[Day-1] Introduction to Statistical Natural Language Processing

(Part II: Introduction II)

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

(2002/8/17)

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-1: Introduction to Statistical Natural Language Processing (mainly on Supervised Learning)

- Part I: Introduction (1)
  - Problems and Characteristics of Natural Language Processing
- **Part II: Introduction (2)**
  - **What, When and Why Statistical Approach**
- Part III: Basic Concepts and Background
  - Feature Space, Probability, Estimator, Stochastic Process, Data Set Classification, and Performance Measure
- Part IV: Typical Applications
  - Word Segmentation, Tagging, Selecting Parse Tree, Aligning Bilingual Corpus
- Part V: Techniques for Improving Performance
  - Smoothing, Class-Based Model, Adaptive Learning, Tips for Checking
- Part VI: Advanced Topics: SVM, ME
  - Support Vector Machine, Maximum Entropy Models
- Appendix: Related Techniques
  - Parameter Estimation, Fractional Factorial Experiment Design, Decision Tree

Part II: Introduction (2)

- Machine Learning in NLP
  - Modes: supervise, un-supervised, bootstrapping
  - Types: learning symbolic relationship, learning model parameters
- Why parameter learning
  - Advantages of Parameterized Systems
  - Types: Neural-Net, Statistical Learning
- Why statistical parameter learning
  - What is a Statistical Language Model
  - Why Statistical NLP
- Why Corpus-Based Statistics-Oriented Approach
  - Why not purely statistical approach
  - How CBSO attack those problems

Learning Modes in NLP (1)

- **Supervised Learning**
  - Learning from Annotated Examples
  - Example: part-of-speech tagging

    1. Collect text and get a dictionary:
       the (det) design (n/v) of (prep) computer (n) ...
    2. Annotate with correct parts-of-speech by human :
       the (det) design (n) of (prep) computer (n) ...
    3. Estimate Model Parameters according to annotation:
       P(n|det)=90/123, P(adj|det)=33/123, P(prep|n) =63/250, …,
       P(n|prep)=61/97, …
    4. Conduct predictions by evaluating likelihood of each candidate :
       P(…, det, n, prep, n, …) = … 90/123 x 63/250 x 61/97 x …

Learning Modes in NLP (2)

- **Supervised Learning (Cont.)**
  - Advantage: capable of achieving 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 (plus inconsistent)
  - ✓ Selective Sampling: Select more effective new data for annotation to increase data collection efficiency
    - Oversample low frequency outcomes, then weight data counts differently during training
  - ✓ Active Learning: Iterative, interactive sampling
    - Sample data that are confusing to the system (i.e., scores are very similar)
    - Label those data and re-train the system (5 to 500 times saving were reported)

Learning Modes in NLP (3)

- **Unsupervised Learning**
  - Learning from Un-annotated samples
  - Example: part-of-speech tagging
    - Do not have human annotation in Step (2) of supervised learning
    - Do not base on human annotation for estimating initial language parameters in Step (3) of supervised learning
  - Advantage: human annotation is not required
  - Disadvantage: performance achieved usually is inferior to that of supervised learning
- **Bootstrapping:**
  - Learning with Un-annotated Training Data, however, start from an annotated *Seed Corpus*
  - A compromise between supervised learning and un-supervised learning
  - Provide most cost effective solution, if used appropriately

Types of Machine Learning (according to the type of knowledge acquired)

- Symbolic: learning symbolic relationship (If ~, then ~)
  - Including: patterns, grammars, rules, decision trees, semantic frames, networks, etc.
    - Example: Grammar Inference, CART [Breiman et al. 1984], Transformation-Based Tagging [Brill 1994]
  - Advantages:
    - Flexible, familiar to human (less learning time)
    - Acquired knowledge usually is more compact and easy to interpret
    - Easily fit into existing linguistic theories

Types of Machine Learning (2)

- Symbolic learning (Cont.)
  - Disadvantages:
    - Relatively awkward in dealing with complex and irregular decision boundary (each rule acts as a hyperplane which cuts the given feature space into two halves; therefore, many rules might be required for a complicated decision boundary)
    - Might require more complicated control mechanism (e.g., one decision tree per outcome space for Target-Template selection problem)
    - Hard Reject (Go/No-Go) approach, usually unable to achieve the best performance
  - Suitable for handling compact and regular situations

Types of Machine Learning (3)

- Parametric: learning parameter values under known parametric forms
  - Including: Neural-Net (learning weighting coefficients), statistical Language Model (learning statistical parameters), etc.
    - Example: Statistical Tri-gram Tagging Model [Church 1988]
  - Advantages:
    - Acquisition and control mechanism are uniform and simple
    - Quantitative measure can be provided (Hard Rejection is a special case); capable of achieving the best performance
    - Adaptability is high (feedback mechanisms can be easily implemented, and adjustment step-size can be pre-specified)

Types of Machine Learning (4)

- Parametric learning (Cont.)
  - Disadvantages:
    - Data size is relatively large (i.e., require large memory space), as statistical parametric forms (e.g., Gaussian, etc.) cannot be applied in many cases (e.g., “distance” cannot be defined between “NP” and “VP”)
    - Mainly due to also keeping unnecessary detailed information under the same decision region
    - Acquired knowledge is not intuitive and not easy to explain
  - Suitable for handling complex and irregular situations
    - Since decision is individually made on each outcome point, very complex decision boundary is allowed

Why Parametric Learning (1)

- NLP requires fine-grained knowledge
  - Inherited characteristics from the given problem
    - Different classes just don't have clean and regular separation boundaries between them
  - A lot of local descriptions are required
    - Which implies that a huge number of rules would be required, if rule-based approach is adopted
  - A simple control mechanism is thus essential to manage huge and messy knowledge required
    - The parametric approach is the ideal candidate. On the other hand, each rule might possess different kinds of combination of matching patterns

Why Parametric Learning (2)

- NLP is a non-deterministic process
  - Hard-Rejection will accumulate errors quickly in a multi-stages design when the accuracy is not very high (say, over 99%)
    - The associated performance might drop to 12% for a 20 stages system with each module having 90% accuracy rate
  - Needs refined Preference-Measure (not just "o" or "1") to generate Best-N candidates in each stage to raise the including rate
  - Parametric system will not rule out any possibility (just assign preference)

Why Parametric Learning (3)

- Unsupervised-learning is the preferred operating mode in many different situations
  - Supervised-Learning requires annotating the corpus which would be a difficult and expensive task in many cases
    - It is not easy to consistently annotate a large corpus, and it would also require a lot of manpower
  - Unsupervised Learning is not easy to go with symbolic approach
    - Un-supervised learning requires an objective measure to tell it where to go during learning procedure
    - It is difficult for symbolic learning to provide such an objective measure
    - Most symbolic learning algorithms operate only under the supervised-mode (i.e., the features and labels based on which rules are induced must be observable)

Why Parametric Learning (4)

- Performance is emphasized more and more
  - Human wage usually occupies a large percentage of total cost, and human power required for operating the system greatly depends on the error rate
    - For example: Machine Translation, OCR, telephone switching system, etc.
    - That is why we care about the *error reduction rate* (advancing from 98% to 99% makes sense)
  - 50% accuracy is not half the value of 90% accuracy
    - Probing Time & Digging Time (cost) might outperform the Value of Oil (or Gold)
  - Computer Memory and raw power are no more the issues
    - As Moore's Law keeps going, we only care about the scarceness of human resources not that of computer
    - Computer resource required for parametric learning is no more a constrain
  - Parametric approaches are more promising for delivering better performance

Why Parametric Learning (5)

- Further performance push requires quantitative knowledge
  - Refined models required quantitative information in almost every field
    - For example, Newton's law: F= ma
  - Detailed study usually unveils non-deterministic phenomenon
    - Uncertainty measure is a kind of quantitative knowledge
  - Quantitative model can outperform qualitative model
    - Qualitative model is a special case of the corresponding quantitative model
  - Symbolic approaches usually are not suitable for providing the quantitative knowledge required
    - Rules only make Go or No-Go decision (i.e., a Hard-Rejection approach), in general

Why Parametric Learning (6)

- Parameterized System is better from the operational point of view
  - Consistency and maintainability is essential when a system scales up
    - Adding/Deleting a rule might introduce side-effect (e.g., blocking some other rules, or generating contradictory in some cases) in non-monotonically reasoning systems
    - Rule ordering is usually sensitive in rule-based approaches, which also impose problems for maintaining the system
  - Customization is easy:
    - A general set of rules frequently cannot fit the demand from different kinds of customers
    - Retain different sets of parameters for various domains and users is needed
    - Domain/User Adaptation is easy for parametric approaches: just re-estimate domain-specific (or user-specific) parameters from various related corpora
  - Capable for self-learning:
    - User feedback mechanism can be easily implemented in a parametric system, thus the system can fit the user better and better

Neural-Net (or Connectionist): A Parametric Approach

- Learning weighting coefficients associated with those connection-links between neurons
  - A black-box approach: could act as a universal approximator
  - If number of data is huge enough, it is better not to make any model assumption (although a model uses data more efficiently, it also introduce modeling error)
- Input/Output is usually a fixed-dimension pattern
  - Number of neurons in the input/output layer is fixed
- Advantages:
  - Provide a quick solution: extensive problem analysis is not required
  - The mechanism is simple and easy to understand: a weapon for everybody
  - Suitable for real-time applications (architecture for parallel processing is implied)
  - Directly minimizing the error rate (no criterion mis-match)
  - If data is abundant, modeling error can be avoided

Neural-Net (Cont.)

- Disadvantages:
  - Not easy to handle the feature whose dimensionality dynamically varies (e.g., number of terminal-symbols under a constituent)
  - Not easy to handle the candidate of hierarchical structure with varying depth (e.g., linguistic constituents)
  - Not easy to handle the case when the number of possible candidates is open (as the number of output neurons is fixed)
  - Parameters are not associated with any physical meaning, thus the capability for further processing (e.g., back-off smoothing, etc.) is limited
  - Learning process is inefficient: requires relatively large amount of training data, and convergence period
  - Generalization capability is usually poor when training data is not abundant
  - Without truly understanding the problem, further improvement is difficult

Statistical Language Model: Another Parametric Approach

- Statistical Learning is performed under the Statistical Language Model
  - Trigram tagging model: $\arg\max_{t_1^n} P(t_1^n \mid w_1^n) = \arg\max_{t_1^n} \prod_i P(t_i \mid t_{i-1}, t_{i-2})$
- A Statistical Language Model consists of an adopted Probabilistic Form and its associated Parameter Values
  - Probabilistic form characterizes the relationship among features (usually symbols in NLP applications) to reflect the characteristics of problem and make computation feasible
  - The associated parameters then quantify the relationship among features
- The *form* plus the associated *parameters* is the knowledge
  - Knowledge is implicitly embedded in (or implied by) those large number of parameters

Statistical Language Model (Cont.)

- The form (feature dependency) is usually problem-dependent, and the way to estimate those parameters (e.g., MLE, Good-Turing, Back-off, EM, ME, SVM, etc.) is more problem-independent
  - Some parameter estimation methods such as SVM (or other kernel based approaches), however, also restrict the family of the forms allowed
  - Usually the form has a larger influence on the performance
- Whether a given statistical language model is good or not should be judged by how closely it can reflect human preference
  - It should not be judged indirectly by its fitness measure to the data, e.g., likelihood or perplexity value

Statistical Learning (1)

- Learn those associated statistical parameters implied by the model (a glass-box approach)
- Advantages:
  - Very flexibly. Don't have the limitations associated with the neural-net (e.g., fixed-dimension feature vector, fixed number of output candidates, etc.)
  - Meaningful operations can be taken on parameters (Smoothing, Clustering, etc.)
  - Decisions based on Bayesian classifier also implies minimum error rate (if the model is correct): provides a promising approach to deliver the best performance
  - Supported by well-established statistics theories: we know how to improve the performance (and why it can be improved) by using many existing techniques

Statistical Learning (2)

- Advantages (cont.):
  - Provide direct and flexible control to support those hierarchical internal structures (i.e., intermediate forms) in multi-stages design
    - Example: Machine Translation [Su 1995]

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

$$= \sum_{I_i} P(T_i, PT_t(i), NF1_t(i), NF2_t(i), NF2_s(i), NF1_s(i), PT_s(i) \mid S_i)$$

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

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

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

- where: S: source sentence, T: target sentence, I: intermediate normal forms
- $I_i = \{PT_t(i), NF1_t(i), NF2_t(i), NF2_s(i), NF1_s(i), PT_s(i)\}$, in which
- PT: parse tree, NF1: normalized syntax tree, NF2: normalized semantic tree
- (1) = generation score (2) = transfer score (3) = analysis score

Statistical Learning (3)

- Advantages (cont.):
  - Problems can be easily decomposed into more manageable and simpler explicit sub-problems
    - By first introducing associated intermediate random variables, and then conditioning on them
    - Typical Form: (introducing intermediate/hidden variables…)

$$P(M \mid F) = \sum_{H_i} P(M, H_i \mid F) = \sum_{H_i} P(M \mid H_i, F) \times P(H_i \mid F)$$

  - Form is more extendable (with respect to model complexity) and scalable (with respect to dimensionality of feature space) when the problem is better understood and modeling is to be advanced

Statistical Learning (4)

- Disadvantages:
  - Require statistical background and modeling capability
  - Require problem analysis stage
  - Require an additional discrimination learning stage to compensate the criterion mismatch (Maximum Likelihood vs. Minimum Error Rate)

Why Statistical Learning for NLP (1)

- More suitable for NLP application
  - Linguistic constituents are usually not fixed-dimension patterns. They have hierarchical structures with varying number of terminal nodes and depth
  - Number of candidates frequently depends on the context (e.g., number of allowable parse-trees) and cannot be known in advance
  - NLP is usually a multi-stages process: we need more direct control over those intermediate forms (neural-net's hidden layer is unmanageable)
- More suitable for unsupervised learning
  - Unsupervised learning is usually preferred, as consistently annotating a large corpus is a difficult and expensive task
  - Unsupervised Learning other than clustering is difficult with Neural Net approach (lacking an objective function for guiding learning process)

Why Statistical Learning for NLP (2)

- More promising to deliver better performance
  - With the aid of modeling, statistical approach is more promising in generating better performance given the same fixed amount of training data (more efficient in data utilization)
    - With respect to the inherited complexity in NLP, the amount of available training data is still too limited (not enough to support those brute force approaches)
    - An appropriate model would form various equivalent classes, thus dramatically reducing the number of parameters required
    - Additional Problem Knowledge (or Domain Knowledge), acquired through analyzing problem, would add extra strength as research goes on
  - Better results have been reported

Why Statistical Learning for NLP (3)

- More efficient training process
  - Theoretically, every model (that can be converted into a mapping function) is able to be implemented with a universal approximator; however, learning every transformation from scratch is inefficient (requires lots of data)
    - Some neural-net approaches add a pre-processor to include feature transformation for promoting data utilization efficiency (e.g., perform state-sequence-segmentation in speech recognition); however, this approach deviates from its advantage of simplicity
  - Statistical approaches usually offer faster convergence speed and require a shorter training session
    - More efficient training process ensures fast testing turn around time, and thus accelerate R&D advancing pace

Why Statistical Learning for NLP (4)

- Real-time requirement is not a serious constrain now
  - With the Moore's Law keeps going, it is possible to implement many applications in software now (e.g., speech recognition)
  - Whether the architecture is more suitable for hardware implementation is thus less concerned during decision making

Purely Statistical Approach

- Have no concept of "part-of-speech" and "root-form" in mind (IBM Models 1-5)
- Cannot learn the rule of "X category cannot follow Y category" (as trigram model)
- They learn "A group of words cannot follow another group of words"

X	Y	P(Y|X)
the	ate	0
a	take	0
an	took	0
	walk	0
	walks	0
	walked	0
	buy	0

$$P(W_i \mid W_{i-1}, W_{i-2}) \Rightarrow (10^5)^3 = 10^{15} \text{ parameters}$$
$$|V| = 100{,}000 \text{ words}$$

Purely Statistical Approaches (Cont.)

- Historical Review
  - Warren Weaver's Suggestion to Machine Translation (1949)
    - Using coding and information theory (Claude Shannon)
  - Noam Chomsky's Debate (1956)
    - No finite-state Markov Process can serve as an English grammar
    - Finite State Markov Chain:

$$P(S_j \mid S_{j-1}, S_{j-2}, \cdots, S_2, S_1) = P(S_j \mid S_{j-1})$$

    - Examples from Chomsky (1956)
      - If S1, then S2.
      - Either S3, or S4
      - The man who said that S5, is arriving today.
  - Psychological Factor

Purely Statistical Approach to Machine Translation (IBM Model-1, 1988)

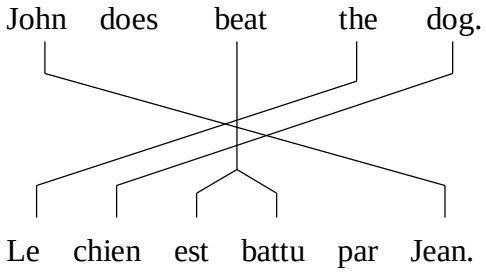
- Regard "Translation Mechanism" as a Decoding Process
  - Find $\arg\max[P(s|t)\cdot P(t)]$, where (s,t) is a source language and target language sentence pair.
- Use Trigram Language Model
  - Let $W_t = [wt_1, \cdots, wt_n]$ then assume

$$P(wt_1, \cdots, wt_n) = P(wt_1)\cdot P(wt_2 \mid wt_1)\prod_i P(wt_i \mid wt_{i-1}, wt_{i-2})$$

- Translation Model

$$\begin{aligned} P(s|t) = {} & P(f1=1|John)\cdot P(Jean|John)\cdot \\ & P(f2=0|does)\cdot \\ & P(f3=2|beat)\cdot P(est|beat)\cdot P(battu|beat)\cdot \\ & P(f4=1|the)\cdot P(le|the)\cdot \\ & P(f5=1|dog)\cdot P(chien|dog)\cdot \\ & P(f6=1|null)\cdot P(par|null) \end{aligned}$$

Purely Statistical Approach to Machine Translation (IBM Model)



Alignment of a translation pair.

Purely Statistical Approach to Machine Translation (IBM Model, Cont.)

- Example [Brown et al., 90]:
  - Find all possible translation lexicons for all words in source language
  - "bagged translation" (French-to-English)
    - Cut a sentence into words
    - Place the words in a bag
    - Try to recover the sentence by rearranging all possible sequences and using 3-gram probability to find the preferred one
  - Assume free word order (IBM-Model-1):
    - "John likes Mary." is as good as "Mary likes John."

Why Not Purely Statistical Approaches

- Parameter space is usually too large
  - Since arbitrary number of modifiers can be inserted between two constituents that are dependent to each other (e.g., NP and VP), the surface form is just too noisy
  - A very large corpus is required to get reliable statistics due to the large parameter space
- Can only deal with local dependency
  - Simplified models which consult less contextual information are used to make the parameter space manageable
  - The dependency relationship between two constituents (e.g., NP, VP) cannot be taken care when they are far apart (lacking tree structure to link head-lexicons)
- Suffer from robustness and sparse data problems
  - Performance usually drops significantly when tested in another domain
  - Frequently over-tuned into the given training-set (e.g., wrong model still gives good result after adaptive learning in aligning bi-lingual corpus)

Why Not Purely Statistical Approaches (Cont.)

- Not utilize the limited amount of data efficiently
  - Only learning knowledge from the corpus (the only knowledge source)
    - Brute force approach might need infeasible amount of data
    - Learning everything from scratch is not efficient
  - Abandoning existing knowledge (e.g., linguistic models) or resource (e.g., WordNet, Hownet) gives itself a handicap
- Not promising for pursuing high performance
  - Although even the wrong approach can still be tuned and improved, it has little hope that it can reach the final goal (say, providing over 90% of translation accuracy)
  - That is what Martin Kay has frequently claimed: "One wants to reach the moon. Without building the rocket, he is climbing the tree everyday; and then claims he is several meters higher each day".

Corpus-Based Statistics-Oriented (CBSO) Approaches (1)

- Establish a language model based on known linguistic knowledge
- Prepare a large corpus (possibly annotated) in order to acquire consistent knowledge implied in the corpus
- Acquire statistical “rules” (i.e., underlying regularity about randomness) from the corpus through statistical induction mechanism, namely, to learn model parameters automatically (or semi-automatically) from the training corpus
- Example: (Analysis Score or Score Function for Disambiguation)

$$
\begin{aligned}
& P(Sem, Syn, Lex \mid Words) \\
&= P(Sem \mid Syn, Lex, Words) \\
&\times P(Syn \mid Lex, Words) \\
&\times P(Lex \mid Words) \\
&\equiv S_{sem} \times S_{syn} \times S_{lex}
\end{aligned}
$$

Corpus-based Statistics-oriented (CBSO) Approaches (2)

- Express linguistic knowledge in terms of a large number of implicit "parameters" (e.g., P(verb|det) = 0)
  - Acquire such knowledge by converting the knowledge acquisition problem into a simple parameter estimation problem
  - Example: Tagging Model:

$$\text{argmax}_{t_1^n} P(t_1^n \mid w_1^n) = \text{argmax}_{t_1^n} \prod_i \cdots P(t_i \mid t_{i-1}) \cdots$$

$$\begin{array}{ccccc} W_1 & W_2 & W_3 & W_4 & \cdots \\ c_1 & c_2 & c_3 & c_4 & \cdots \\ & \cdots & & & \\ W_1' & W_2' & W_3' & W_4' & \cdots \\ c_1' & c_2' & c_3' & c_4' & \cdots \end{array} \quad \Rightarrow \quad P(\text{verb}|\text{det}) = \frac{\#[\text{det verb}]}{\#[\text{det}]}$$

Corpus-based Statistics-oriented (CBSO) Approaches (3)

- Why linguistics models are still needed (c.f. purely statistical approaches):
  - They provide the information of 'equivalent classes' (in known linguistics knowledge)
  - As a result, the number of statistical parameters required to describe linguistics phenomena can be drastically reduced
    - e.g., syntactic behavior of a new word (suspected to be a noun) can be predicted from the existing noun model
  - e.g., Trigram model for parts of speech disambiguation:
    - ➪ CBSO: (100) ** 3 = 10 ** 6 parameters
    - ↔ Purely statistical ~= 100,000 (words) ** 3 = 10 ** 15

Corpus-based Statistics-oriented (CBSO) Approaches (4)

- Advantages:
  - Easy to acquire the required fine-grained knowledge in terms of millions of parameters
  - Easy for domain adaptation (simply by estimating a different set of parameters for a different domain)
  - Existing linguistic features (and linguistic knowledge) can be utilized and long distance dependency phenomenon in language can be handled

A Comparative Review

- Rule-Based Approaches
  - Heuristic Rule: a "determiner" can not be followed by a "verb"
- Purely Statistical Approaches

x	y	p(y|x)
the	ate	0
a	took	0
an	walks	0
	buy	0

  - Conclusion: a word in x can not be followed by a word in y
  - #parameters: 10 ** 15; can only handle local dependency
- CBSO Approaches

$$\begin{array}{cccccc} W_1 & W_2 & W_3 & W_4 & \cdots & \\ c_1 & c_2 & c_3 & c_4 & \cdots & \Rightarrow \quad P(\text{verb}|\text{det}) = 0 \end{array}$$

  - #parameters: 10 ** 6; can handle long-distance dependency

Why Corpus-Based Statistics-Oriented NLP

- Inherit the good properties of objectiveness, consistency, trainability, cost-effectiveness for statistical approaches
- Long distance dependency is manageable
  - The higher level dependency relationship can transform (or map) the noisy (surface form) space into another clean space to greatly reduce the number of parameters required
  - CBSO stochastic models are established on top of non-terminal symbols (e.g., NP, VP), not on terminal symbols (i.e., word strings)
  - More contextual information can be used
  - Syntactic and semantic information could be consulted

Why CBSO NLP (Cont.)

- Sparse data problem is less severe
  - Intermediate forms can be introduced to reduce the parameters required (computational requirement is factorized: n1xn2xn3 becomes n1+n2+n3)
  - Parameter space is small with respect to purely statistical approaches
- Easy to meet the desirable designing goals of wide coverage, robustness, adaptability, controllability, parameterization and cost-effectiveness

Typical CBSO Approach (1)

- Decide appropriate features to be extracted from observations
  - What to use for solving the problem (e.g., ambiguity resolution)
  - This step is usually the most important step
- Adopt suitable statistical language model: Probabilistic Form
  - Probabilistic form characterizes the relationship among features to reflect problem characteristics and make computation feasible
  - Knowledge is implicitly implied by (or distributed in) by a large number of parameters
- Parameters Estimation & Learning (Adjusting) Process
  - Obtain a specific set of model parameters that can maximize the desired performance criterion

Typical CBSO Approach (2)

- Solution Search: Basically a generate-and-test process
  - First generate a set of possible candidates (e.g., look up all associated tags from lexicon database, or find all matched production-rules, etc.)
  - Form the associated feature vector for each candidate (e.g. part-of-speech trigram and lexicon-part-of-speech for trigram tagging model)
  - Use the adopted statistical language model to evaluate the score

Typical CBSO Approach (3)

- Beam Search is usually required
  - Adopt Soft-Rejection; therefore, every candidate has a score
  - The number of possible candidates frequently increases explosively (combinatory explosion), and pruning is usually necessary.
- Select Best-N Candidates
  - Select Best-N candidates based on assigned probability score in each stage (e.g., tokenization module would send top-2 token-sequences to parser, etc.)
  - "N" could be decided by Beam-Width, Static Threshold, and Dynamic Threshold (e.g., how far a candidate deviates from the best one)

Typical Form for Ambiguity Resolution

- Typical form for Ambiguity Resolution: (Bayesian)

$$BestCandidate = \arg\max_{Candidate} P(Candidate \mid Features)$$

- Hidden Random Variables are frequently introduced
  - Introducing intermediate form can simplify the problem

$$BestCandidate = \arg\max_{Candidate} \sum_{IR} P(Candidate, IR \mid Features)$$

  - IR: Intermediate Representations (Hidden Linguistic Structures)
  - One IR for each level of abstraction (e.g., POS tags, parse trees, semantic normal forms)
  - Simplification can be made over the hidden variables by applying chain rule and independency assumptions: e.g.,

$$P(Sem, Syn, Lex \mid Words) \approx P(Sem \mid Syn) \times P(Syn \mid Lex) \times P(Lex \mid Words)$$
$$\equiv \hat{S}_{sem} \times \hat{S}_{syn} \times \hat{S}_{lex}$$

Typical Form for Ill-Formedness Recovery

- Why Ill-formedness Recovery
  - Traditional fail-soft approach (usually select the longest fragment when the process is forced to stop) has following drawbacks:
    - Fail to deliver satisfactory result : usually keep the same word order (as no further processing can be performed)
    - Complicate the subsequent processing
  - Quality can be improved, if an ill-formed candidate can be recovered in early stages (so that more appropriate actions can be applied in successive phases). For example, “which one” would be translated into “哪一個” (instead of “哪個一”).
  - Simplify the design of the successive phases: only deal with a few expected normal forms (a close set), instead of numerous unexpected forms (an open set)
  - Examples: Guess associated tags and semantic classes missing from lexicon database, generate most likely parse trees for those sentence not covered by the adopted grammar (e.g., “which one?”)

Typical Form for Ill-Formedness Recovery (Cont.)

- Typical procedure for Ill-formedness Recovery
  - Detect ill-formed situation
  - Generate a set of possible well-formed candidates
  - Use the adopted statistical model to evaluate the possibility of each well-formed candidate
  - Select the most likely Best-N well-formed candidates for further processing
- Typical form for Ill-formedness Recovery [Lin 99]

$$RecoveredCandidate = \arg\max_{Well-FormedCandidate} P(Well\text{-}FormedCandidate \mid Ill\text{-}FormedCandidate)$$