Dropout

Sargur N. Srihari
srihari@buffalo.edu

This is part of lecture slides on Deep Learning:
http://www.cedar.buffalo.edu/~srihari/CSE676

Regularization Strategies

1. Parameter Norm Penalties
2. Norm Penalties as Constrained Optimization
3. Regularization and Under-constrained Problems
4. Data Set Augmentation
5. Noise Robustness
6. Semi-supervised learning
7. Multi-task learning
8. Early Stopping
9. Parameter tying and parameter sharing
10. Sparse representations
11. Bagging and other ensemble methods
12. Dropout
13. Adversarial training
14. Tangent methods

Topics in Dropout

- What is dropout?
- Dropout as an ensemble method
- Mask for dropout training
- Bagging vs Dropout
- Prediction intractability

Overfitting in Deep Neural Nets

- Deep nets have many non-linear hidden layers
  - Making them very expressive to learn complicated relationships between inputs and outputs
  - But with limited training data, many complicated relationships will be the result of training noise
    - So they will exist in the training set and not in test set even if drawn from same distribution
- Many methods developed to reduce overfitting
  - Early stopping with a validation set
  - Weight penalties (L^1 and L^2 regularization)
  - Soft weight sharing

Regularization with unlimited computation

- Best way to regularize a fixed size model is:
  - Average the predictions of all possible settings of the parameters,
  - Weighting each setting with the posterior probability given the training data
    - This would be the Bayesian approach
- Dropout does this using considerably less computation
  - By approximating an equally weighted geometric mean of the predictions of an exponential number of learned models that share parameters

Dropout is a bagging method

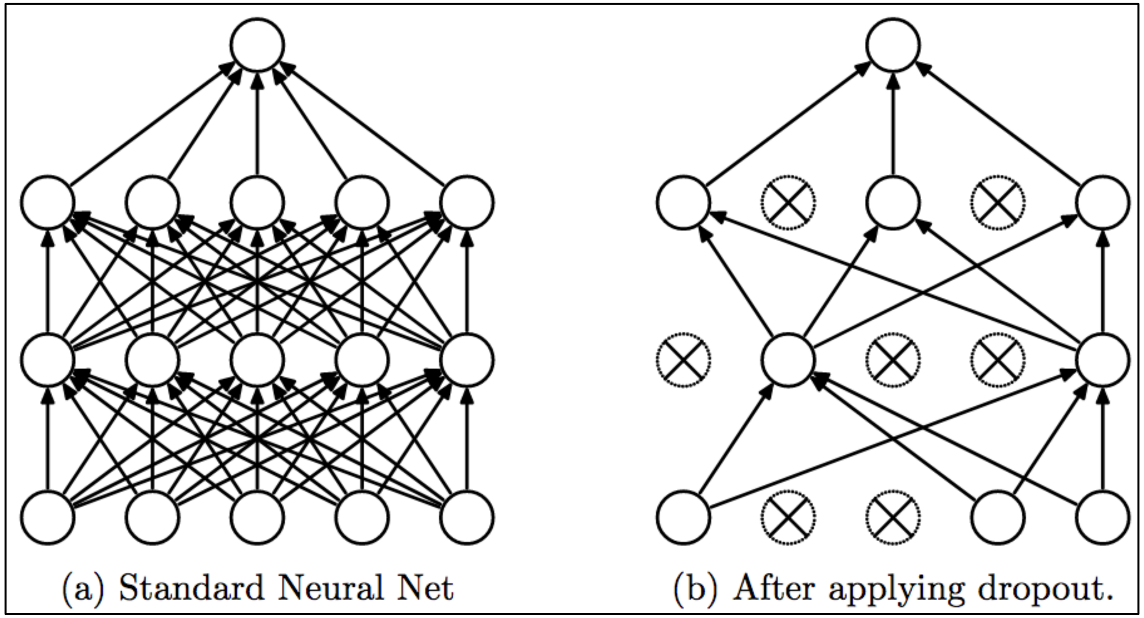
- Bagging is a method of averaging over several models to improve generalization
- Impractical to train many neural networks since it is expensive in time and memory
  - Dropout makes it practical to apply bagging to very many large neural networks
    - It is a method of bagging applied to neural networks
- Dropout is an inexpensive but powerful method of regularizing a broad family of models

Removing units creates networks

- Dropout trains an ensemble of all subnetworks
  - Subnetworks formed by removing non-output units from an underlying base network
- We can effectively remove units by multiplying its output value by zero
  - For networks based on performing a series of affine transformations or on-linearities
  - Needs some modification for radial basis functions based on difference between unit state and a reference value

Dropout Neural Net

- A simple way to prevent neural net overfitting



Drop hidden and visible units from net, i.e., temporarily remove it from the network with all input/output connections. Choice of units to drop is random, determined by a probability p, chosen by a validation set, or equal to 0.5

(a) A standard neural net with two hidden layers

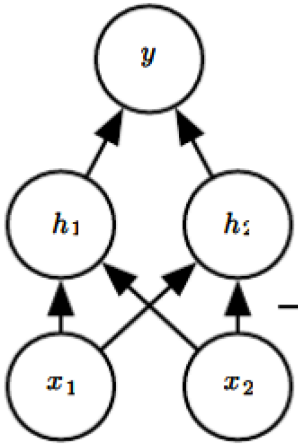
(b) A thinned net produced by applying dropout, crossed units have been dropped

Dropout as bagging

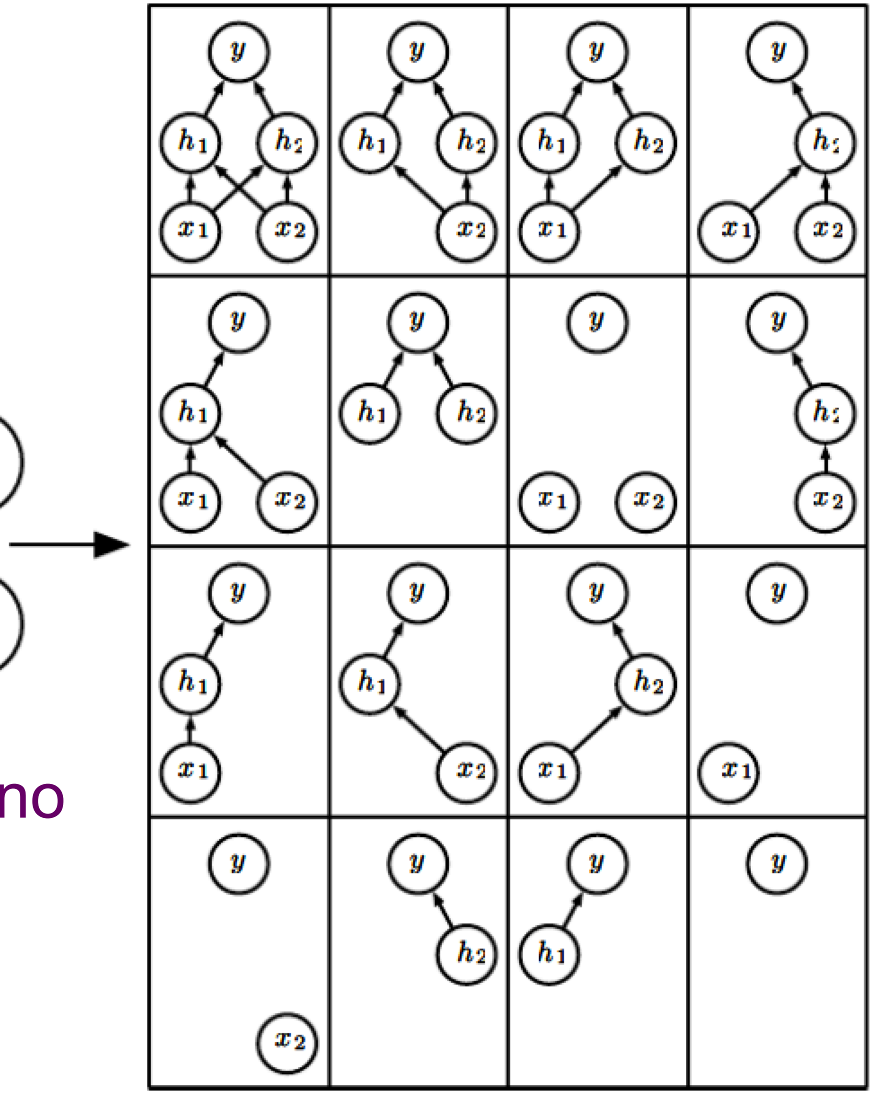
- In bagging we define k different models, construct k different data sets by sampling from the dataset with replacement, and train model i on dataset i
- Dropout aims to approximate this process, but with an exponentially large no. of neural networks

Dropout as an ensemble method

- Remove non-output units from base network.
- Remaining 4 units yield 16 networks



Base network


Ensemble of subnetworks

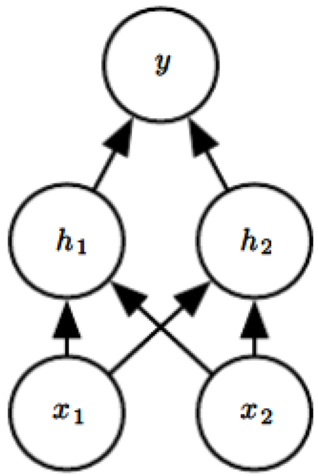
- Here many networks have no path from input to output
- Problem insignificant with large networks

Mask for dropout training

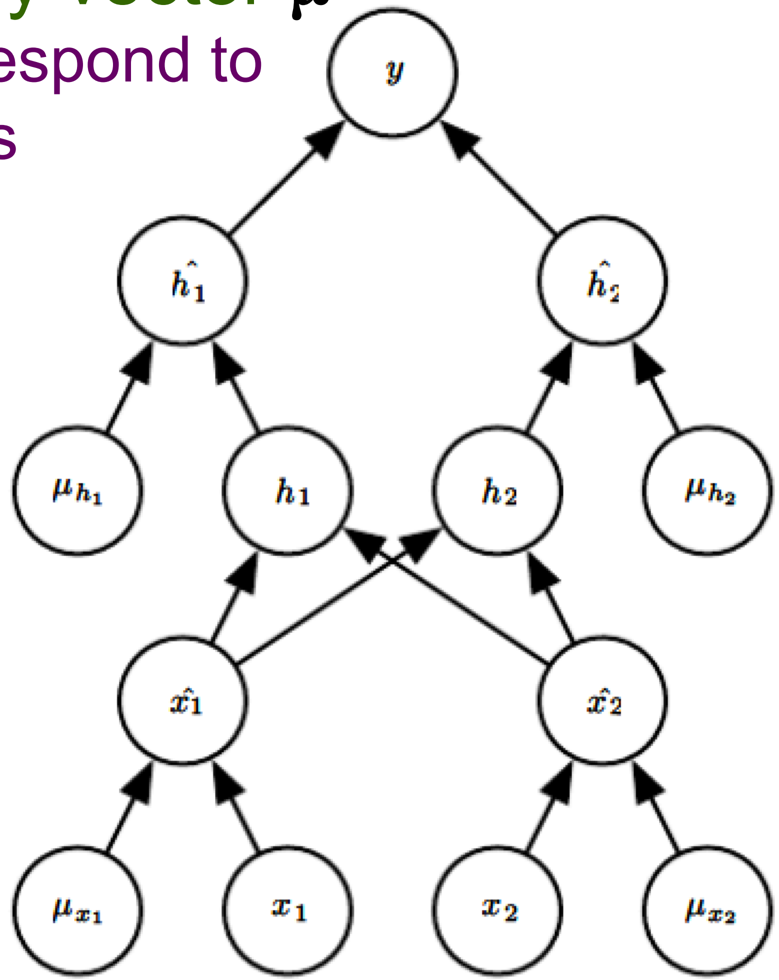
- To train with dropout we use minibatch based learning algorithm that takes small steps such as SGD
- At each step randomly sample a binary mask
  - Probability of including a unit is a hyperparameter
    - 0.5 for hidden units and 0.8 for input units
- We run forward & backward propagation as usual

Forward Propagation with dropout

Feed-forward network



- Network with binary vector $\boldsymbol{\mu}$ whose elements correspond to input and hidden units
- Elements of $\boldsymbol{\mu}$
- With probability of 1 being a hyperparameter
  - 0.5 for hidden
  - 0.8 for input
- Each unit is
  - Multiplied by corresponding mask



- Forward prop as usual
- Equivalent to randomly selecting one of the subnetworks of previous slide

Formal description of dropout

- Suppose that mask vector $\boldsymbol{\mu}$ specifies which units to include
- Cost of the model is specified by $J(\boldsymbol{\theta},\boldsymbol{\mu})$
- Drop training consists of minimizing $E_{\boldsymbol{\mu}}(J(\boldsymbol{\theta},\boldsymbol{\mu}))$
- Expected value contains exponential no. of terms
- We can get an unbiased estimate of its gradient by sampling values of $\boldsymbol{\mu}$

Bagging training vs Dropout training

- Dropout training not same as bagging training
  - In bagging, the models are all independent
  - In dropout, models share parameters
    - Models inherit subsets of parameters from parent network
    - Parameter sharing allows an exponential no. of models with a tractable amount of memory
- In bagging each model is trained to convergence on its respective training set
  - In dropout, most models are not explicitly trained
    - Fraction of subnetworks are trained for a single step
    - Parameter sharing allows good parameter settings

Prediction: Bagging vs. Dropout

- Bagging:
  - Ensemble accumulates votes of members
  - Process is referred to as inference
    - Assume model needs to output a probability distribution
    - In bagging, model i produces $p^{(i)}(y|\boldsymbol{x})$
    - Prediction of ensemble is the mean $\boxed{\frac{1}{k}\sum_{i=1}^{k} p^{(i)}(y \mid \boldsymbol{x})}$
- Dropout:
  - Submodel defined by mask vector $\boldsymbol{\mu}$ defines a probability distribution $p(y|\boldsymbol{x},\boldsymbol{\mu})$
  - Arithmetic mean over all masks is $\boxed{\sum_{\mu} p(y \mid \boldsymbol{x}, \mu)}$
    - Where $p(\boldsymbol{\mu})$ is the distribution used to sample $\boldsymbol{\mu}$ at training time

Intractability of prediction

- Dropout prediction is $\boxed{\sum_{\mu} p(y \mid \boldsymbol{x}, \mu)}$
- It is intractable to evaluate due to an exponential no. of terms
- We can approximate inference using sampling
  - By averaging together the output from many masks
    - 10-20 masks are sufficient for good performance
- Even better approach, at the cost of a single forward propagation:
  - use geometric mean rather than arithmetic mean of the ensemble member's predicted distributions