Treebank Training, Inside-Outside Estimation and EM Algorithm

Detlef Prescher

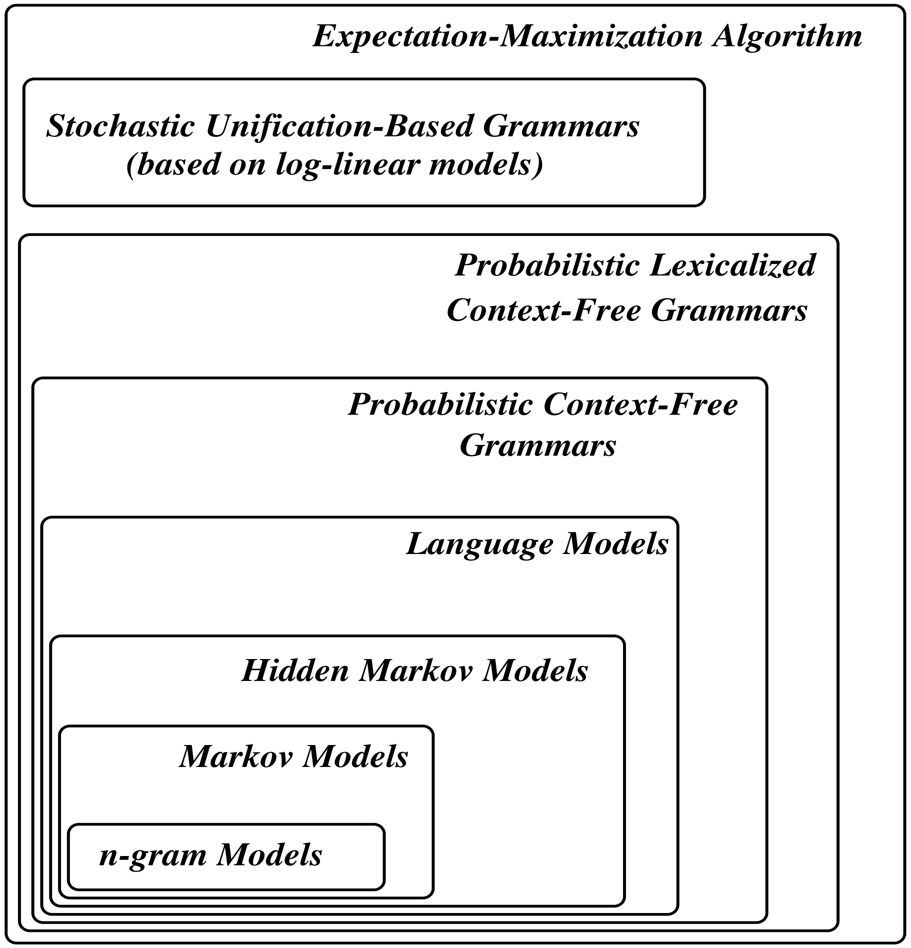
DFKI GmbH, Language Technology Lab
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
prescher@dfki.de

Januar 15, 2002

Overview

- Introduction: The Linguistic World of EM
- Brief Review: Training of Probabilistic Context-Free Grammars
  - History
  - Treebank Grammars
  - Inside-Outside Estimation
- The EM Algorithm
- Probabilistic Context-Free Grammars and EM
  - Treebank-Training as EM
  - Inside-Outside Estimation Meets Dynamic EM
- Conclusion

Introduction: (Part of) The Linguistic World of EM



History: Training of Probabilistic Context-Free Grammars

1972 **Forward-backward algorithm** for Hidden Markov Models (equivalent to training of regular grammars on free text) with convergence proofs (Baum et al)

1977 **EM algorithm** for training of symbolic analyzers on incomplete data (Dempster, Laird, Rubin); with wrong convergence proofs

1979 **Inside-outside algorithm** for training context-free grammars on free text (Baker); however without convergence proofs, and training corpora consisting of one single sentence

1983 **Convergence proofs** for the EM algorithm (Wu)

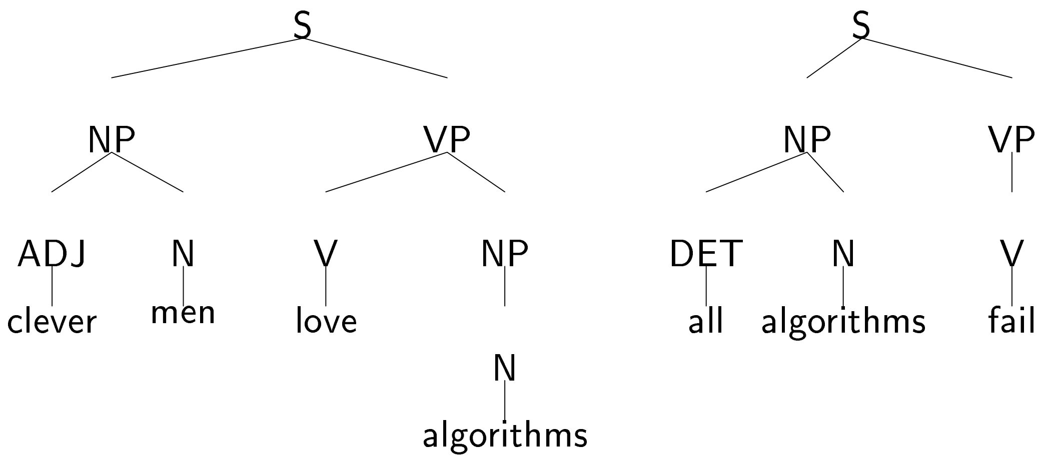
1990 **Inside-outside algorithm** extended to training corpora of arbitrary size (Lari and Young); still without convergence proofs

1995 **Treebank training** (Charniak) exploiting the Penn Treebank, a syntactically annotated corpus of English (Marcus, 1993); earlier approaches rely on the LOB corpus (Sampson et al, 1988)

Brief Review: Treebank Grammars

Treebank training: assign the **empirical probability** $p(r) := \frac{f(r)}{\sum_{r \in G_A} f(r)}$ to the grammar rule r. Here, $f(r)$ is the frequency of rule r occuring in the treebank, and G_A is the set of all rules with the same left-hand side A.

Example:



S	$\rightarrow$	NP VP	(1.0)
NP	$\rightarrow$	ADJ N	(0.33333)
NP	$\rightarrow$	DET N	(0.33333)
NP	$\rightarrow$	N	(0.33333)
VP	$\rightarrow$	V NP	(0.5)
VP	$\rightarrow$	V	(0.5)
DET	$\rightarrow$	all	(1.0)
ADJ	$\rightarrow$	clever	(1.0)
N	$\rightarrow$	men	(0.33333)
N	$\rightarrow$	algorithms	(0.66666)
V	$\rightarrow$	love	(0.5)
V	$\rightarrow$	fail	(0.5)

Note that it is possible to analyze sentences not occuring in the treebank e.g. *clever men fail (0.014)* or *clever algorithms fail (0.028)*.

Brief Review: Treebank Grammars

Properties of treebank grammars:

- Treebank training is the **standard method for supervised training of a context-free grammar**.

- **Treebank grammars are proper**: the sum of the probabilities of all grammatical sentences of the treebank grammar has not a probability less than one (Chi 1998, Booth and Thompson 1973).

- **Treebank training equals maximum-likelihood estimation**, i.e. competing probabilistic context-free grammars (respectively, competing assignments of rule probabilities) yield equal or lower treebank probabilities[1] than the treebank grammar itself — **if the competing grammars fulfill the stringent conditions of Booth and Thompson (1973)**.

[1] The **treebank probability** is the product of the probabilities of all trees occuring in the treebank

Brief Review: Treebank Grammars

Problems: treebank grammars are easy to handle in practice, however some difficulties with theory arise:

- **The easiest proof of the property, that treebank grammars are maximum-likelihhod estimates, relies on a deep theorem of information theory**, the so-called information inequality,
- **How is treebank training theoretically related to competing training methods for context-free grammars, especially to IO estimation?**

Our contribution:

- Treebank training inherits the properties of the EM algorithm for annotated data; this leads easily to the insight that treebank training is maximum-likelihood estimation,
- Treebank training is closely related to IO estimation, since both treebank training and inside-outside estimation are variants of the EM algorithm,

Brief Review: IO Estimation

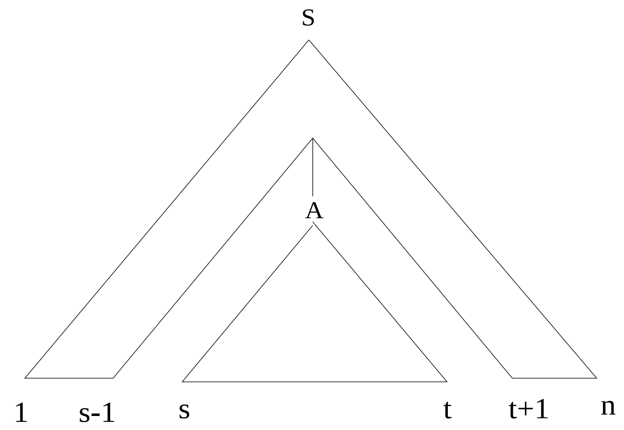
IO estimation (for a training corpus of sentences y):

$$\hat{p}(r) := \frac{\sum_y C_y(r)}{\sum_y C_y(A)} \qquad (A = \mathsf{lhs}(r))$$

using rule and category counts $C_y(r)$ and $C_y(A)$ based on **recursively computable inside and outside probabilities** associated to sentence spans.

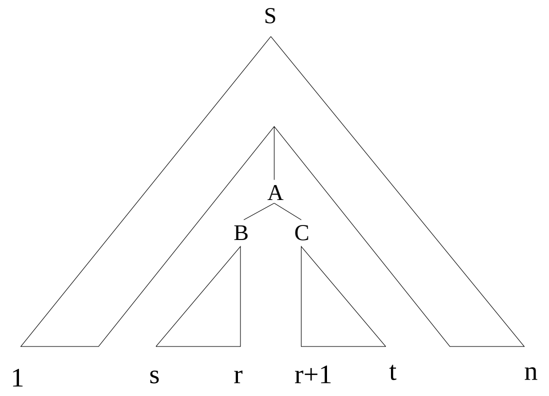
Definition:

- inside probabilities
  $e(s,t,A) := p(A \Rightarrow^* w_s \dots w_t)$
- outside probabilities
  $f(s,t,A) := p\left(S \Rightarrow^* w_1 \dots w_{s-1} A w_{t+1} \dots w_n\right)$



Brief Review: IO Estimation

Baker's variant: IO for a training corpus of one single sentence.



Using the sentence probability $P := p(S \Rightarrow^* w_1 \ldots w_n)$ and a grammar in CNF:

$$\hat{p}(A \to a) := \frac{\frac{1}{P}\sum_{1 \le t \le n,\ w_t = a} e(t, t, A) \cdot f(t, t, A)}{\frac{1}{P}\sum_{s=1}^{n}\sum_{t=s}^{n} e(s, t, A) \cdot f(s, t, A)} \ ,$$

and

$$\hat{p}(A \to BC) := \frac{\frac{1}{P}\sum_{s=1}^{n-1}\sum_{t=s+1}^{n}\sum_{r=s}^{t-1} p(A \to BC) e(s, r, B) e(r+1, t, C) f(s, t, A)}{\frac{1}{P}\sum_{s=1}^{n}\sum_{t=s}^{n} e(s, t, A) f(s, t, A)} \ .$$

Brief Review: IO Estimation

Properties of IO Estimation:

- The standard method for **unsupervised training** of a manually developed context-free grammar.
- In contrast to supervised training (treebank training), IO estimation is an **iterative procedure** (on free text), but
- **dynamic programming techniques** make IO estimation efficient (in contrast to state-of-the-art training methods for stochastic UBGs)

Problem: The theoretical behaviour of the **log-likelihood** (corpus probabilities resp.), while iteratively applying IO estimation:

- **monotonicity** has been observed in many experiments, but as well as
- **convergence to stationary points** (especially to local maxima) has never been formally proved.

Brief Review: IO Estimation

Our solution: Solve the problems with IO Estimation trusting **"common wisdom"**:

"**IO is a dynamic programming-variant of EM**".

I.e. put IO estimation into the context of the theoretically well-known **expectation-maximization algorithm** and formally prove the following

Theorem: The results of EM, when applied to PCFGs, equal the results of IO estimation. Thus:

- IO inherits the **well-known convergence behaviour of EM**,
- EM for PCFGs inherits the **dynamic behaviour of IO**.

Expectation-Maximization Algoprithm

The EM algorithm augments the rules of an existing symbolic analyzer with probability weights and exploits for this purpose a corpus of analysable data.

EM input:

- a big corpus of so-called **incomplete data** (e.g. words or sentences)
- a so-called **symbolic analyzer** (e.g. a tagger or parser) generating for each incomplete data type a set of so-called **complete data** types (e.g. word-tag pairs or syntax trees)

EM output: a so-called **stochastic analyzer** weighting the analyses of the given symbolic analyzer (e.g. yielding most probable tags or trees).

Thus, the typical EM applications are: Ranking, disambiguation, or informed use (e.g. in IR applications) of symbolically generated analyses.

Expectation-Maximization Algorithm

Input: a training corpus $\tilde{p}(.)$ of incomplete data, a symbolic analyzer $X(.)$, and a family of parameterized probability models $p_\Theta(.)$ on complete data

Output: (for each starting model p_{Θ_0}) a sequence of trained models $p_{\Theta_1}, \; p_{\Theta_2}, \; p_{\Theta_3}, \; \ldots$ with montonically increasing log-likelihood values $L(p_{\Theta_1}) \leq L(p_{\Theta_2}) \leq L(p_{\Theta_3}) \leq \ldots$

Procedure:

1. **for each iteration** $i := 1, 2, 3 \ldots$ **do**
2. $\quad \Phi := \Theta_{i-1}$;
3. $\quad$ **(E-step) compute** auxiliary function (for all Θ)
$$Q_\Phi(\Theta) = \sum_{y \in \mathcal{Y}} \tilde{p}(y) \sum_{x \in X(y)} p_\Phi(x|y) \cdot \log \; p_\Theta(x);$$
4. $\quad$ **(M-step) compute** the global maximum of $Q_\Phi(.)$
$$\hat{\Theta} = \mathrm{argmax}_\Theta \;\; Q_\Phi(\Theta);$$
5. $\quad \Theta_i := \hat{\Theta}$;
6. **print** $\Theta_0, \; \Theta_1, \; \Theta_2, \; \Theta_3, \; \ldots$;

Expectation-Maximization Algorithm

Our contribution to EM theory is the insight that the EM algorithm equals a sequence of maximum-likelihood estimations:

3. **generate** current corpus of complete data $\tilde{p}_\Phi(x)$;
4. **compute** maximum-likelihood estimate on this corpus
$$\hat{\Theta} = \text{argmax}_\Theta \;\; L_\Phi(\Theta);$$

The key step of this modification of the EM algorithm (E plus M step) is the introduction of the **current corpus of complete data**:

$$\tilde{p}_\Phi(x) := \tilde{p}(y) \cdot p_\Phi(x|y) \quad \text{ with } y \in \mathcal{Y} \text{ such that } x \in X(y) \ ,$$

since its **log-likelihood function** $L_\Phi(\Theta) := \sum_{x \in \mathcal{X}} \tilde{p}_\Phi(x) \cdot \log p_\Theta(x)$ equals the auxiliary function of the EM algorithm:

$$L_\Phi(\Theta) \ = \ Q_\Phi(\Theta) \ .$$

Expectation-Maximization Algorithm

Properties: The (generalized) EM algorithm has a good convergence behaviour. In detail: the log-likelihood values of the re-estimated parameters

- monotonically inrease,
- converge to a stationary value of the log-likelihood function (i.e. a local minimum, saddle point, or maximum),
- converge in some cases even to a local maximum (but in general not to a global maximum).

Problem: No dynamic-programming techniques are known for the general EM algorithm. Some variants of the EM algorithm may be very slow.

Our solution: Show that the special case of EM for PCFG's combines the advantages of both the general EM algorithm and the IO algorithm.

Probabilistic Context-Free Grammars and EM

Our contribution: (generalized) EM estimation for PCFG's yields:

$$\hat{p}(r) = \frac{\sum_y p(.|y)\,[\ f_r\ ]}{\sum_y p(.|y)\,[\ f_A\ ]} \qquad (A = \mathsf{lhs}(r))$$

using **expected frequencies**:

$$p(.|y)\,[\ f_r\ ] := \sum_{x \in X(y)} p(x|y) \cdot f_r(x)\ .$$

Simple special case: the (generalized) EM algorithm for annotated data, i.e. for $X(y) = \{y\}$, equals treebank training:

$$\hat{p}(r) = \frac{\sum_y f_r(y)}{\sum_y f_A(y)} = \frac{f(r)}{f(A)} = \frac{f(r)}{\sum_{r \in G_A} f(r)} \qquad (A = \mathsf{lhs}(r))$$

Treebank Training as EM

Our contribution: some theoretical statements to treebank training.

- Treebank training inherits the properties of the (generalized) EM algorithm for annotated data; Especially: the log-likelihhod function monotonically increases, i.e.

$$L(\hat{p}) \geq L(p) \quad \text{for all } p \ .$$

  This is a new proof that treebank training equals maximum-likelihood estimation.

- More important: Treebank training is closely related to IO estimation, since both treebank training and inside-outside estimation are variants of the EM algorithm.

IO Estimation Meets Dynamic EM

IO estimation (for a training corpus of sentences y):

$$\hat{p}(r) := \frac{\sum_y C_y(r)}{\sum_y C_y(A)} \qquad (A = \mathsf{lhs}(r))$$

using rule and category counts $C_y(r)$ and $C_y(A)$ based on **recursively defined inside and outside probabilities** associated to sentence spans.

EM estimation for PCFGs:

$$\hat{p}(r) = \frac{\sum_y p(.|y)\,[\ f_r\ ]}{\sum_y p(.|y)\,[\ f_A\ ]} \qquad (A = \mathsf{lhs}(r))$$

using **expected frequencies**: $p(.|y)\,[\ f_r\ ] := \sum_x p(x|y) \cdot f_r(x)$.

Key step: resolve non-dynamics of EM proving $p(.|y)\,[\ f_r\ ] = C_y(r)$.

Conclusion

We have presented:

- a brief review of treebank, IO and EM estimation for PCFGs,
- a formal proof relating treebank training to EM and IO estimation
- a formal proof confirming "common wisdom" that **IO is a dynamic-programming variant of EM**,
- **a formal proof of the convergence behaviour of IO estimation** which is omitted in the original literature.
- **a formal proof of the dynamic behaviour of the EM algorithm for PCFG's**