Part II: Basic Concept for Unsupervised Learning

- Learning Model from Partially Observable Features
  - How to train a model in which not all the values about its features adopted are known in the training set?
- What kind of Parameters are Learnable?
  - Given sufficient training data, what kind of parameters can be learned?
- Maximum Likelihood Estimation (MLE)
  - How to obtain those parameter under unsupervised learning ?
- Performance Issues
  - Mismatch between the criteria of maximum likelihood and minimum error rate, and between the performance of the training set and the testing set
- Performance versus Corpus Size & Model Complexity
  - General trend for the performance we can obtain

Learning from Partially Observable Features (I)

- Incomplete Data Space:
  - All observable features in the training set are in the *incomplete* data space.
  - The incomplete data space only contains the partial information about those features that will be adopted in the model.
- Complete Data Space:
  - All features that will be adopted in the model are in the *complete* data space.
  - The feature vector in the incomplete data space is mapped from the complete data space (with usually a many-to-one relationship).
  - In many cases, people have to infer parameters of complete data space from the sample space of incomplete data.

Learning from Partially Observable Features (II)

- Unsupervised-learning: learning model in the complete data space through the features in the incomplete data space
  - Under supervised-learning, all the features involved in the model can be directly observed in the training data
    - For example, both “Words” and their associated “Tags” can be seen in the annotated corpus for the statistical tri-gram tagging model
  - Under unsupervised-learning, not all the feature values are known in the training data
    - Only “Words”, not their associated “Tags”, can be seen in the un-annotated corpus for the above statistical tri-gram tagging model
  - The feature vector in the complete data space contains all the feature elements adopted in the model, although part of them are missing from the training data

Learning from Partially Observable Features (III)

- Example [Dempster 77]
  - 197 animals are multinomially distributed into 4 observable categories (in the incomplete data space); however, they should be divided into 5 categories according to their true genetic model (in the complete data space). Please see Part III for details.
  - Incomplete Data Space: [y1, y2, y3, y4]
  - $y=(y_1, y_2, y_3, y_4)=(125,18,20,34)$.
  - cell probabilities: $(1/2+1/4\pi, 1/4(1-\pi), 1/4(1-\pi), 1/4\pi)$ [with one parameter π]

$$g(y|\pi)=\frac{(y_1+y_2+y_3+y_4)!}{y_1!y_2!y_3!y_4!y_5!}\left(\frac{1}{2}+\frac{\pi}{4}\right)^{y_1}\left(\frac{4}{4}-\frac{1}{4}\pi\right)^{y_2}\left(\frac{1}{4}-\frac{1}{4}\pi\right)^{y_3}\left(\frac{1}{4}\pi\right)^{y_4}$$

Learning from Partially Observable Features (IV)

- Example [Dempster 77] (cont.)
  - Complete Data Space: [X1, X2, X3, X4, X5]
  - cell probabilities: (1/2, 1/4π, 1/4(1- π), 1/4(1- π), 1/4π) [with one parameter π]
  - $y_1 = x_1+x_2$, e.g., (125, 0), or (124,1), or … Note: (X1, X2) => Y1: many-to-one mapping. $y_2 = x_3$, $y_3 = x_4$, $y_4 = x_5$.

$$f(x|\pi) = = \frac{(x_1+x_2+x_3+x_4+x_5)!}{x_1!x_2!x_3!x_4!x_5!}\left(\frac{1}{2}\right)^{x_1}\left(\frac{1}{4}\pi\right)^{x_2}\left(\frac{1}{4}-\frac{1}{4}\pi\right)^{x_3}\left(\frac{1}{4}-\frac{1}{4}\pi\right)^{x_4}\left(\frac{1}{4}\pi\right)^{x_5}$$

  - Given Y1=X1+X2, both are drawn from multi-nominal distributions sharing the same parameters, we guess the values of x_1 and x_2 (might be non-integer)

$$x_1 = 125\frac{\frac{1}{2}}{\frac{1}{2}+\frac{\pi}{4}},\ x_2 = 125\frac{\frac{\pi}{4}}{\frac{1}{2}+\frac{\pi}{4}};\quad \pi^* = 0.6268$$

Learning from Partially Observable Features (V)

- NLP Example: part of speech tagging
  - Find the appropriate tag sequence c_1^n (=c_1, c_2,…, c_n), to be associated with the given word sequence w_1^n (=w_1, w_2,…, w_n), in the training corpus. Please see Part III for details.
  - Incomplete Data Space: the given word sequence w_1^n
  - Complete Data Space: both the given word sequence w_1^n and its associated tag sequence c_1^n (=c_1, c_2,…, c_n)
  - Given a word “design” that has two possible tags {noun, verb}, is it possible to know the real tag of “design” in a particular context from a large number of instances of “design” in the given untagged corpus ?
    - Note: {noun, verb} => “design”: many-to-one mapping

Why build model on the features that we don't know their values?

- Sometimes, the model in the complete data space is more suitable for the real situation
  - Example: the above genetic problem
- Or, the unobservable features are happen to be the ones that we are really interested
  - Example: POS in tagging problem

Why build model on the features that we don't know their values (cont.)?

- Last, in many cases, introducing more intermediate parameters (and then conditioning on them) help to decompose the original problem into a set of more manageable and simpler sub-problem
  - This is the divide-and-conquer strategy
  - Examples:
    - Intermediate forms (e.g., parse tree and semantic form) adopted in the model of Corpus-Based Statistics-Oriented (CBSO) Machine Translation System

$$P(T_i|S_i,\Lambda)=\sum_{I_i}P(T_i,I_i|S_i,\Lambda)=\sum_{I_i}P(T_i|I_i,S_i,\Lambda)\times P(I_i|S_i,\Lambda)$$

    - States in the Hidden Markov Model (HMM) adopted in the task of speech recognition

$$P(o_1^n|\Lambda)=\sum_{s_1^n}P(o_1^n,s_1^n|\Lambda)=\sum_{s_1^n}P(o_1^n,s_1^n|\Lambda)\times P(s_1^n|\Lambda)$$

What kind of Parameters are Learnable (I)

- Learnability Issues [Duda & Hart 73]
  - Learning parameters from unlabelled data is possible for many cases, BUT
  - Learning parameters may not always stop at the best desired parameter set even though the likelihood value is maximized
- Parameter Learning & Identifiable Issues
  - Most learning methods assumes that observed data **x** was drawn from a distribution $p(\mathbf{x}|\theta)$ of known form (statistics structure) with a set of parameters θ
  - Statistical learning is to uncover the parameter set θ by estimating their values via maximum likelihood estimation.

What kind of Parameters are Learnable (II)

- Parameter Learning & Identifiable Issues (cont.)
  - A parameter set θ is identifiable (or, learnable) in the specified feature space, if for any other $\theta' \neq \theta$, there exists at least an **x** such that $p(\mathbf{x}|\theta) \neq p(\mathbf{x}|\theta')$.
    - We are unable to identify the true parameter set θ, if each element in a specific sub-set in the parameter space shares the same likelihood function.
    - If a parameter set θ is not identifiable, then we will have a collection of global optimum points in the parameter space (with equal maximum likelihood value)
    - Situation might change when we adopt different feature spaces or models.
  - Most mixtures of commonly encountered density functions are identifiable
  - Mixtures of discrete distributions are not so obliging

What kind of Parameters are Learnable (III)

- Example of Unidentifiable Distribution [Duda & Hart 73]

$$p(\mathbf{x}|\theta) = \frac{P(\omega_1)}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\theta_1)^2 \right] + \frac{P(\omega_2)}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\theta_2)^2 \right]$$

  - $p(x|\theta)$: is not identifiable if $P(\omega_1) = P(\omega_2)$
    - θ_1 and θ_2 can be exchanged without affecting the distribution
    - $< \theta_1, \theta_2 >$ has non-unique solutions even though the one(s) with maximum likelihood value could be found by any searching or learning method
  - $p(x|\theta)$: is identifiable, if $P(\omega_1) \neq P(\omega_2)$

What kind of Parameters are Learnable (IV)

- Example: Bigram model for POS tagging

$$\Lambda = \left\{ P(w|c), P\left(c_{right}|c_{left}\right); \text{for all possible combinations} \right\}$$

$$\begin{aligned} P\left(w_1^n|\Lambda\right) &= \sum_{c_1^n} P\left(w_1^n, c_1^n|\Lambda\right) \\ &= \sum_{c_1^n} P\left(w_1^n|c_1^n, \Lambda\right) \times P\left(c_1^n|\Lambda\right) \\ &\equiv \sum_{c_1^n} \left\{ \prod_{i=1}^{n} P\left(w_i|c_i\right) \times P\left(c_i|c_{i-1}\right) \right\} \end{aligned}$$

  - $P(w_1^n|\Lambda)$ is not identifiable for many possible w_1^n.

Multiple Local Maximum Points (I)

- The Log-likelihood function of the K-parameters Exponential Families has only one global optimum point in the parameter space; thus they are identifiable

  - K-parameters Exponential Families include many functions that we are familiar, such as Gaussian, multi-nomial, etc. [Bickel 77]

$$P(X,\theta) = \left\{ \exp\left[\sum_{i=1}^{k} c_i(\theta)\ T_i(X) + d(\theta) + S(X) \right] \right\} I_A(X)$$

$$P(X,\theta) = \exp\left[\frac{\mu}{\sigma^2} x - \frac{x^2}{2\sigma^2} - \frac{1}{2}\left(\frac{\mu^2}{\sigma^2} + \log(2\pi\sigma^2) \right) \right], \text{for Gaussian case.}$$

  - Log-likelihood function of the K-parameters Exponential Families is a convex function

Multiple Local Maximum Points (II)

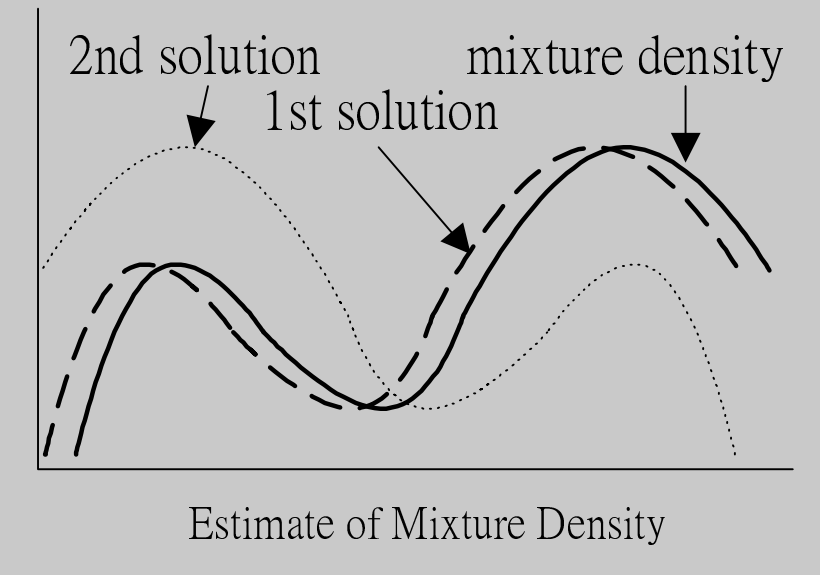
- Mixture density functions, such as above examples, are not belong to those families
  - In general, they have several local optimum points in the parameter space.
  - They even might have several global optimum points in the parameter space
- However, we do not really worry about this issue in NLP tasks, because we usually only search for the nearest local maximum (not even for the unique global optimum point)

Multiple Local Maximum Points (III)

- Example

$$p(\mathbf{x}|\theta) = \frac{P(\omega_1)}{\sqrt{2\pi}}\exp\left[-\frac{1}{2}(x-\theta_1)^2\right] + \frac{P(\omega_2)}{\sqrt{2\pi}}\exp\left[-\frac{1}{2}(x-\theta_2)^2\right]$$

  - Unsupervised learning with Maximum Likelihood Estimation would produce two local maximums (two global optimum points if $P(w_1) = P(w_2)$)



Differences between Supervised and Unsupervised Learning

- Supervised:
  - Features adopted by the Language Model are completely observable
  - Both Observation and Model are in the same *Complete* Data Space
- Unsupervised:
  - Features adopted by the Language Model are not completely observable
  - *Observation* is in the *Incomplete* Data Space; however, the model is in the complete data space
- Supervised learning is more efficient than unsupervised learning
  - Supervised learning generally will have better performance given the same amount of training data

Maximum Likelihood Estimation (MLE)

- The parameters in the statistical language model are usually estimated via Maximum Likelihood Estimation (MLE) method
  - MLE find the point in the parameter space that has the maximum likelihood to generate the all observations.
  - Log Likelihood function: $$Log\ p(\mathbf{x}|\theta) = Log\prod_{i=1}^{n} f_i(x_i,\theta) = \sum_{i=1}^{n} Log\ f_i(x_i,\theta).$$
  - MLE is usually conducted on the Log Likelihood Function (a monotonic mapping)
  - Multiinomial: $$p(\mathbf{x}|\theta) = \theta_1^{k_1}\theta_2^{k_2}\theta_3^{k_3},\ \hat{\theta}_i^{MLE} = \frac{k_i}{n}.$$
  - Gaussian: $$\begin{cases} p(\mathbf{x}_1^n|\theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{1}{2\sigma^2}(x_i-\mu)^2\right\}; \\ \hat{\mu}_{MLE} = \frac{\sum_{i=1}^{n} x_i}{n}, \hat{\sigma}^2_{MLE} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{n}. \end{cases}$$

Maximum Likelihood Estimation (cont.)

- Using relative frequency as the estimation may introduce estimation error for entries that occurs very infrequently or unseen parameters
  - Zero occurrence results in zero probability.
  - Result poor performance in the testing set if the size of the training set is very limited
- MLE approaches achieve the recognition implicitly and indirectly through the estimation process; thus
  - Recognition (or disambiguation) is done through the formula:

$$\hat{C} = \arg\max_{C_i} P\left(O_1^n | C_i, \Lambda\right)$$

  - Maximizing likelihood is not equivalent to minimizing the error rate in a training set.

Performance Issues (I)

- Model Fitting vs. Recognition:
  - Unsupervised learning picks the candidate by comparing the fitness of various models to the observations, measured by likelihood value, of different candidates.
  - Disambiguation is thus done indirectly through comparing the fitness of various models.
  - Motivation for above approach:
    - Baysian classifier is the minimum error rate classifier
    - If we approach the true density function by refining the model (choosing better form and adopting better estimation method), we can obtain the best performance

Performance Issues (II)

- Model Fitting vs. Recognition (cont.):
  - However, this indirect approach does not utilize the data efficiently in many real situations:
    - The form of the true density function is not really known (the true density function might not be manageable even the form is given)
    - The deviation of the estimated density function off the decision boundary does not affect the performance (so why we should bother about that?)
    - Training data is limited, and the ones that really affect performance are those that are near the decision boundary (i.e., the classes separation plane); those data that are far from the decision boundary might even bring in adverse effect.
    - Although Baysian classifier gives the best decision boundary, it is not the only way to find that.
    - Any classifier that can find the best decision boundary also deliver the best performance.

Performance Issues (III)

- Model Fitting vs. Recognition (cont.):
  - The point, in the parameter space, obtained from MLE might not be the point that can minimize the error rate
  - It is actually the correct ranking order of the desired candidate, not the parameter estimation error, that we really care.
    - A feature that fit every class well might add nothing to our performance
    - It is the competition among different candidates, not how well each candidate performs, that really matters.
  - Directly pursue the minimum error rate (in the training set only) is called Discriminative Training [Juang 92]

Performance Issues (IV)

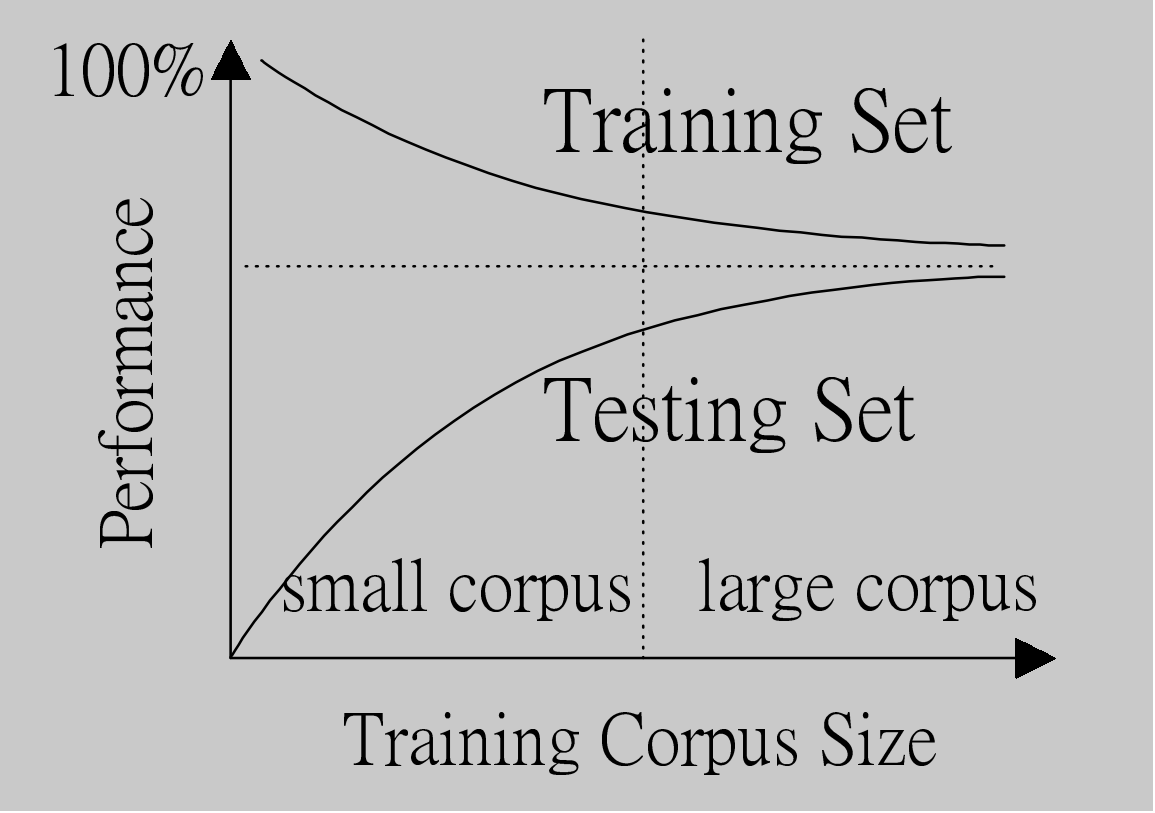
- Training Set Performance versus Testing Set Performance
  - The effect caused by the estimation error is not unveiled in the training set
  - The parameter obtained from MLE in the training set might not be the MLE point we will get from the testing set
  - The modeling error manifested in the training set can always be reduced by increasing the model complexity or through refining the model parameters (via adaptive learning).
  - However, the modeling error usually compete with the estimation error in the testing set (not in the training set)
  - Robustness issue addresses the problem about how to achieve the similar performance in both the training set and the testing set.
  - Usually simpler models and less refined models are more robust

Performance Issues (V)

- Performance of learning (especially unsupervised) is thus greatly affected by:
  - Discrimination power associated with the adopted model
    - Which is the capability to achieve the minimum error rate in the training set
  - Robustness of the model:
    - Which is the capability to overcome the characteristics variation between the training samples and the data in the real world (testing set)

Performance Trends versus Training Corpus Size

- Problems of Corpora with Small size
  - Estimation Error: Training Set Performance /= Testing Set Performance



Performance Trends versus Module Complexity:

- Problems of Models with High Complexity
  - Although increasing the Model Complexity can reduce the modeling error in the training set (thus reducing the error rate in the training set), it does not increase testing set performance without limit