Neural networks

Deep learning - unsupervised pre-training
Topics: why training is hard

- Depending on the problem, one or the other situation will tend to prevail

- If first hypothesis (underfitting): use better optimization
  - this is an active area of research

- If second hypothesis (overfitting): use better regularization
  - unsupervised learning
  - stochastic «dropout» training
**Topics:** unsupervised pre-training

- **Solution:** initialize hidden layers using unsupervised learning
  - force network to represent latent structure of input distribution
  - encourage hidden layers to encode that structure

![character image](image1.png)

![random image](image2.png)
Topics: unsupervised pre-training

- Solution: initialize hidden layers using unsupervised learning
  - force network to represent latent structure of input distribution

- encourage hidden layers to encode that structure

Why is one a character and the other is not?
Topics: unsupervised pre-training

- Solution: initialize hidden layers using unsupervised learning
  - this is a harder task than supervised learning (classification)
  - hence we expect less overfitting

Why is one a character and the other is not?

character image

random image
Topics: unsupervised pre-training

- We will use a greedy, layer-wise procedure
  - train one layer at a time, from first to last, with unsupervised criterion
  - fix the parameters of previous hidden layers
  - previous layers viewed as feature extraction
Topics: unsupervised pre-training

• We call this procedure unsupervised pre-training
  
  ‣ first layer: find hidden unit features that are more common in training inputs than in random inputs
  
  ‣ second layer: find combinations of hidden unit features that are more common than random hidden unit features
  
  ‣ third layer: find combinations of combinations of ...
  
  ‣ etc.

• Pre-training initializes the parameters in a region such that the near local optima overfit less the data
**Topics:** fine-tuning

- Once all layers are pre-trained
  - add output layer
  - train the whole network using supervised learning
- Supervised learning is performed as in a regular feed-forward network
  - forward propagation, backpropagation and update
- We call this last phase fine-tuning
  - all parameters are “tuned” for the supervised task at hand
  - representation is adjusted to be more discriminative
DEEP LEARNING

Topics: pseudocode

• for \( l = 1 \) to \( L \)
  
  ‣ build unsupervised training set (with \( h^{(0)}(x) = x \ )):
  
  \[
  \mathcal{D} = \left\{ h^{(l-1)}(x^{(t)}) \right\}_{t=1}^{T}
  \]
  
  ‣ train “greedy module” (RBM, autoencoder) on \( \mathcal{D} \)

  ‣ use hidden layer weights and biases of greedy module to initialize the deep network parameters \( W^{(l)}, b^{(l)} \)

• Initialize \( W^{(L+1)}, b^{(L+1)} \) randomly (as usual)

• Train the whole neural network using (supervised) stochastic gradient descent (with backprop)
DEEP LEARNING

Topics: pseudocode

- for \( l = 1 \) to \( L \)
  - build unsupervised training set \((\text{with } h^{(0)}(x) = x)\):
    \[
    D = \left\{ h^{(l-1)}(x^{(t)}) \right\}_{t=1}^T
    \]
  - train “greedy module” (RBM, autoencoder) on \( D \)
  - use hidden layer weights and biases of greedy module to initialize the deep network parameters \( W^{(l)}, b^{(l)} \)
- Initialize \( W^{(L+1)}, b^{(L+1)} \) randomly (as usual)
- Train the whole neural network using (supervised) stochastic gradient descent (with backprop)
Topics: pseudocode

- for $l=1$ to $L$
  - build unsupervised training set (with $h^{(0)}(x) = x$):
    $$\mathcal{D} = \left\{ h^{(l-1)}(x^{(t)}) \right\}_{t=1}^{T}$$
  - train “greedy module” (RBM, autoencoder) on $\mathcal{D}$
  - use hidden layer weights and biases of greedy module to initialize the deep network parameters $W^{(l)}, b^{(l)}$

- Initialize $W^{(L+1)}, b^{(L+1)}$ randomly (as usual)
- Train the whole neural network using (supervised) stochastic gradient descent (with backprop)
WHAT KIND OF UNSUPERVISED LEARNING?

**Topics:** stacked RBMs, stacked autoencoders

• Stacked restricted Boltzmann machines:
  ‣ Hinton, Teh and Osindero suggested this procedure with RBMs
    - A fast learning algorithm for deep belief nets.
      Hinton, Teh, Osindero., 2006.
    - To recognize shapes, first learn to generate images.

• Stacked autoencoders:
  ‣ Bengio, Lamblin, Popovici and Larochelle studied and generalized the procedure to autoencoders
  ‣ Ranzato, Poultney, Chopra and LeCun also generalized it to sparse autoencoders
    - Efficient Learning of Sparse Representations with an Energy-Based Model.
WHAT KIND OF UNSUPERVISED LEARNING?

Topics: stacked RBMs, stacked autoencoders

• Stacked denoising autoencoders:
  ‣ proposed by Vincent, Larochelle, Bengio and Manzagol

• And more:
  ‣ stacked semi-supervised embeddings
  ‣ stacked kernel PCA
  ‣ stacked independent subspace analysis
    - Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis, Le, Zou, Yeung and Ng, 2011.