[Day-2] Unsupervised Learning for Natural Language Processing

(Part I: Introduction)

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 I: Introduction

- What is Unsupervised Learning
  - Characteristics & differences with supervised learning
- Clustering and Hidden Markov Model
  - Introduction and examples
- Cross-Entropy for Feature Selection
  - Definition and E-Set example
- When Should Unsupervised Learning be Used
  - Problem characteristics and suitable situations for unsupervised learning
- Why Unsupervised Learning is Becoming Popular
  - Environmental factors & paradigm shift

Modes of Learning

- Supervised Learning
  - Learning from Annotated Examples
  - Advantage: capable to achieve better performance (as more information is carried by the annotation) given the same amount of training data
  - Disadvantage: human annotation is usually time-consuming and expensive
- Unsupervised Learning: Clustering, Viterbi, EM
  - Learning with Un-annotated Examples
  - Advantage: human annotation is not required
  - Disadvantage: performance achieved usually is inferior to that of supervised learning
- Bootstrapping: Co-training
  - Learning with Un-annotated Training Data, however, start from an Annotated *Seed Corpus*
  - A compromise between the supervised learning and un-supervised learning
  - Provide most cost effective solution, if used appropriately

Clustering for forming Classes

- Goal:
  - To collect different tokens having similar behavior into various equivalent classes to improve the robustness of the adopted statistical language model
- Applications:
  - Word sense disambiguation, Machine Translation, etc.
- Typical Procedures
  - Define feature space (e.g., frequent words occurrence vector, etc.)
  - Define similarity measure (e.g., distance, angle difference, etc.)
  - Cluster data iteratively according to the given criterion (e.g., minimum mean square error, maximin-distance, maximum-likelihood-value, etc.)
  - Stop when desired criteria are matched.

Clustering for forming Classes (Cont.)

- Commonly used Clustering Algorithms
  - Dynamic Clustering: find the best K clusters (for a given K)
  - Hierarchical Clustering: #clusters decrement/increment by one per iteration
    - Agglomerative (Bottom-up/Clumping approach): n singletons => K clusters
    - Divisive (Top-Down/Splitting approach): single cluster => K clusters

Dynamic Clustering

- Procedure:
  - Employs an iterative algorithm to optimize a clustering criterion function.
  - At each iteration, data points are assigned to clusters.
  - Then, the cluster representatives are updated to reflect any change in the data point assignment.
  - The new cluster models are used in the next iteration.
  - Continue until a stable partition is obtained.
  - ➢ The number of clusters is known beforehand.
- Example: K-means clustering
  1. Choose the number of classes, K
  2. Choose initial class means: $\mu_1, \mu_2, \ldots, \mu_K, .$
  3. Classify each data x_i to one of the K classes.
  4. Re-compute the estimates for μ_i using the results of 3.
  5. If the μ_i are consistent then STOP; otherwise go to (3) and (4).

Hierarchical Clustering [Bottom-Up]

- Advantage: Number of Clusters need not be pre-specified
- Procedure:
  - Initially, every point in the data set is considered as a separate cluster.
  - At any stage of a hierarchical clustering algorithm, the two of the existing clusters which are most *similar* are merged to create a new cluster, thus reducing the number of potential clusters by one.
  - Terminate when the desired criteria are matched.
  - ➢ The number of clusters is unknown beforehand.
- Examples [Brown 92]
  - Friday Monday Thursday ...
  - people guys folks fellows ...
  - water gas coal liquid acid ...
  - man woman boy girl ...
  - head body hands eyes ...

Hidden Markov Model (HMM)

- In most NLP applications, HMM is a Markov Chain with its associated state-label in each stage unknown
  - The associated state-sequence is unknown, it can only be indirectly guessed through the given observation sequence
  - Likelihood of observation: (⇔ words in tagging task)

$$P(o_1^n \mid \lambda) = \sum_{s_1^n} P(o_1^n, s_1^n \mid \lambda) = \sum_{s_1^n} P(o_1^n \mid s_1^n, \lambda) P(s_1^n \mid \lambda)$$

$$P(s_1^n \mid \lambda) = \prod_i P(s_i \mid s_{i-1}, \lambda)$$

  - Likelihood of state sequence: (⇔ tags in tagging task)

$$\arg\max_{s_1^n} P(s_1^n \mid o_1^n, \lambda) = \arg\max_{s_1^n} \prod_i P(o_i \mid s_i, \lambda) P(s_i \mid s_{i-1}, \lambda)$$

[Solution: Viterbi Decoding Algorithm]

Hidden Markov Model (HMM)

- Tagging example: The associated tag-sequence (C_1 to C_n) is unknown (and it is to be found out), only the word-sequence (w_1 to w_n) is given, and

$$P(C_1^n \mid w_1^n, \lambda) \approx \left[\prod_i P(w_i \mid C_i, \lambda) P(C_i \mid C_{i-1}, \lambda)\right] / P(w_1^n)$$

- Three HMM problems [Rabiner 93]:
  - How to compute output probability of observed output symbols
  - How to find the best corresponding state-sequence {s1, …, sn}
  - How to estimate the associated parameter set Λ = (**A, B,** Π) from the given observation sequence (Unsupervised)
  - (How to efficiently resolve the above problems)

Hidden Markov Model (1)

- Parameter set $\Lambda = (\mathbf{A}, \mathbf{B}, \Pi)$
  - State Transition Probability Matrix **A**:

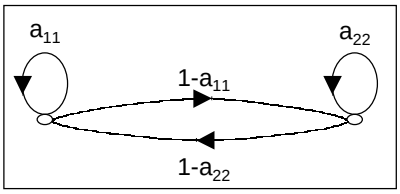
$$a_{ij} = P(q_t = j \mid q_{t-1} = i), \qquad a_{ij} \geq 0, \sum_j a_{ij} = 1$$

  - Observation Symbol Probability Matrix B:

$$b_j(k) = P(o_t = k \mid q_t = j)$$

  - Initial State Distribution Probability Vector π :

$$\pi_i = P(q_1 = i)$$



π1=0.4 π2=0.6
P(H) = P_1 P(H) = P_2
P(T) = 1-P_1 P(T) = 1-P_2

O = H H T T H T H H T T H . . .
S = 2 1 1 2 2 2 1 2 2 1 2 . . .

Hidden Markov Model (2)

- Tagging Example

$$P\left(C_1^n \mid w_1^n\right) = \left[\prod_{i=1}^{n} P\left(C_i \mid C_{i-1}\right) P\left(w_i \mid C_i\right)\right] / P\left(w_1^n\right)$$

$$\pi_i = P\left(C_{1,i}\right),\ a_{ij} \Leftrightarrow P\left(C_{t,j} \mid C_{t-1,i}\right),\ b_j(k) \Leftrightarrow P\left(w_{t,j,k} = w_t \mid C_{t,j}\right)$$

[subscript 't': time index]

- Usually adopt Dynamic Programming and A* Searching Algorithms [Winston 92]
  - Dynamic Programming is first adopted to perform forward searching for finding the best candidate (state sequence)
  - A* algorithm is then applied backwards to get desired Top-N candidates

HMM Problem I (Decoding)

- Compute Observation Probability

- Direct computation:
  - Introducing hidden state variables

$$P(O \mid \lambda) = \sum_{Q} P(O, Q \mid \lambda) = \sum_{Q} P(O \mid Q, \lambda) P(Q \mid \lambda)$$

$$P(Q \mid \lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

$$P(O \mid Q, \lambda) = b_{q_1}(o_1) b_{q_2}(o_2) \ldots b_{q_T}(o_T)$$

$$P(O \mid \lambda) = \sum_{Q = q_1^T} \pi_{q_1} b_{q_1}(o_1) \prod_{t=1, T-1} a_{q_t q_{t+1}} b_{q_{t+1}}(o_{t+1})$$

HMM Problem II

- Finding Best (Hidden) State Sequence

- Goal: $Q^* = \arg\max_{Q} P(Q \mid O, \lambda) = \arg\max_{Q} P(Q, O \mid \lambda)$
- Partial Solution: maximum probability of state sequences after seeing partial observation

$$\delta_t(i) = \max_{q_1 \ldots q_{t-1}} P(q_1^{t-1}, o_1^{t-1}, q_t = i, o_t)$$

- Recursive Relationship: between partial solutions

$$\delta_{t+1}(j) = \max_i \delta_t(i) a_{ij} b_j(o_{t+1})$$

$$\psi_{t+1}(j) = \arg\max_i \delta_t(i) a_{ij} b_j(o_{t+1})$$

- Termination and Tracing back best state sequence

$$q_T^* = \arg\max_i \delta_T(i) \qquad q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad P^*(O, Q \mid \lambda) = \max_i \delta_T(i)$$

HMM Problem III

- Parameter Estimation (Unsupervised, EM)

- Transition probability from state i to state j (at time t)

$$\xi_t(i,j) \equiv P(q_t = i, q_{t+1} = j \mid O, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O \mid \lambda) = \sum_s \alpha_t(s)\beta_t(s)}$$

$$\gamma_t(i) = \sum_j \xi_t(i,j), \quad \text{with } \alpha_t(i) \equiv P(o_1^t, q_t = i \mid \lambda) \;\&\; \beta_t(i) \equiv P(o_{t+1}^T \mid q_t = i, \lambda)$$

- Estimating probability as expected number of transitions / probabilities: Iteratively starting from a guess on (**A, B,** Π)

$$\bar{\pi}_i = \gamma_1(i) = \text{expected transition probability from state i at time (t = 1)}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_i(t)} = \frac{\text{expected \# transitions from state i to j}}{\text{expected \# transitions from state i}}$$

$$\bar{b}_{ik} = \frac{\sum_{t:o_t=k} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)} = \frac{\text{expected \# transition from state i with output k}}{\text{expected \# transition from state i}}$$

More NLP Examples with Unsupervised Learning: Machine Translation System

$$P(T_i \mid S_i) = \sum_{I_i} P(T_i, I_i \mid S_i)$$

$$\cong \sum_{I_i} \{ \left[P(T_i \mid PT_t(i)) \times P(PT_t(i) \mid NF1_t(i)) \times P(NF1_t(i) \mid NF2_t(i)) \right] \cdots (1)$$

$$\times \left[P(NF2_t(i) \mid NF2_s(i)) \right] \cdots (2)$$

$$\times \left[P(NF2_s(i) \mid NF1_s(i)) \times P(NF1_s(i) \mid PT_s(i)) \times P(PT_s(i) \mid S_i) \right] \} \cdots (3)$$

where

- S_i: Source Sentence T_i: Target Sentence
- I_i: intermediate forms for the source-target pair
- PT: parse tree (s: source & t: target)
- NF1: level-1 normal form NF2: level-2 normal form
- (1) generation score (2) transfer score (3) analysis score

Cross-Entropy for Feature Selection

- In supervised learning, the error rate is usually the criterion for feature selection, which is, however, not available in unsupervised learning in many cases
- Other highly correlated measures (with the error rate) should be adopted
  - Characteristics of good features: possess large inter-cluster distance and with small intra-cluster variance
  - Or, in other words: with great probabilistic distribution mis-match (less overlapping portion in probabilistic distribution)
  - Cross-Entropy (also known as Divergence) is a probabilistic distribution mis-match measure [Tou&Gonzalez 74, and Blahut 87]

Cross-Entropy for Feature Selection (Cont.)

- Cross-Entropy Definition
  - Cross-Entropy (Discriminating Information, Kullback-Leibler Distance):

$$L(\mathbf{q_0};\mathbf{q_1}) = \sum_{k=1}^{K} q_{0k} \log \frac{q_{0k}}{q_{1k}} = \sum_{k=1}^{K} q_{0k} l(k)$$

    - Expected value of log-likelihood ratio between two sets of probability vectors of observing features from two classes C_0 and C_1
    - k: an observed feature, *l(k):* log-likelihood ratio for feature k
    - q_{ok}: the probability that k is from C_0, q_{1k}: the probability that k is from C_1
  - Discrimination Information always increases *via* adding nontrivial features
    - The discrimination Information of each feature can be evaluated with the aid of a seed corpus (or a cross-validation set)
  - Symmetrical form is known as the Divergence: $L(\mathbf{q_o}; \mathbf{q_1}) + L(\mathbf{q_1}; \mathbf{q_o})$

Cross-Entropy Example: E-Set Recognition [Su&Lee 94]

- E-Set: {b, c, d, e, g, p, t, v, z}, total nine English letters with the same ending vowel "E" (a confusion set).
- Speech Recognition: conducted in a multi-speaker, isolated-word mode.
- Training Set: 900 tokens from 100 speakers
- Testing Set: another 900 tokens from the same 100 speakers
- Each letter is modeled by a 5-state left-to-right HMM
- HMM Baseline Recognition Rates: 61.7% for the testing set, and 80.2% for the training set.

Cross-Entropy Example (1)

- Input observation vector X:
  - where S_j^i is either the averaged or the accumulated state log-likelihood (score) for state *j* of word *i.* There are 45 elements in this *score vector*.

$$X^t = \left[S_1^1, S_2^1, S_3^1, \cdots S_3^9, S_4^9, S_5^9 \right]$$

- Subspace Projection:
  - Maximin Algorithm is proposed for feature selection:
    - Divergence $D_{i,j}(k)$ between the class i and the class j, for 1 <= i, j <= 9; i < j, 1 <= k <= 45.

$$D_{ij}(k) = \int_{Y_k} (P_i(Y_k) - P_j(Y_k)) \cdot \ln \left(\frac{P_i(Y_k)}{P_j(Y_k)} \right) dY_k$$

    - Find $D_{min}(k) = \min_{i,j} D_{i,j}(k)$ for 1 <= i, j <= 9; i < j, 1 <= k <= 45.
    - Sort $D_{min}(k)$ in descending order, then the sequence of subspaces is obtained

Cross-Entropy Example (2)

TABLE I

THE 45 FEATURE INDICES LISTED IN THE ORDER OF DESCENDING DIVERGENCE VALUES (LEFT TO RIGHT FIRST, THEN TOP TO BOTTOM)

1 (B_1)	6 (C_1)	31 (T_1)	41 (Z_1)	26 (P_1)
21 (G_1)	17 (E_2)	37 (V_2)	36 (V_1)	16 (E_1)
22 (G_2)	18 (E_3)	11 (D_1)	38 (V_3)	2 (B_2)
12 (D_2)	10 (C_5)	4 (B_4)	28 (P_3)	43 (Z_3)
30 (P_5)	20 (E_5)	34 (T_4)	27 (P_2)	7 (C_2)
32 (T_2)	19 (E_4)	14 (D_4)	33 (T_3)	45 (Z_5)
13 (D_3)	9 (C_4)	40 (V_5)	39 (V_4)	42 (Z_2)
25 (G_5)	15 (D_5)	24 (G_4)	8 (C_3)	29 (P_4)
5 (B_5)	3 (P_3)	23 (G_3)	35 (T_5)	44 (Z_4)

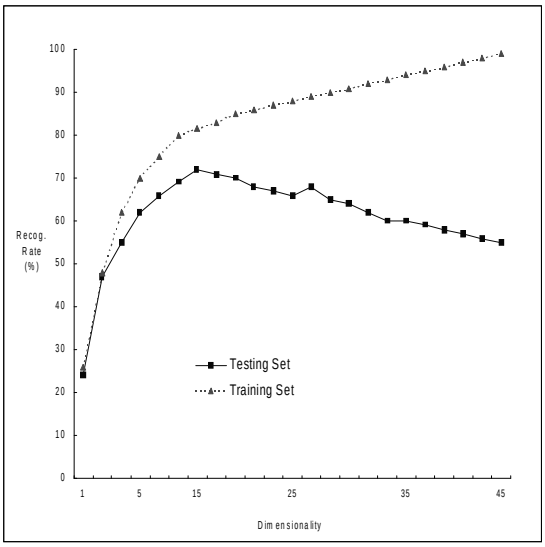
Cross-Entropy Example (3)

TABLE II

THE 45 CORRESPONDING DIVERGENCE VALUES LISTED IN DESCENDING ORDER

0.099410	0.066085	0.042937	0.038249	0.037496
0.028690	0.020272	0.019982	0.017487	0.011872
0.009741	0.005572	0.005305	0.004210	0.004037
0.004005	0.003803	0.003728	0.003656	0.003621
0.003214	0.003200	0.003085	0.002916	0.002690
0.002147	0.002064	0.001937	0.001900	0.001661
0.001651	0.001637	0.001496	0.001221	0.000969
0.000969	0.000596	0.000562	0.000384	0.000381
0.000229	0.000194	0.000187	0.000076	0.000042

Subspace-based Recognizer: Recognition Rate vs. Dimensionality



Characteristics of Unsupervised Learning

- Error rate of training set is not available
  - Other object measure is needed to guide both feature selection and parameter searching processes
- Human preference is not unveiled
  - Criterion mismatch is more serious
- The form of Language model is more critical
  - It would favor the case with less probability terms from iteration to iteration
- Many local maximums exist even for likelihood measure
  - The corresponding form is usually not in the regular exponential family
  - Local trap in searching space is more severe

Characteristics of Unsupervised Learning (Cont.)

- Any constraint would significantly help
  - Speech recognition: the syllables state segmentation sequence is unknown however, the corresponding text string is known
  - Sentence alignment: each alignment-passage is unknown, however, two documents are known to correspond to each other
  - Machine Translation: the detailed structure mapping is unknown, however, two sentences are known to be the corresponding translation.
- Learning efficiency is relatively lower comparing with supervised learning
  - A larger corpus is required to achieve the similar performance (if it is possible) of the supervised learning, as it is operating without the information implied by the annotation which is available in supervised learning

When to Use Unsupervised Learning (1)

- Problem Characteristics
  - The task at hand requires huge amount of fine grained knowledge to achieve acceptable performance
    - Usually imply that providing supervised learning examples might not be affordable
  - Inherent Constraints (or implied dependency) among the linguistic units are strong
    - Have a better chance to predict the best candidate through good language model
    - Bilingual corpus can help for imposing constraint
  - Training data have enough explicit Inherent anchor points
    - Help to impose contextual constrains on their neighbors (e.g., un-ambiguous words in tagging part-of-speech, paragraph markers in sentence alignment)
    - Make the task for resolving ambiguity easier

When to Use Unsupervised Learning (2)

- Problem Characteristics (Cont.)
  - Annotation Task is difficult to proceed (highly confusing in assigning labels)
    - For example, part of speech tagging versus word sense disambiguation (even human have difficulty to assign appropriate word senses according to WordNet definition)
    - Annotation cost would be high, also the consistency and quality will be hard to maintain
    - The entropy reduction (i.e., information gain) obtained from annotating the corpus would not be large, which would imply that supervised learning might not get significantly better result
    - Therefore, annotating corpus may not be worth

When to Use Unsupervised Learning (3)

- Resource Scarceness: Mass amount of un-annotated data is available
  - Information concentration (advantage own by the supervised-learning) can be compensated by the extra amount of training data
  - With huge amount of data, constraints can be imposed to the corpus for helping to select more certain parts (e.g., selecting the boundary sentences of paragraphs for training the sentence segmentation model)
  - Implied information might be enough to cover the unobservable knowledge which would be, otherwise, directly provided in the supervised-learning case (from less amount of training data)

When to Use Unsupervised Learning (4)

- Cost for Preparing Learning Samples is high
  - Cost for Collecting Training Samples:
    - Have people do the data entry work (e.g., LEXIS-NEXIS)
    - From public resources (e.g., LCD, ROCLING, etc.)
    - From the web/news/bbs with automatic tools
  - Cost for Annotating Training Samples :
    - This is usually the bottleneck for supervised learning: requiring number of qualified persons for annotating the corpus and doing consistency check; besides, it also requires a long period of time for large scale projects
  - Choose unsupervised learning if you cannot afford the cost required

When to Use Unsupervised Learning (5)

- Features, domains/tasks updated frequently
  - Re-annotation & re-training are frequently required, if supervised learning is adopted
    - Example 1: uncertain in classification hierarchy to be adopted
    - Example 2: uncertain in discriminative features to be adopted
  - Will dimensionality, values, or labels of the feature vector change frequently?
  - Do you expect the tasks or requirements (e.g., US/NIST MT Evaluation) to keep changing in each phase?
  - Unsupervised-learning is prefer if the frequency is high

When to Use Unsupervised Learning (6)

- Size of available resource keep increasing with time:
  - System parameters could be updated frequently for incremental improvement, which would be difficult for supervised learning
- Good Language Model that can echo the human preference is available
  - Without the annotation to unveil the human preference (as the supervised learning possesses), the human preference must be implicitly implied in the given language model
  - With a good language model, it would have a better chance to learn language parameters that are capable to achieve satisfactory performance
- In summary: when the projected cost for annotating corpus to achieve the desired performance is high, and it is expected that unsupervised learning can achieve competitive performance by adopting better language model, choose unsupervised learning

Why unsupervised learning is getting popular in NLP (1)

- The fact that NLP requires huge and find-grained knowledge and the infeasibility to annotate a big corpus is increasingly perceived
- In general, better performance requires deeper analyses; however, the annotating task gets more and more difficult when the analysis gets deeper
  - The increase of inherent non-determinism make the task of assigning tags more difficult.
- Many public corpora only provide minimum degree of annotation (partially anchored); the cost for further annotation is beyond the reach of most researchers
  - The environment and supports for supervised learning is limited

Why unsupervised learning is getting popular in NLP (2)

- On the other hand, the cost for possessing an un-annotated corpus diminishes to almost nothing
  - Almost free through resource sharing (e.g., LDC, ROCLING, etc.) or through acquiring from WWW; however, annotated corpora are still rare
- The size of on-line corpora increases rapidly in Internet age
  - The degree of knowledge concentration is no more essential.
  - Multi-lingual corpus is more available
    - Implicit constraints and implied annotations can reduce the degree of ambiguity
    - WWW provides an abundant resource

Why unsupervised learning is getting popular in NLP (3)

- Corpus-Based Statistic-Oriented (CBSO) approaches prevail in NLP community; however, it is a rapidly changing field
  - New model is tried in a fast pace (an exciting field)
  - New features, classes are tested and refined rapidly
  - Very difficult to keep the associated annotation updated accordingly, if the supervised-learning approach is adopted
- New applications emerge, and the capability to be customizable and self-learnable is increasingly emphasized
  - Annotating associated corpora for various domains is usually not affordable