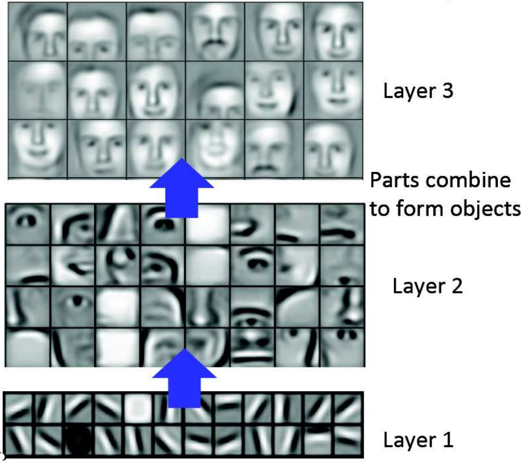
Deep Learning and Representation Learning

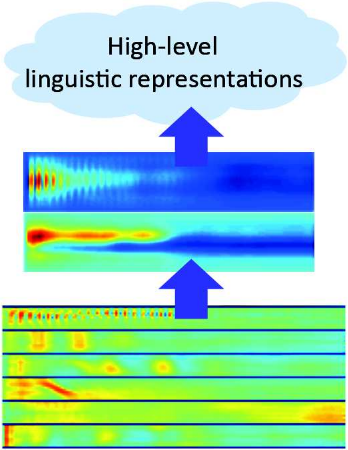
Discussion by: Piyush Rai

(Some figures from Ruslan Salakhutdinov)

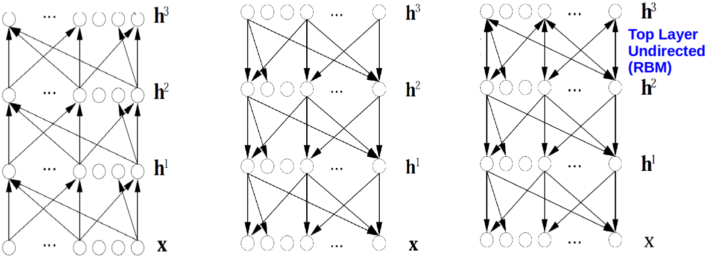
August 01, 2014

Deep Feature Learning





Some Deep Architectures



Deep Neural Net **Sigmoidal Belief Net** **Deep Belief Net**

Undirected graphical model based deep architectures (e.g., Deep Belief Nets):

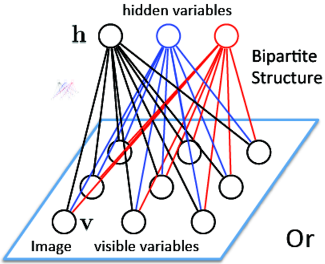
- Usually based on undirected models such as Restricted Boltzmann Machine (RBM) as building blocks
- inference for the hidden variables is easy: $P(\mathbf{h}|\mathbf{x}) = \prod_i P(\mathbf{h}_i|\mathbf{x})$
- it's possible to train the model in a layer-wise fashion. Training not so easy but possible via approximations such as **Contrastive Divergence**

Today

- Restricted Boltzmann Machine and its variants
- Autoencoder and its variants
- Building invariances: Convolutional Neural Networks
- Deep architectures for supervised learning
- Global training of deep architectures

A typical RBM

- Binary visible $\mathbf{v} \in \{0,1\}^D$, binary hidden units $\mathbf{h} \in \{0,1\}^F$



Probability of the joint configuration is given by the Boltzmann distribution:

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\big(-E(\mathbf{v},\mathbf{h};\theta)\big)$$

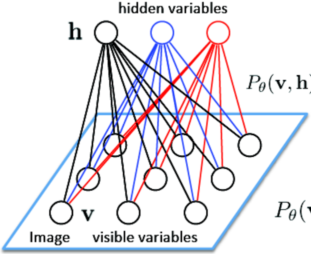
Or

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{i=1}^{D}\sum_{j=1}^{F} \overbrace{W_{ij} v_i h_j}^{\textbf{Pair-wise}} + \sum_{i=1}^{D} \overbrace{v_i b_i}^{} + \sum_{j=1}^{F} \overbrace{h_j a_j}^{\textbf{Unary}} \right)$$

$$\mathcal{Z}(\theta) = \sum_{\mathbf{h},\mathbf{v}} \exp\big(-E(\mathbf{v},\mathbf{h};\theta)\big)$$

RBM for real-valued data

- Real-valued visible units **v**, binary-valued hidden units **h**



$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{i=1}^{D}\sum_{j=1}^{F} \overbrace{W_{ij} h_j \frac{v_i}{\sigma_i}}^{\textbf{Pair-wise}} + \sum_{i=1}^{D} \overbrace{\frac{(v_i - b_i)^2}{2\sigma_i^2}}^{\textbf{Unary}} + \sum_{j=1}^{F} \overbrace{a_j h_j}^{} \right)$$

$$\theta = \{W, a, b\}$$

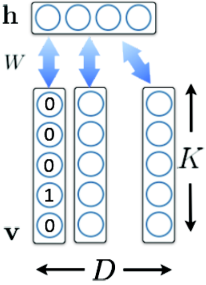
$$P_\theta(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} P_\theta(v_i|\mathbf{h}) = \prod_{i=1}^{D} \mathcal{N}\left(b_i + \sum_{j=1}^{F} W_{ij} h_j, \sigma_i^2\right)$$

Gaussian-Bernoulli RBM:

- Stochastic real-valued visible variables $\mathbf{v} \in \mathbb{R}^D$.
- Stochastic binary hidden variables $\mathbf{h} \in \{0,1\}^F$.
- Bipartite connections.

RBM for word counts

- Count-valued visible units **v**, binary-valued hidden units **h**



$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\overbrace{\sum_{i=1}^{D}\sum_{k=1}^{K}\sum_{j=1}^{F} W_{ij}^{k} v_i^k h_j}^{\textbf{Pair-wise}} + \overbrace{\sum_{i=1}^{D}\sum_{k=1}^{K} v_i^k b_i^k + \sum_{j=1}^{F} h_j a_j}^{\textbf{Unary}} \right)$$

$$\theta = \{W, a, b\}$$

$$P_\theta(v_i^k = 1|\mathbf{h}) = \frac{\exp\left(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k\right)}{\sum_{q=1}^{K} \exp\left(b_i^q + \sum_{j=1}^{F} h_j W_{ij}^q\right)}$$

Replicated Softmax Model: undirected topic model:

- Stochastic 1-of-K visible variables.
- Stochastic binary hidden variables $\mathbf{h} \in \{0,1\}^F$.
- Bipartite connections.

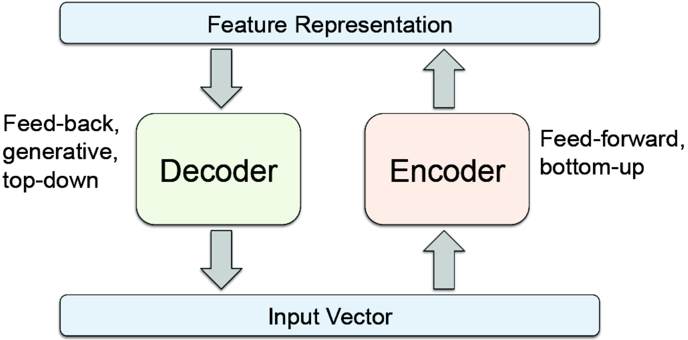
(Salakhutdinov & Hinton, NIPS 2010, Srivastava & Salakhutdinov, NIPS 2012)

Conditional RBM

- Traditional RBM $P_\theta(\mathbf{x}, \mathbf{h})$ has observed variables $\mathbf{x}$, hidden variables $\mathbf{h}$ and parameters $\theta = \{W, \mathbf{b}, \mathbf{c}\}$
- Often, we may have some context variables (or covariates) $\mathbf{z}$
- Conditional RBM $P_\theta(\mathbf{x}, \mathbf{h}|\mathbf{z})$ assumes that the parameters $\theta = f(\mathbf{z}, \omega)$ where ω are the actual "free" parameters
- Example: hidden unit bias $\mathbf{c} = \beta + M\mathbf{z}$; weights W could also depend on the context variables/covariates in some applications

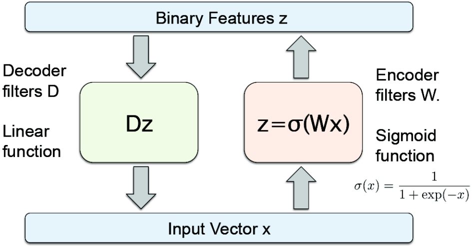
Autoencoder

- Provides a direct parametric mapping from inputs to feature representation
- Often used as a building block in deep architectures (just like RBMs)
- Basic principle: Learns an encoding of the inputs so as to recover well the original input from the encodings



Autoencoder

- Real-valued inputs, binary-valued encodings



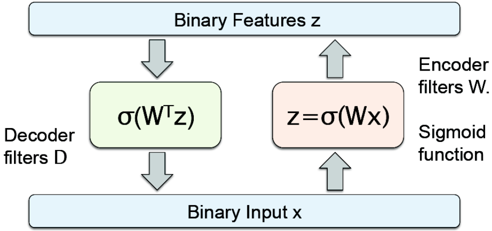
- Sigmoid encoder (parameter matrix W), linear decoder (parameter matrix D), learned via:

$$\arg\min_{D,W} E(D, W) = \sum_{n=1}^{N} ||D\mathbf{z}_n - \mathbf{x}_n||^2 = \sum_{n=1}^{N} ||D\sigma(W\mathbf{x}_n) - \mathbf{x}_n||^2$$

- If encoder is also linear, then autoencoder is equivalent to PCA

Autoencoder

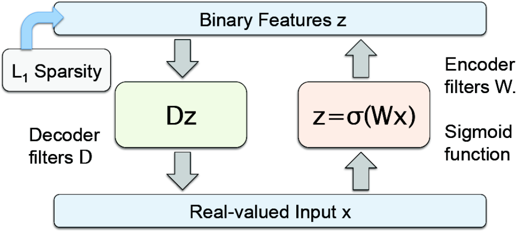
- Binary-valued inputs, binary-valued encodings



- Similar to an RBM
- Need constraints to avoid an identity mapping (e.g., by imposing sparsity on the encodings or by "corrupting" the inputs)

Sparse Autoencoders

- Sparse binary encodings. Can impose L_1 penalty on the codes

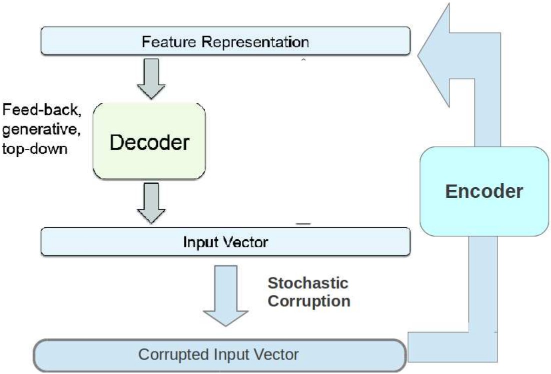


- Predictive Sparse Decomposition (learns an explicit mapping from the input to the encoding)

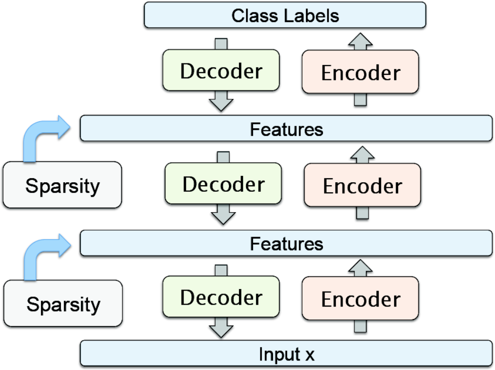
$$\arg\min_{D,W,\mathbf{z}} \sum_{n=1}^{N} ||D\mathbf{z}_n - \mathbf{x}_n||^2 + \lambda|\mathbf{z}_n| + ||\sigma(W\mathbf{x}_n) - \mathbf{z}_n||^2$$

Denoising Autoencoders

- Idea: introduce stochastic corruption to the input; e.g.:
  - Hide some features
  - Add gaussian noise



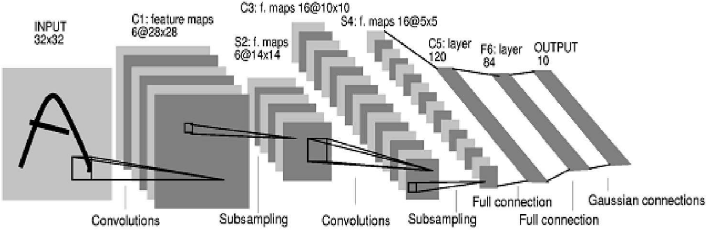
Stacked Autoencoders

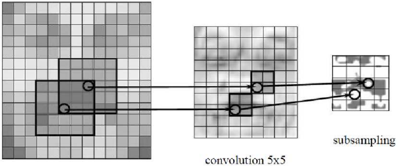


- Can be learned in a greedy layer-wise fashion

Building Invariances: Convolutional Neural Network

- Exploits topological structure in the data via three key ideas: local receptive field, shared weights, and spatial or temporal sub-sampling
- Ensures some degree of shift/scale/distortion invariance in the learned representation



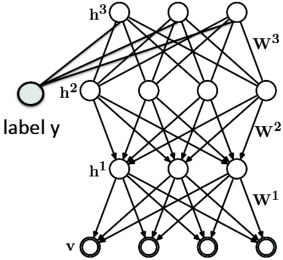


Building Invariances: Other ways

- Generating transformed examples
  - .. via introducing random deformations that don't change the target label
- Temporal coherence and slow feature analysis

Supervised Learning with Deep Architectures

- Consider a Deep Belief Net trained in a supervised fashion
- Given labels y, train on the joint log-likelihood of inputs and their labels $\log P(\mathbf{x}, y)$
- Usually a two-step procedure is used
  1. Unsupervised pre-training of DBN without labels
  2. Fine-tuning the parameters by maximizing the conditional log-likelihood $\log P(y|\mathbf{x})$



Global Training of Deep Architectures?

- Early successes were mainly attributed to layer-wise pre-training
- Some recent successes with global training of deep architectures
  - Lots of labeled data, lots of tasks, artificially transformed examples
  - Proper initialization, efficient training (e.g., using GPUs), adaptive step-sizes
  - Choice of nonlinearities
- Unsupervised layer-wise pre-training seems to act like a regularizer
  - .. less of a necessity when labeled training data is abundant

Other extensions of deep architectures

- Hierarchical Deep Models: Deep + (NP)Bayes
  - Putting an HDP over the states of the top layer of a deep model
  - Allows sharing of statistical strengths across categories/classes and/or helps generalize to novel/unseen categies by transfer learning
- Deep models for multimodal data (text and images)

Thanks! Questions?