[Day-2] Unsupervised Learning for Natural Language Processing

(Part II: Basic Concepts and Background)

Keh-Yih Su and Jing-Shin Chang
kysu@bdc.com.tw, shin@nlp.csie.ncnu.edu.tw

(2002/8/18)

Behavior Design Corporation (BDC)
No. 5, 2F, Industrial East Road IV, Science-Based Industrial Park
Hsinchu, Taiwan 30077, R.O.C.

Department of Computer Science and Information Engineering
National Chi-Nan University
Puli, Nantou, Taiwan 545, R.O.C.

Day-2: Unsupervised Learning for Natural Language Processing

- Part I: Introduction
  - What and When for Unsupervised Learning, Why it is getting popular
- **Part II: Basic Concepts and Background (using EM as an example)**
  - **Incomplete Data Space**
  - **Learnability**
- Part III: Typical Unsupervised Learning Algorithms: Viterbi & EM
  - Procedures, Characteristics
- Part IV: Potential Traps & Source of Problems
  - Various Mismatches, Model Deficiencies, Local Maximum, and Over-fitting
- Part V: Suggested Strategies for Better Performance
  - Lessons Learned from Past Experience
  - Recommended Procedures for Unsupervised Learning
- Part VI: Advanced Topic: Co-Training
  - Basic Principles
  - Example: Chinese New Word Extraction
- References

Part II: Basic Concept for Unsupervised Learning

- Learning Desired Model from Partially Observable Features
  - How to train a model in which not all 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 parameters 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 (1)

- 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, we need to infer parameters of complete data space from the sample space of incomplete data.

Learning from Partially Observable Features (2)

- 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 (3)

- Learning Step 1: Guess the associated data in the complete data space
  - Viterbi: Hard Labeling
  - EM (Expectation and Maximization): Soft Labeling
- Learning Step 2: Perform Maximum Likelihood Estimation
  - Viterbi: Find the parameter set that has the maximum probability to generate those hard-labels in complete data space
  - EM: Find the parameter set that has the maximum probability to generate those soft-labels in complete data space

Learning from Partially Observable Features (4)

- Generic Model 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]

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

Learning from Partially Observable Features (5)

- Generic Model Example [Dempster 77] (cont.)
  - Complete Data Space: [X1, X2, X3, X4, X5]
  - Cell probabilities: $(1/2, 1/4\pi, 1/4(1-\pi), 1/4(1-\pi), 1/4\pi)$ [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 (6)

- 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 Day-2 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": two-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 the real operating model
  - Example: the above genetic problem (which is more closely match the real underlying operation mechanism)
- 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-problems
  - 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 (1)

- 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(**x**|θ) 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 (2)

- Parameter Learning & Identifiable Issues (cont.)
  - A parameter set θ is identifiable (or, learnable) in the specified feature space, if for any other θ' \= θ , there exists at least an **x** such that p(**x**| θ) \= p(**x** | θ').
    - 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 continuous density functions (i.e., continuous RVs) are identifiable
  - Mixtures of discrete distributions are not so obliging

What kind of Parameters are Learnable (3)

- 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|θ): is not identifiable if P(ω_1) =P(ω_2)
    - θ_1 and θ_2 can be exchanged without affecting the distribution
    - < θ_1, θ_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|θ): is identifiable, if P(ω_1) \=P(ω_2)

What kind of Parameters are Learnable (4)

- Example of 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 (1)

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

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

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

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

  - Log-likelihood function of the K-parameters Exponential Families is a convex function
  - Search can start from any initial point

Multiple Local Maximum Points (2)

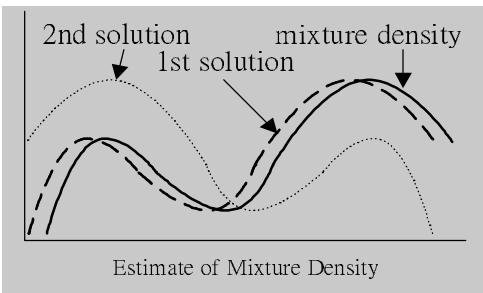
- Mixture density functions, such as above examples, are not belong to those families
  - In general, they might have many local optimum points in the parameter space.
  - They even might have several global optimum points in the parameter space
- However, we do not really worry about this issue in NLP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

Multiple Local Maximum Points (3)

- 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 $P(w_1) = P(w_2)$)



Labeled data are required to attach class-labels

- If models are class-specific (e.g., part-of-speech), not just learning some parameter values (e.g., in the above generic model), then unlabeled data alone are not enough
  - Information for identifying Class-Labels is required
  - For example, we have no idea about which Gaussian component should be associated to a specific class-label in the above example
- Unlabeled data can be used to identify the underlying distributions, and only a small number of labeled data is required to assign the labels
  - Converge exponentially quickly in the number of labeled samples [Castelli 95]

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:

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

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

Performance Issues (1)

- Observations Fitness vs. Human Preference:
  - Unsupervised learning searches the best parameter set under the given form in the complete data space that has the best chance to generate those observations in the incomplete data space (i.e., find the parameter set that *fits data* most)
  - During disambiguation, candidates are then selected by comparing their fitness, measured by likelihood value, to the model
  - However, the system performance is evaluated by the *error rate*, which is actually a measure of *human preference*
  - This criterion mis-match is the same as we observed in the supervised learning

Performance Issues (2)

- Observations Fitness vs. Human Preference (Cont.):
  - However, since human preference is unveiled in the training set, increasing model complexity might not reduce the error rate in the training set (the over-fitting phenomenon observed in the supervised learning might not occur)
  - Similarly, the training set performance might not be improved when we reduce the training corpus size, although the likelihood value will increase
  - Furthermore, during the unsupervised learning process, the likelihood value is guaranteed to increase from iteration to iteration. Nonetheless, the error rate in the training set might not decrease from iteration to iteration (the over-tuning phenomenon observed in the supervised learning might not occur)

Performance Issues (3)

- Observations Fitness vs. Human Preference (Cont.):
  - In the supervised learning, the criterion mis-match can be somehow compensated by performing the adaptive learning. However, post-learning compensation cannot be done in unsupervised learning, as the correct answer is not known
  - Therefore, the performance of unsupervised learning in the training set rely more on the correlation between Observations Fitness and Human Preference inherited from the model

Performance Issues (4)

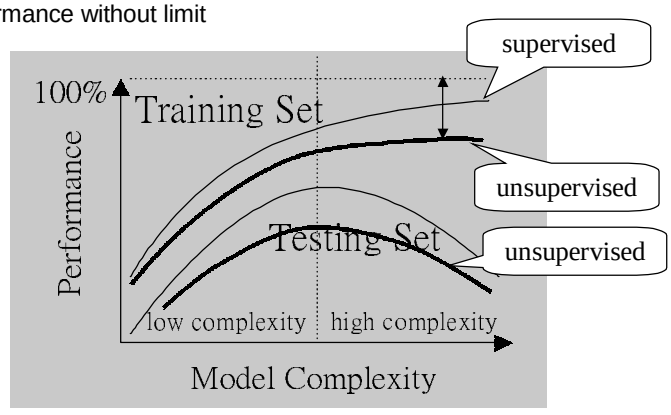
- Training Set versus Testing Set (same as supervised learning)
  - The effect caused by the estimation error will not be unveiled in the training set
  - The parameter obtained from MLE in the training set might not be the MLE point we will get from the testing set
  - The fitness (measured by the associated likelihood value) manifested in the training set can always be enlarged by increasing the model complexity
  - However, the modeling error usually competes with the estimation error in the testing set (not in the training set)

Performance Issues (5)

- Performance of unsupervised learning 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 maintain the similar performance in testing set
    - Usually simpler models are more robust

Performance Trends versus Module Complexity:

- Problems of Models with High Complexity
  - Increasing the Model Complexity might not reduce the error rate in the training set), and it also does not increase testing set performance without limit



Performance Trends versus Training Corpus Size

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