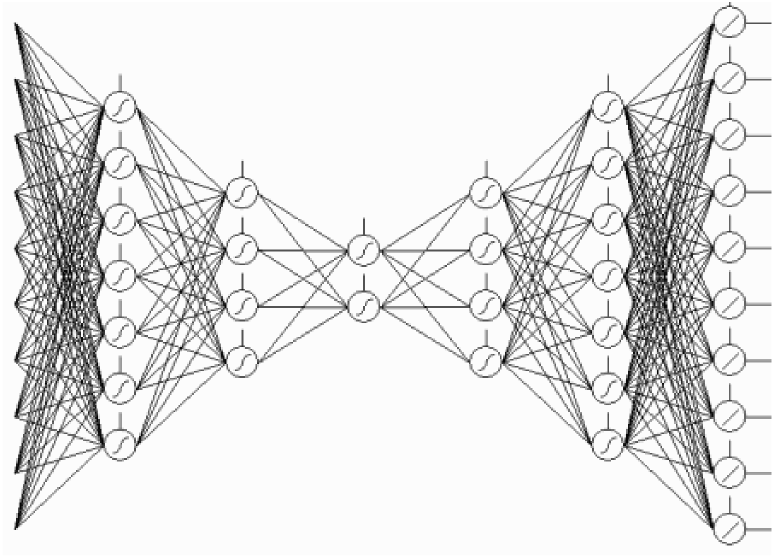
- Suppose $\mathbf{X} = [X_1, ..., X_D]$ consists of binary values only.
- Let the last layer of hidden values be $\mathbf{h}^L = [h_1^L, ..., h_{K_L}^L]$.
- Then we can define the probability of the reconstruction $\mathbf{r}$ given the last layer of hidden unit values $\mathbf{h}^L$ as follows:

$$\theta_{d|h} = \frac{1}{1 + \exp(-\mathbf{W}_d^L \mathbf{h}^L)}$$

$$P(r_d|\mathbf{h}^L) = \theta_{d|h}^{r_d}(1 - \theta_{d|h})^{(1-r_d)}$$

$$P(\mathbf{R} = \mathbf{r}|\mathbf{h}^L) = \prod_{d=1}^{D} P(\mathbf{r}_d|\mathbf{h}^L)$$

Example: Deep Generalized Autoencoders



Learning Deep Generalized Autoencoders

- To learn a deep generalized autoencoder, we optimize the objective function shown below where we recall that $\mathbf{h}_n^L$ is computed using a forward pass through the autoencoder starting from input $\mathbf{x}_n$:

$$\mathcal{J}(\mathcal{D}, \mathbf{W}^{1:L}) = \sum_{n=1}^{N} \log P(\mathbf{R} = \mathbf{x}_n | \mathbf{h}_n^L)$$

- This function effectively replaces the mean squared error between the input and the output used in a standard autoencoder with a log loss derived from $P(\mathbf{R} = \mathbf{x}_n | \mathbf{h}_n^L)$.
- Using a Gaussian distribution will recover squared loss. Using a Laplacian distribution will recover absolute loss. A product of Bernoulli's will result in cross-entropy loss.
- However, the resulting code in the middle of the autoencoder

Learning Deep Generalized Autoencoder Codes

- However, there are no latent variables to integrate or optimize over in a deep generalized autoencoder. The code vector is a deterministic function of the input.
- However, the resulting code in the middle of the a deep generalized autoencoder is still real-valued.
- Unlike in the generalized factor analysis model family, there is no way to obtain other kinds of codes.

Restricted Boltzmann Machines

- RBMs are closely related to the factor analysis model family.
- They are also closely related to a class of models from statistical physics called Ising models or spin glass models that are joint models for binary variables.
- They are also closely related to a class of models studied in statistics called Markov Random Fields, which generalize Ising models and Markov chains.
- In the ML literature, these models are described using connectionist terminology.
- In the classical RBM, $\mathbf{h} \in \{0, 1\}^K$ is a vector of hidden units while $\mathbf{x} \in \{0, 1\}^D$ is a vector of observed values. The model parameterizes $P(\mathbf{X} = \mathbf{x}, \mathbf{H} = \mathbf{h})$.

Restricted Boltzmann Machines

- Instead of being composed of two locally normalized probability distributions (e.g., $P(X|Z)$ and $P(Z)$), they are constructed from a joint *energy function* that requires explicit normalization. These models are also called energy-based models (EBMs).

$$E_W^k(\mathbf{x}, h_k) = -\sum_{d=1}^{D} (W_{dk}^P x_d h_k + W_k^B h_k)$$

$$E_W(\mathbf{x}, \mathbf{h}) = \sum_{k=1}^{K} E_W^k(\mathbf{x}, h_k)$$

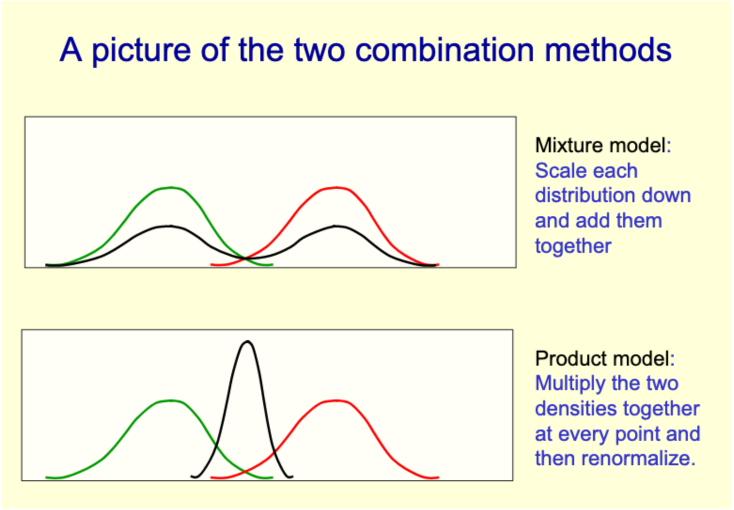
$$P_W(\mathbf{X} = \mathbf{x}, \mathbf{H} = \mathbf{h}) = \frac{\exp\left(-E_W(\mathbf{x}, \mathbf{h})\right)}{\sum_{\mathbf{x}'} \sum_{\mathbf{h}'} \exp\left(-E_W(\mathbf{x}', \mathbf{h}')\right)}$$

Restricted Boltzmann Machines

- We can equivalently write the model as a product of distributions, one for each hidden unit. For this reason, the model is sometimes also called a product of experts (PoE).

$$P_W(\mathbf{X} = \mathbf{x}, \mathbf{H} = \mathbf{h}) = \frac{\prod_{k=1}^{K} \exp(-E_W^k(\mathbf{x}, h_k))}{\sum_{\mathbf{x}'} \sum_{\mathbf{h}'} \prod_{k'=1}^{K} \exp(-E_W^{k'}(\mathbf{x}, h_{k'}))}$$

Products vs Mixtures


A picture of the two combination methods

Mixture model: Scale each distribution down and add them together

Product model: Multiply the two densities together at every point and then renormalize.

Slide from Geoff Hinton, CSC321, Lecture 24

Restricted Boltzmann Machines: Conditionals

Unlike for the factor analysis family, both $P(\mathbf{X}|\mathbf{H}=\mathbf{h})$ and $P(\mathbf{H}|\mathbf{X}=\mathbf{x})$ are fully factorized distributions that are easily computable:

$$P_W(\mathbf{X}=\mathbf{x}|\mathbf{h}) = \prod_{d=1}^{D} P_W(X_d = x_d|\mathbf{h})$$

$$P_W(\mathbf{H}=\mathbf{h}|\mathbf{x}) = \prod_{k=1}^{K} P_W(H_k = h_k|\mathbf{x})$$

Restricted Boltzmann Machines: Conditionals

Unlike for the factor analysis family, both $P(\mathbf{X}|\mathbf{H} = \mathbf{h})$ and $P(\mathbf{H}|\mathbf{X} = \mathbf{x})$ are fully factorized distributions that are easily computable:

$$P_W(X_d = x_d|\mathbf{h}) = \frac{\exp\left(\sum_{k=1}^{K} W_{dk}^P x_d h_k\right)}{1 + \exp\left(\sum_{k=1}^{K} W_{dk}^P h_k\right)}$$

$$P_W(H_k = h_k|\mathbf{x}) = \frac{\exp\left(\sum_{d=1}^{D} W_{dk}^P x_d h_k + W_k^B h_k\right)}{1 + \exp\left(\sum_{d=1}^{D} W_{dk}^P x_d + W_k^B\right)}$$

Restricted Boltzmann Machines: Visible Marginal

- It's also actually possible to sum over all joint settings of the hidden variables analytically to get an expression for $P_W(\mathbf{X})$, but these values are typically not computable:

$$P_W(\mathbf{X}=\mathbf{x}) = \frac{\prod_{k=1}^{K}\left(1+\exp\left(\sum_{d=1}^{D} W^P_{dk}x_d + W^B_k\right)\right)}{\sum_{\mathbf{x}'}\prod_{k=1}^{K}\left(1+\exp\left(\sum_{d=1}^{D} W^P_{dk}x'_d + W^B_k\right)\right)}$$

Restricted Boltzmann Machines: Stochastic MLE

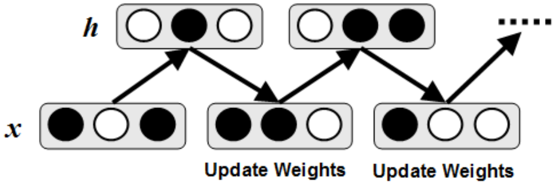
- Again, unlike the FA model family, the RBM family can be learned using marginal likelihood maximization, but sampling is required to efficiently approximate marginal likelihood gradients. The result is a stochastic maximum marginal likelihood algorithm.

$$\frac{\partial \mathcal{L}(\mathcal{D}, \mathbf{W})}{\partial W^P_{dk}} = \frac{1}{N}\sum_{n=1}^{N} x_{dn} P(H_k = 1|\mathbf{x}_n) - \frac{1}{S}\sum_{s=1}^{S} \hat{x}_{ds} P(H_k = 1|\hat{\mathbf{x}}_s)$$

$$\hat{\mathbf{x}}_s \sim P_W(\mathbf{X})$$

Restricted Boltzmann Machines: Stochastic MLE

- The sampling procedure alternates between sampling $\hat{\mathbf{x}}$ and $\hat{\mathbf{h}}$ using the structure shown below. Only the $\hat{\mathbf{x}}$ samples are used for learning.



Restricted Boltzmann Machines: Applications

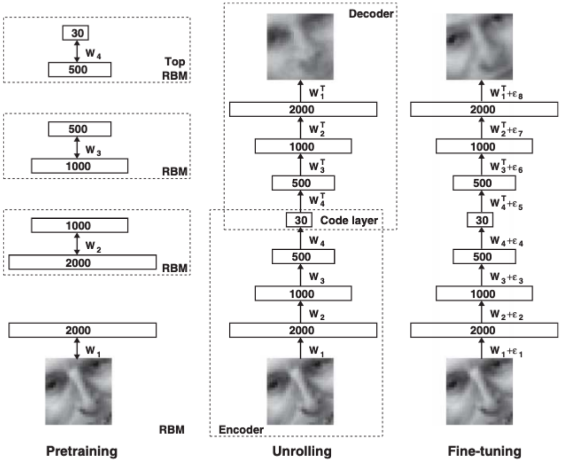


Image from Hinton and Salakhutdinov. *Reducing the Dimensionality ofData with Neural Networks*. Science. 2006.