Inducing Probabilistic Grammars by Bayesian Model Merging

Andreas Stolcke, Stephen Omohundro

Proceedings of the Second International ICGI Colloqium on Grammatical Inference and Applications, volume 862, Lecture Notes on Artificial Intelligence, Berlin, September 1994

Roadmap

- Problem Statement
- What makes this problem important (and interesting)?
- Solutions proposed earlier
- Bayesian Model Merging
- Experimental Results
- Critique

Problem Statement

- A *deterministic grammar* consists of a set of rules, that define a language.
- A *stochastic grammar*, in addition, consists of a set of probabilities.
- It describes how *likely* a string is, in a language.
- Given a set of samples from a distribution, can we construct a stochastic grammar for the distribution?

What makes this problem important?

- Speech Recognition
- Handwriting Recognition
- Natural Language Processing
- Biological Sequence Processing
- and so on ...
- What makes this problem interesting?
  Given the structure, the parameters can be learned using standard techniques (e.g., Maximum Likelihood Estimation). Learning the structure is much harder.

Solutions Proposed Earlier

- **Successive State Splitting (HMMs)**

  1. Bell, Timothy C., John G. Cleary, Ian H. Whitten. *Text Compression*. Englewood Cliffs, N.J., Prentice Hall, 1990.
  2. Ron, Dana, Yoram Singer, Naftali Tishbi. The power of amnesia. *Advances of Neural Information Processing Systems, 6*, Morgan Kauffman, San Mateo, CA, 1994.

- **Stochastic Context Free Grammars**

  1. Cook, Craig M., Azriel Rosenfeld, Alan R. Aronson. Grammatical inference by hill climbing. *Information Sciences*, volume 10, pages 59 – 80, 1976.

Bayesian Model Merging

- A model building process that learns the structure.
- Two state process:
  1. **Data Incorporation.** Begin with a model that generates only the training set.
  2. **Model Merging.** Progessively merge states in that model, moving towards models that are more general and compact.

Example: Hidden Markov Model

- Language: $L = (ab)^+$.
- Training set: $X = \{ab, abab\}$.
- Step 1: Data Incorporation

$$S \to aS_1, S_1 \to bS_2, S_2 \to F$$

$$S \to aS_3, S_3 \to bS_4, S_4 \to aS_5, S_5 \to bS_6, S_6 \to F$$

Example: Hidden Markov Model

- Step2: Merging
  Merge S_1, S_3:

$$S \rightarrow aS_1, S_1 \rightarrow bS_2, S_2 \rightarrow F$$

$$S_1 \rightarrow bS_4, S_4 \rightarrow aS_5, S_5 \rightarrow bS_6, S_6 \rightarrow F$$

  Merge S_2, S_4:

$$S \rightarrow aS_1, S_1 \rightarrow bS_2, S_2 \rightarrow F$$

$$S_2 \rightarrow aS_5, S_5 \rightarrow bS_6, S_6 \rightarrow F$$

Example: Hidden Markov Model

Merge S_6, S_2:

$$S \rightarrow aS_1, S_1 \rightarrow bS_2, S_2 \rightarrow F$$

$$S_2 \rightarrow aS_5, S_5 \rightarrow bS_2$$

Merge S_1, S_5:

$$S \rightarrow aS_1, S_1 \rightarrow bS_2, S_2 \rightarrow F$$

$$S_2 \rightarrow aS_1$$

Which States should I Merge?

- Merging aims at maximizing the probability of the model, given the data.
- According to Bayes Rule:

$$P(model|data) = \frac{P(data|model)P(model)}{P(data)}$$

- Use standard optimization techniques (Breadth First Search, Beam Search, etc.) to optimize $P(model|data)$

Example: Context Free Grammar

- Language: $L = \{a^n b^n | n > 0\}$
- Training set: $X = \{ab, aabb, aaabbb\}$
- Step 1: Data Incorporation

$$S \rightarrow AB|AABB|AAABBB$$

$$A \rightarrow a$$

$$B \rightarrow b$$

- Step 2: Chunk $(AB) \rightarrow X$:

$$S \rightarrow X|AXB|AAXBB$$

$$X \rightarrow B$$

Example: Context Free Grammar

- Step 3: Chunk $(AXB) \rightarrow Y$:

$$S \rightarrow X|Y|AYB$$

$$X \rightarrow AB$$

$$Y \rightarrow AXB$$

- Step 4: Merge S, Y:

$$S \rightarrow X|ASB$$

$$X \rightarrow AB$$

- Step 5: Merge S, X:

$$S \rightarrow AB$$

$$S \rightarrow ASB$$

Experimental Results

- **Hidden Markov Models.** BMM was compared to the Baum-Welch algorithm.

  1. For *formal regular languages*, BMM did better. Baum-Welch was strongly dependant on the initial parameter settings. It failed to generate the target grammar in more than half of the trials. BMM did not require any initial parameter settings and performed much better.
  2. For *real-life data*, phonetic pronunciation models from the TIMIT database was used. BMM models were slightly better than Baum-Welch models. But Baum-Welch models required twice as many transitions.

Experimental Results

- **Context Free Grammars.** BMM was compared to Cook's algorithm.
  1. All grammars that Cook's algorithm could generate could also be generated using BMM.
  2. BMM could generate the grammar for the language of palindromes, $L = \{ww^R | w \in \{a, b\}^*\}$, which Cook's algorithm can not.

Critique

- Strengths:
  1. The algorithm always attempts to find more *compact* models: follows Occam's razor.
  2. Outperforms previously known algorithms for learning both HMMs and CFGs.
- Weaknesses:
  1. Search methods used are computationally expensive. Could heuristic search procedures be used?
  2. Experimental setup needs to be explained in greater detail, for reproducibility. Stolcke's thesis gives the setup in greater detail.
  3. Many implementation details are skipped. Again, for reproducing the results, one must read the thesis.

Thank You