Learning as Optimization

Sargur Srihari
srihari@cedar.buffalo.edu

Topics in Learning as Optimization

- Evaluation of Learned Model
- Empirical Risk and Overfitting
  - Bias vs. Variance Trade-off
  - Design and Evaluation of Learning Procedures
  - Goodness of Fit
  - PAC bounds
- Discriminative versus Generative Training
- Learning Tasks
  - Model Constraints
  - Data Observability
  - Taxonomy of Learning Tasks

Optimization Viewpoint

- We have:
  1. Hypothesis Space
     - Set of candidate models

     e.g., set of all BNs or MNs given a set of variables
  2. Objective Function
     - Criterion for quantifying preference over models
- Learning Task
  - Find a high-scoring model within model class
- Learning as optimization is predominant approach

Criteria for Optimization

- Numerical Criteria (Loss functions) we optimize
  1. Density Estimation
     - Relative Entropy or K-L Divergence
       - Expected value of log-difference
       - Equivalent to Empirical Risk over instances $\mathcal{D}$
         - » Called Empirical log-loss

$$D(P^* \| P) = E_{\xi \sim P^*}\left[\log\left(\frac{P^*(\xi)}{P(\xi)}\right)\right]$$

$$-\frac{1}{|\mathcal{D}|}\sum_{m=1}^{M}\log P(\xi[m]:\mathcal{M})$$

  2. Classification
     - Classification error: *0/1* loss
     - Hamming loss
     - Conditional log-likelihood

$$E_{(x,y)\sim P}[I\{h_P(x) \neq y\}]$$

$$E_{(x,y)\sim P^*}[\log P(y \mid x)]$$

- Can view learning as optimization

Why discuss Optimization?

- Different choices of Objective Functions
  - Have ramification to results of learning procedures
- Has implications to further discussions on learning

Empirical Distribution & Log-loss

- We have an unknown distribution P^*
- Empirical Distribution

$$\hat{P}_D(A) = \frac{1}{M}\sum_m I\{\xi[m] \in A\}$$

Probability of Event A is the fraction of samples that satisfy A

where $\xi[1], \xi[2]\ldots$ is a sequence of iid samples from P^*

  - For a sufficiently large training set P_D, will be quite close to P^*

$$\boxed{\lim_{M\to\infty} \hat{P}_{D_M}(A) = P^*(A)}$$

- Consider empirical log-loss $-\frac{1}{|\mathcal{D}|}\sum_{m=1}^{M} \log P(\xi[m] : \mathcal{M})$
  - Can be shown that distribution that maximizes likelihood of data set D is the empirical distribution

Data Requirements

- How many data samples are needed?
- Consider two cases
  1. 100 binary random variables
  2. Bayesian network where a node has k parents
- In both we will see that the hypothesis space is too large

Data Requirement with Binary Variables

- Consider 100 binary variables
  - 2^{100} possible joint assignments
- $\mathcal{D}$ has $1{,}000$ instances (most likely distinct)
  - probability of 0.001 to each assignment
  - 0 to remaining 2^{100}-1000
- Example is extreme, but phenomenon is general

Data Requirement with Bayesian network

- M^* is a Bayesian network with a variable 'Fever'
  - "$Fever$" has many parents (diseases) $X_1,..X_k$
  - In a table CPD
    - parameters grow exponentially with no. of parents k
  - Unlikely $\mathcal{D}$ has cases representing all parent instantiations
    - i.e., all combinations of diseases $X_1,..X_k$
    - Results in very poor CPDs
- Data requirement: exponential with BN connectivity
  - No of samples needed grows linearly with no. of parameters
  - No. of parameters grows exponentially with network connectivity

Problem with using Empirical Risk

- Learned model tends to overfit to training data
- But our goal is to answer queries about unseen data
  - Not data samples we have seen
    - New patients in disease example
    - New unsegmented images
- Need to generalize well
- This leads to a trade-off
  - Should not overfit but generalize well

Overfitting and Bias-Variance Trade-off

- If hypothesis space is very limited (few bins), unable to represent true P^*
  - Even with unlimited data
  - Called Bias
- If hypothesis space is large (expensive)
  - Examples: binary data set and disease BN
  - If data set is small even random fluctuations in data set have an effect
  - Results in large variance over different data sets

Implication to Model Structure

- Not allow too rich a class of models
- With limited data
  - Error due to variance $\geq$ Error due to bias
  - Restrict to models too simple to correctly encode P^*
  - Ability to estimate parameters reliably

    compensates for
  - Error from incorrect structural assumptions
- Restricting models →learn important edges
  - Distance to P^* is better
- Combining approaches
  - Hard constraint over model class
  - Optimization objective that leads us away from over-fitting

Generalization Performance

- Have not discussed
  - How well model generalizes
    - Performance on unseen data sets
  - Design of hypothesis space to reduce over-fitting
- Next:
  1. Basic experimental protocol

     in design/evaluation of learning procedure
  2. Theoretical framework for appropriate complexity

Relative and Absolute Learning

1. Relative
   - Compare two or more alternatives
     - Different hypothesis spaces
     - Different training objectives
   - Methods: *Holdout* and *Cross-Validation*
     - Training, Validation, Testing
2. Absolute
   - Whether model captures true distribution
   - Method: *Goodness of Fit*

Evaluating Generalization Performance

- Performance on unseen data
- Hold-out method
  - Partition data into two sets: $\mathcal{D}_{train}$, $\mathcal{D}_{test}$
  - Use randomized procedure to decide partition
  - Learn modes using $\mathcal{D}_{train}$ and an objective function
  - Measure performance using $\mathcal{D}_{test}$ and loss function
  - Provides unbiased estimate on new samples
- Performance on $\mathcal{D}_{train}$ usually better than $\mathcal{D}_{test}$
  - If difference is large then we are overfitting
    - Use less expressive model
    - or method to discourage overfitting

k-fold Cross Validation

- If data set is small
- Divide into k sets
- Test on one while training on the remaining
- For writing a paper on learning
  - Report results using cross-validation
- In practice
  - Train on all data

Selecting a Learning Procedure

- Validation set
- If we are testing among a several procedures
  - Using the same testing set to select the procedure yields an over-optimistic result
    - Since selection is optimized to $\mathcal{D}_{test}$
- Use three sets

$\mathcal{D}_{train}$: used to learn the model

$\mathcal{D}_{validate}$: evaluate different variants of training

$\mathcal{D}_{test}$: evaluate final performance

Goodness of Fit

- Whether learned model captures everything about the distribution
  - Harder to answer
- After determining the parameters we have a hypothesis about distribution
  - Now ask whether data behave as if they were generated from this distribution

Implementation of Goodness of Fit

- Let f be a property of a data set
  - E.g., empirical log-loss
- Evaluate $f(\mathcal{D}_{train})$
- From learned $\mathcal{M}$ generate new data sets $\mathcal{D}$
- If $f(\mathcal{D}_{train})$ deviates significantly from distribution of $f(\mathcal{D})$
  - Reject hypothesis that $\mathcal{D}_{train}$ was generated from $\mathcal{M}$

Goodness of Fit: Choosing f

- Natural choice: Empirical log-loss in data set

Expected Loss

$$E_D[loss(\xi : M)] = \frac{1}{|\mathcal{D}|}\sum_{\xi \in D} loss(\xi : \mathcal{M}) \qquad \frac{1}{|\mathcal{D}|}\sum_{m=1}^{M} \log P(\xi[m] : \mathcal{M})$$

  - Test if log-loss for $\mathcal{D}_{train}$ differs significantly from expected empirical log-loss (on $\mathcal{D}$ sampled from $\mathcal{M}$)
    - Expected value is simply the entropy of $\mathcal{M}$
      - Entropy of BN can be computed efficiently
- To check for significance
  - Consider tail distribution of log-loss– more involved
  - Approximation:
    - variance of log-loss as function of M
  - Alternatively: generate large no of data sets $\mathcal{D}$ of size M from model and estimate distribution over $E_D[loss(\xi:\mathcal{M})]$

Probably Approximately Correct

- Given target loss function we can estimate empirical risk on training set $\mathcal{D}_{train}$
- Because of over-fitting to $\mathcal{D}_{train}$
  - may not reflect performance on new data $\mathcal{D}_{new}$
- Are the two performances ($\mathcal{D}_{train}$ & $\mathcal{D}_{new}$) related?
  - Does low training loss imply low expected loss?
- We cannot guarantee with certainty quality of learned model
  - Data set $\mathcal{D}$ is sampled stochastically from P^*
  - Sample may be unrepresentative
    - Probability is low but not zero

PAC Bound

- Assume relative entropy loss function
- P^*_M :distribution over data sets $\mathcal{D}$ of size M
  - sampled from P^*
- Learning Procedure L given $\mathcal{D}$ returns model $\mathcal{M}_{L(D)}$
- We want to prove results of the form

Let ε be our approximation parameter

$\partial > 0$ our confidence parameter

For M large enough we have

$$P^*_M(\{\mathcal{D} : D(P^* \| P_{M_{L(\mathcal{D})}}) \leq \varepsilon\}) \geq 1 - \partial$$

No of samples M needed to reach bound is called is called sample complexity

Type of result is called a PAC bound

  - i.e., for sufficiently large M for most data sets of size M sampled from P^*, learning procedure L applied to $\mathcal{D}$

    will learn a close approximation to P^*

PAC Bound in Practice

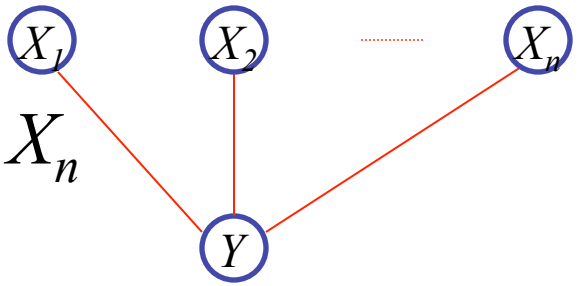
- Can only be obtained if hypothesis space contains the model to correctly represent P^*
- Many cases hypothesis space may not contain P^*
  - Cannot expect to learn a model whose relative entropy to P^* is guaranteed to be low
- Best we can hope is to get a model whose error is at most ε worse than the lowest error found in hypothesis space
  - Expected loss beyond minimum possible error is called excess risk

Discriminative vs Generative Training

- We assumed that goal is to learn model $\mathcal{M}$ to be a good approximation for P^*
- Often we want the model to perform well on a given task
  - E.g., predicting Y from X
- Generative training objective:
  - To get $\mathcal{M}$ close to $P^*(Y,X)$
- Discriminative Training objective:
  - To get $P(Y|X)$ close to $P^*(Y|X)$
- Same model class trained in these two ways can get different results

Example of Generative/Discriminative Training

- Markov network with
  - Target variable Y, Features $X_1,..X_n$
  - $\Phi_i(X_i,Y)$, $\Phi_0(Y)$



- If trained generatively to fit $P(Y,X_1,..X_n)$ we are learning a naïve Markov model
  - Which is equivalent to Naïve Bayes
    - Since network is singly connected
    - Pairwise clique potentials look like CPDs

$$P(Y,X_1,..X_k) = P(Y)\prod_{i=1}^{k} P(X_i \mid Y)$$

- If trained discriminatively, to fit $P(Y|X_1,..X_n)$, we are learning the logistic regression model

$$P(Y=1 \mid x_1,..x_k) = sigmoid\left\{w_0 + \sum_{i=1}^{k} w_i x_i\right\}$$

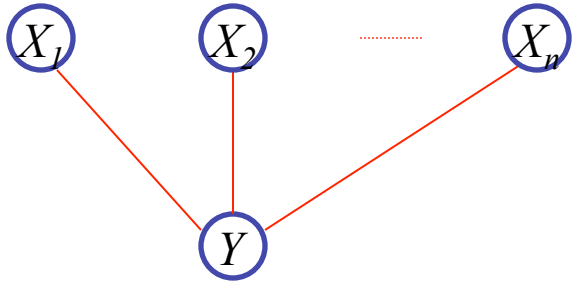
Discrim. Training Disadvantages

- Generative model still useful for prediction
  - E.g., Naïve Bayes used for classification
- Discriminative model trained for prediction $P(Y|X)$ does not encode distribution over X
  - No conclusion about these variables
- Discriminative training less appealing for BNs
  - DT of naïve Bayes gives pairwise potentials locally normalized CPDs
- Discriminative training mostly used for CRFs

Generative Models have higher Bias

- Generative models make more assumptions about form of distribution
  - Encode independences about feature variables X
  - Discriminative models make independence assumptions only about Y and their dependency on X
- Generative model defines $P(Y,X)$
  - Thereby induces $P(Y|X)$ and $P(X)$ using same overall model
    - Thus training needs to fit both well
  - Discriminative model needs a good fit only for $P(Y|X)$
- With limited sets of data generative model less likely to over-fit
  - As data set grows discriminative model less likely to be affected by model assumptions

Example: OCR problem



Y= character label $(A, B \ldots)$

X_i are pixels with 8-bit values $(1,..256)$

- Generatively train naïve Markov
  - Naïve Markov (or Bayes) learns distribution over 256 values given each label $A,B,\ldots$
  - Each estimated independently
    - Low-dimensional estimation problem

Pixel independence Is false assumption

- Discriminatively train logistic regression
  - Jointly optimizes 26 x 256 parameters
    - Much higher dimensional estimation problem

$$P(Y=1 \mid x_1,..x_k) = sigmoid\left\{w_0 + \sum_{i=1}^{k} w_i x_i\right\}$$

Pixel correlation accounted for

- For sparse data naïve Bayes is better
- Discriminative Classifier can use much richer features
  - Edges, image patch centered at pixel which use same pixels

Learning Tasks

- Different Variants of the learning task
- Vary along three axes:
- Two types of Input to learning procedure
  1. Prior knowledge about constraints about $\mathcal{M}$
     - Know graph structure, learn parameters
     - Learn both structure and parameters
     - Not know full set of variables over which P^* is defined
  2. Data $\mathcal{D} = \{d[1],..d[M]\}$ which are i.i.d. samples of P^*
     - Fully observed
     - Partially observed
       - E.g., Patient records: not all tests performed, Disease not fully certain
     - Hidden variables
       - Their inclusion can simplify structure, e.g., genetic susceptibility to a disease
- Output is model $\mathcal{M}$
  3. May include structure and parameters
- Define a Taxonomy of Learning Tasks

Parameter Estimation for Known Structure

Numerical Optimization

- Numbers are difficult to elicit from people
- Parameter Estimation is basis for more advanced scenarios

- Bayesian networks with complete data
  - Parameter estimation is easily solved
  - Possible closed-form solution
- Markov networks
  - Global partition function induces entanglements of parameters
  - For fixed structure problems
    - Optimization problem is convex
      - Iterative numerical optimization
      - Each step requires inference: expensive

Convex minimization :
(i) local minimum is global minimum.
(ii) set of all (global) minima is convex.
(iii)For *strictly* convex minimum is unique.

Approaches to Structure Learning

- Constraint-based Structure Learning
- Score-based Structure Learning
- Bayesian Model Averaging

Constraint-based Structure Learning

- View Bayesian network as a representation of independencies
- Sensitive to failures of individual independence tests
- If one test returns a wrong answer it misleads the network construction procedure

Score-based Structure Learning

- View a BN as specifying a statistical model
- Learning is a model selection problem
  - Hypothesis space is superexponential ($2^{**}O(n^2)$)
    - Situation is worse for Markov nets since cliques can be of size greater than two
  - Each network structure is given a score
    - Optimize to find highest score
- Search problem may not have an elegant and efficient solution

Bayesian Model Averaging

- Generates an ensemble of possible structures
- Average the prediction of all possible structures
- Due to immense number of structures, approximations are needed

Dealing with Incomplete Data

- Multiple hypotheses regarding values of unobserved variables lead to
- Combinatorial range of alternative models
- Induce a non-convex, multimodal optimization problem in parameter space
- Known algorithms work iteratively
  - Using EM style algorithms