Unsupervised Natural Language Parsing

Kewei Tu, ShanghaiTech University
Yong Jiang, Alibaba DAMO Academy
Wenjuan Han, National University of Singapore
Yanpeng Zhao, University of Edinburgh

March 29, 2021

2. Generative Approaches:
Parameter Learning

Motivation

Structure learning is hard!

Motivation

Structure learning is hard!
Another solution is to directly perform parameter learning with a fixed structure (e.g., enumerating all possible rules).

- Most research on parameter learning focuses on DMV, a form of generative dependency grammars.
- These parameter learning methods are generally applicable for PCFGs.

Motivation

Structure learning is hard!
Another solution is to directly perform parameter learning with a fixed structure (e.g., enumerating all possible rules).

- Most research on parameter learning focuses on DMV, a form of generative dependency grammars.
- These parameter learning methods are generally applicable for PCFGs.
- Before introducing the DMV model, let us review a preliminary model: HMM.

Unsupervised POS Tagging with HMM

- Goal: find syntax clusters for each word in a sentence.

Unsupervised POS Tagging with HMM

- Goal: find syntax clusters for each word in a sentence.
- a.k.a. POS induction: give each word x_i a label z_i for sentence x.

Unsupervised POS Tagging with HMM

- Goal: find syntax clusters for each word in a sentence.
- a.k.a. POS induction: give each word x_i a label z_i for sentence x.

General approach for POS induction: Hidden Markov models (HMMs)

- Start probabilities: $P(\mathrm{z}_1 | \mathrm{z}_0 = START)$
- Transition probabilities: $P(\mathrm{z}_i | \mathrm{z}_{i-1})$
- Emission probabilities: $P(\mathrm{x}_i | \mathrm{z}_i)$

Unsupervised POS Tagging with HMM

- Goal: find syntax clusters for each word in a sentence.
- a.k.a. POS induction: give each word x_i a label z_i for sentence x.

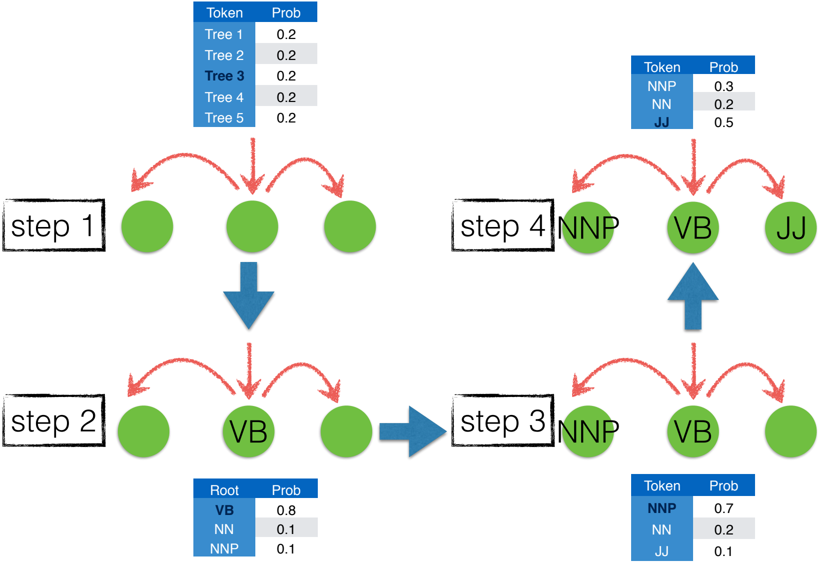
General approach for POS induction: Hidden Markov models (HMMs)

- Start probabilities: $P(\mathrm{z}_1 | \mathrm{z}_0 = START)$
- Transition probabilities: $P(\mathrm{z}_i | \mathrm{z}_{i-1})$
- Emission probabilities: $P(\mathrm{x}_i | \mathrm{z}_i)$

Example: To generate a sentence I SWAM with POS sequence PRONOUN VERB.

$$
\begin{aligned}
&P(\mathrm{x}, \mathrm{z}) = P(\mathrm{z}_1 = Pronoun | \mathrm{z}_0 = START) \\
&\cdot P(\mathrm{z}_2 = Verb | \mathrm{z}_1 = Pronoun) \cdot P(\mathrm{x}_1 = I | \mathrm{z}_1 = Pronoun) \\
&\cdot P(\mathrm{x}_2 = swam | \mathrm{z}_2 = Verb)
\end{aligned}
$$

Old Dependency Models



[Paskin (2002) and Carroll & Charniak (1992)]

DMV with an example

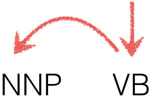
step 1

Token	Prob
VB	0.8
NN	0.1
NNP	0.1

step 2

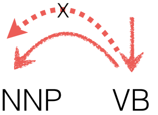
Decision	Prob
CONT	0.9
STOP	0.1

step 3



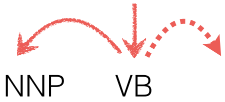
Token	Prob
NNP	0.7
NN	0.2
JJ	0.1

step 4



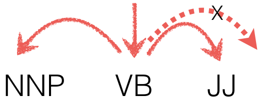
Decision	Prob
CONT	0.3
STOP	0.7

step 5



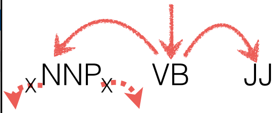
Decision	Prob
CONT	0.9
STOP	0.1

step 6/7



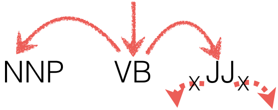
Token	Prob
NNP	0.3
NN	0.2
JJ	0.5
Decision	Prob
CONT	0.2
STOP	0.8

step 8/9



Decision	Prob
CONT	0.1
STOP	0.9
Decision	Prob
CONT	0.2
STOP	0.8

step 10/11



Decision	Prob
CONT	0.4
STOP	0.6
Decision	Prob
CONT	0.3
STOP	0.7

DMV Model Representation (Klein & Manning, ACL 2004)

Formal Definition (Dependency Model with Valence):

- Sentence x, parse tree z, model joint probability $P(\mathrm{x}, \mathrm{z}; \Theta)$
- three kinds of grammars rules: root, attach and decision.
- $dir(h, c)$: dependency direction from parent token h to child token c.
- $val(h)$ indicates valency: whether h has already generated a child.

DMV Model Representation (Klein & Manning, ACL 2004)

Formal Definition (Dependency Model with Valence):

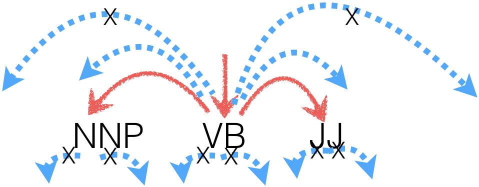
- Sentence x, parse tree z, model joint probability $P(\mathrm{x}, \mathrm{z}; \Theta)$
- three kinds of grammars rules: root, attach and decision.
- $dir(h, c)$: dependency direction from parent token h to child token c.
- $val(h)$ indicates valency: whether h has already generated a child.

Rule Schema:

- Root: $p_{root}(c)$
- Attach: $p_{attach}(c|h, dir)$
- Decision: $p_{decision}(CONT|h, dir, val), p_{decision}(STOP|h, dir, val)$

DMV for Computing Sentence & Parse Probability

$$P(\mathrm{x}, \mathrm{z}; \Theta) = p_{root}(r(\mathrm{x}, \mathrm{z})) \times$$
$$\prod_{(h,c) \in \mathrm{z}} (p_{attach}(c|h, dir) p_{decision}(STOP|h, dir, val)$$
$$\times \prod_{dir \in \{\leftarrow, \rightarrow\}} p_{decision}(CONT|h, dir, val))$$



Extensions of DMV

- Headden III et al. (2009) introduced the valence into the condition of attach sampling.

$$p_{attach}(c|h, dir) \Rightarrow p_{attach}(c|h, dir, val)$$

Extensions of DMV

- Headden III et al. (2009) introduced the valence into the condition of attach sampling.

$$p_{attach}(c|h, dir) \Rightarrow p_{attach}(c|h, dir, val)$$

- Spitkovsky et al. (2012) conditioned decision and child token generation on sibling words, sentence completeness, and punctuation context.

Extensions of DMV

- Headden III et al. (2009) introduced the valence into the condition of attach sampling.

$$p_{attach}(c|h, dir) \Rightarrow p_{attach}(c|h, dir, val)$$

- Spitkovsky et al. (2012) conditioned decision and child token generation on sibling words, sentence completeness, and punctuation context.
- Yang et al. (2020) proposed a second-order extension of DMV that incorporates grandparent-child or sibling information (p here).

$$p_{attach}(c|h, dir) \Rightarrow p_{attach}(c|h, p, dir, val)$$

Three Different Models of Representing Rule Probabilities

Methods	Representation (Parameters: Θ)
Table-based	$p(c \mid h, dir)$
Feature-based [B-K et al. 2010]	$p(c \mid h, dir) \propto \mathrm{w}^T \mathrm{f}(h, c, dir)$
Neural-based [Jiang et al. 2016]	$p(c \mid h, dir) = softmax_c(f(h, dir))$

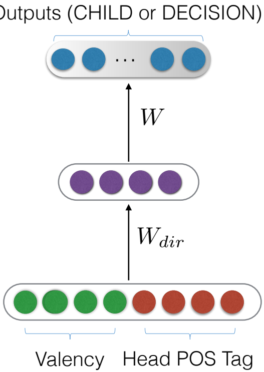
Token	Prob
NNP	0.7
NN	0.2
JJ	0.1

Table-based model

[0.8, -0.2, 0.3,0.4] . [0, 1, 0, 1]

$p \propto$

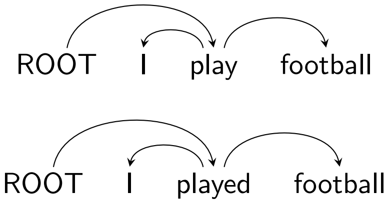
Feature-based model



Neural-based model

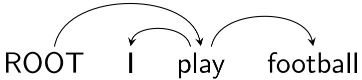
Differences

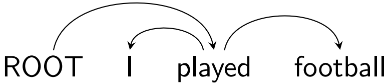
Drawbacks of Table-based methods: Symbols are independent with each other. However, some words behave alike in parsing.



Differences

Drawbacks of Table-based methods: Symbols are independent with each other. However, some words behave alike in parsing.





The feature-based and neural-based methods can tackle this problem:

- Utilizing hand-crafted sparse features (log-linear model).
- Neuralize the grammar.

Learned Correlations of POS tags for Neural-based Models

[Jiang et al. 2016]

Comparisons

Methods	Pros	Cons
Table-based	Simple parameter learning	Independent symbols.
Feature-based	Modeling symbol similarity	Need mannual-designs
Neural-based	Automatic learned similarity	-

Supervised Parameter Estimation

Given a set of annotated sentences $\mathcal{X} = \mathrm{x}^1, \mathrm{x}^2, \ldots, \mathrm{x}^N$ and parses $\mathcal{Z} = \mathrm{z}^1, \mathrm{z}^2, \ldots, \mathrm{z}^N$, how to learn parameters Θ of generative grammars.

$$J(\Theta) = -\frac{1}{N}\sum_{d=1}^{N} \log \underbrace{P(\mathrm{x}^d, \mathrm{z}^d)}_{\textbf{joint probability}} = -\frac{1}{N}\sum_{d=1}^{N} \log \underbrace{\sum_{\mathrm{r} \in \mathcal{R}(\mathrm{x}^d, \mathrm{z}^d)} p(\mathrm{r})}_{\textbf{rule factorization}}$$

where $p(\mathrm{r})$ is normalized.

Supervised Parameter Estimation

Given a set of annotated sentences $\mathcal{X} = \mathrm{x}^1, \mathrm{x}^2, \ldots, \mathrm{x}^N$ and parses $\mathcal{Z} = \mathrm{z}^1, \mathrm{z}^2, \ldots, \mathrm{z}^N$, how to learn parameters Θ of generative grammars.

$$J(\Theta) = -\frac{1}{N} \sum_{d=1}^{N} \log \underbrace{P(\mathrm{x}^d, \mathrm{z}^d)}_{\textbf{joint probability}} = -\frac{1}{N} \sum_{d=1}^{N} \log \underbrace{\sum_{\mathrm{r} \in \mathcal{R}(\mathrm{x}^d, \mathrm{z}^d)} p(\mathrm{r})}_{\textbf{rule factorization}}$$

where $p(\mathrm{r})$ is normalized.

- Table-based: parameter estimation based on cooccurrence counts. The following is an Attach rule example:

$$p(c|h, \rightarrow) = \frac{count(c, h, \rightarrow)}{count(h, \rightarrow)}$$

- Feature-based & Neural-based: parameter estimation based on gradiend-based algorithms.

Unsupervised Learning

Given a set of unannotated sentences $\mathcal{X} = \mathrm{x}^1, \mathrm{x}^2, \ldots, \mathrm{x}^N$, how to learn parameters Θ.

- Maximum likelihood estimation:

$$J(\Theta) = -\frac{1}{N}\sum_{d=1}^{N} \log P(\mathrm{x}^d) = -\frac{1}{N}\sum_{d=1}^{N} \log \underbrace{\sum_{\mathrm{z}^d} P(\mathrm{x}^d, \mathrm{z}^d)}_{\textbf{marginalized likelihood}}$$

$$= -\frac{1}{N}\sum_{d=1}^{N} \log \sum_{\mathrm{z}^d} \prod_{\mathrm{r} \in \mathcal{R}(\mathrm{x}^d, \mathrm{z}^d)} p(\mathrm{r})$$

Unsupervised Learning

Given a set of unannotated sentences $\mathcal{X} = \mathrm{x}^1, \mathrm{x}^2, \ldots, \mathrm{x}^N$, how to learn parameters Θ.

- Maximum likelihood estimation:

$$J(\Theta) = -\frac{1}{N}\sum_{d=1}^{N}\log P(\mathrm{x}^d) = -\frac{1}{N}\sum_{d=1}^{N}\log \underbrace{\sum_{\mathrm{z}^d} P(\mathrm{x}^d, \mathrm{z}^d)}_{\textbf{marginalized likelihood}}$$

$$= -\frac{1}{N}\sum_{d=1}^{N}\log \sum_{\mathrm{z}^d} \prod_{\mathrm{r}\in\mathcal{R}(\mathrm{x}^d,\mathrm{z}^d)} p(\mathrm{r})$$

- Here $p(\mathrm{r})$ can be parameterized by table-based, feature-based or neural-based methods.
- EM algorithm to optimize the objective function.

Unsupervised Learning: EM algorithms-Part 1

For unsupervised learning, a guess-and-update approach is utilized.

$$
\begin{aligned}
\log P(\mathrm{x};\Theta) &= \sum_{\mathrm{z}} q(\mathrm{z}) \log P(\mathrm{x};\Theta) \\
&= \sum_{\mathrm{z}} q(\mathrm{z}) \log \frac{P(\mathrm{x},\mathrm{z};\Theta)}{P(\mathrm{z}|\mathrm{x};\Theta)} \\
&= \sum_{\mathrm{z}} q(\mathrm{z}) \log \frac{P(\mathrm{x},\mathrm{z};\Theta)}{q(\mathrm{z})} \frac{q(\mathrm{z})}{P(\mathrm{z}|\mathrm{x};\Theta)} \\
&= \sum_{\mathrm{z}} q(\mathrm{z}) \log \frac{P(\mathrm{x},\mathrm{z};\Theta)}{q(\mathrm{z})} + \mathrm{KL}(q(\mathrm{z})||P(\mathrm{z}|\mathrm{x};\Theta)) \\
&\geq \sum_{\mathrm{z}} q(\mathrm{z}) \log \frac{P(\mathrm{x},\mathrm{z};\Theta)}{q(\mathrm{z})}
\end{aligned}
$$

Unsupervised Learning: EM algorithms-Part 2

We can obtain a new objective function:

$$
\begin{aligned}
J'(\Theta, Q(\mathrm{z})) &= -\sum_{d=1}^{N}\Big(\sum_{\mathrm{z}^d} q(\mathrm{z}^d)\log\frac{P(\mathrm{x}^d,\mathrm{z}^d;\Theta)}{q(\mathrm{z}^d)}\Big) \\
&= J(\Theta) + \sum_{d=1}^{N}\mathrm{KL}(q(\mathrm{z}^d)||P(\mathrm{z}^d|\mathrm{x}^d;\Theta))
\end{aligned}
$$

Unsupervised Learning: EM algorithms-Part 2

We can obtain a new objective function:

$$
\begin{aligned}
J'(\Theta, Q(\mathrm{z})) &= -\sum_{d=1}^{N}\Big(\sum_{\mathrm{z}^d} q(\mathrm{z}^d)\log\frac{P(\mathrm{x}^d,\mathrm{z}^d;\Theta)}{q(\mathrm{z}^d)}\Big) \\
&= J(\Theta) + \sum_{d=1}^{N}\mathrm{KL}(q(\mathrm{z}^d)||P(\mathrm{z}^d|\mathrm{x}^d;\Theta))
\end{aligned}
$$

- E-step, fix Θ, optimize $q(\mathrm{z}^d)$:

$$
\begin{aligned}
\arg\min_{Q(\mathrm{z})} J'(\Theta, Q(\mathrm{z})) &= \arg\min_{Q(\mathrm{z})}\sum_{d=1}^{N}\mathrm{KL}(q(\mathrm{z}^d)||P(\mathrm{z}^d|\mathrm{x}^d;\Theta)) \\
\Rightarrow q(\mathrm{z}^d) &= P(\mathrm{z}^d|\mathrm{x}^d;\Theta)
\end{aligned}
$$

- M-step, fix $q(\mathrm{z}^d)$, optimize Θ:

$$
J'(\Theta, Q(\mathrm{z})) = -\sum_{d=1}^{N}\Big(\sum_{\mathrm{z}^d} q(\mathrm{z}^d)\log\frac{P(\mathrm{x}^d,\mathrm{z}^d;\Theta)}{q(\mathrm{z}^d)}\Big)
$$

Unsupervised Learning: EM algorithms-Part 3

For table-based probabilistic grammars:

- E-step: utilize dynamic programming (the inside-outside algorithm) to compute a vector of expected frequencies.

$$e(r, \mathrm{x}) = E_{q(\mathrm{z})} c(\mathrm{r}, \mathrm{x}, \mathrm{z})$$

- M-step: update Θ using expected frequencies.

$$p(r) \propto \sum_{\mathrm{x}} e(r, \mathrm{x})$$

Unsupervised Learning: Online EM algorithms

Cons of EM algorithm:

- EM algorithm is a batch-style algorithm, suffers from slow convergence.
- If we have large-scale unlabeled data, performing EM algorithm is very time-consuming.

Online EM algorithm provides significant speed-up.

Unsupervised Learning: Online EM algorithms

Cons of EM algorithm:

- EM algorithm is a batch-style algorithm, suffers from slow convergence.
- If we have large-scale unlabeled data, performing EM algorithm is very time-consuming.

Online EM algorithm provides significant speed-up.

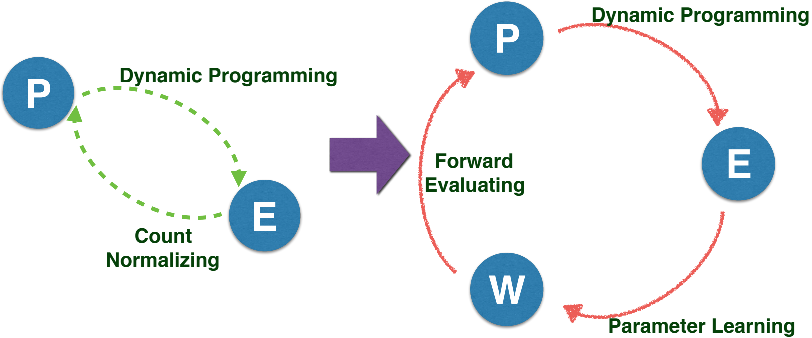
Key ideas

- Updating parameters (in M-step) after running E-step on a mini-batch of samples rather than the entire corpus.
- During E-step, interpolating the q distribution with distributions from previous steps.

[Liang et al. 2009]

Unsupervised Learning: Modified EM algorithms

EM algorithm $\rightarrow$ Modified EM algorithm (for feature/neural-based methods).



[B-K et al. 2010, Jiang et al. 2016]

Unsupervised Learning: Direct Gradient Descent

Another approach is directly computing the gradient:

$$\begin{aligned}\nabla_\Theta \log P(\mathrm{x}) &= \sum_{\mathrm{z}} P(\mathrm{z}|\mathrm{x};\Theta)\nabla_\Theta \log P(\mathrm{x},\mathrm{z};\Theta) \\ &= \sum_{\mathrm{z}} P(\mathrm{z}|\mathrm{x};\Theta) \sum_{\mathrm{r}\in\mathcal{R}(\mathrm{x}^d,\mathrm{z}^d)} c(\mathrm{r},\mathrm{x},\mathrm{z})\nabla_\Theta \log p(\mathrm{r}) \\ &= \sum_{r\in\mathcal{R}(\mathrm{x}^d,\mathrm{z}^d)} e(\mathrm{r},\mathrm{x})\nabla_\Theta \log p(\mathrm{r})\end{aligned}$$

A trick from [Eisner. 2016]: we can use back-propagation to calculate the expected frequencies $e(\mathrm{r},\mathrm{x})$.

$$e(\mathrm{r},\mathrm{x}) = \frac{\partial \log P(\mathrm{x};\Theta)}{\partial \log p(\mathrm{r};\Theta)}$$

No need for the outside algorithm.

Problems

- MLE objective only aims to explain the training data, which lacks of inductive bias.
- Local optima problem.

Improvements for MLE: Maximum A Posteriori

$$J(\Theta) = -\log P(\Theta|\mathrm{X}) \propto -\sum_{d=1}^{N} \log \underbrace{\sum_{\mathrm{z}^d} P(\mathrm{x}^d, \mathrm{z}^d|\Theta)}_{\textbf{marginalized likelihood}} - \underbrace{\log P(\Theta)}_{\textbf{prior}}$$

Improvements for MLE: Maximum A Posteriori

$$J(\Theta) = -\log P(\Theta|\mathrm{X}) \propto -\sum_{d=1}^{N} \log \underbrace{\sum_{\mathrm{z}^d} P(\mathrm{x}^d, \mathrm{z}^d|\Theta)}_{\textbf{marginalized likelihood}} - \underbrace{\log P(\Theta)}_{\textbf{prior}}$$

- Probabilistic grammars are built out of multinomial distributions, so the Dirichlet distributions are natural candidates as priors which encourage smoothness or sparsity [Kurihara and Sato, 2004; Johnson et al. 2007; Tu et al. 2016].

Improvements for MLE: Maximum A Posteriori

$$J(\Theta) = -\log P(\Theta|\mathrm{X}) \propto -\sum_{d=1}^{N} \log \underbrace{\sum_{\mathrm{z}^d} P(\mathrm{x}^d, \mathrm{z}^d|\Theta)}_{\textbf{marginalized likelihood}} - \underbrace{\log P(\Theta)}_{\textbf{prior}}$$

- Probabilistic grammars are built out of multinomial distributions, so the Dirichlet distributions are natural candidates as priors which encourage smoothness or sparsity [Kurihara and Sato, 2004; Johnson et al. 2007; Tu et al. 2016].
- Cohen and Smith (2008; 2009) leverage logistic-normal prior distributions to encourage symbol correlation.

Improvements for MLE: Maximum A Posteriori

$$J(\Theta) = -\log P(\Theta|\mathrm{X}) \propto -\sum_{d=1}^{N} \log \underbrace{\sum_{\mathrm{z}^d} P(\mathrm{x}^d, \mathrm{z}^d|\Theta)}_{\textbf{marginalized likelihood}} - \underbrace{\log P(\Theta)}_{\textbf{prior}}$$

- Probabilistic grammars are built out of multinomial distributions, so the Dirichlet distributions are natural candidates as priors which encourage smoothness or sparsity [Kurihara and Sato, 2004; Johnson et al. 2007; Tu et al. 2016].
- Cohen and Smith (2008; 2009) leverage logistic-normal prior distributions to encourage symbol correlation.
- EM is sometimes not useable in MAP inference, so variational inference and MCMC are often used.

Improvements for MLE: Viterbi Likelihood

$$
\begin{aligned}
J(\Theta) &= -\frac{1}{N} \log \sum_{d=1}^{N} \max_{\mathsf{z}^d} P(\mathsf{x}^d, \mathsf{z}^d) \\
&= -\frac{1}{N} \log \sum_{d=1}^{N} \max_{\mathsf{z}^d} \prod_{\mathsf{r} \in \mathcal{R}(\mathsf{x}^d, \mathsf{z}^d)} P(\mathsf{r})
\end{aligned}
$$

Improvements for MLE: Viterbi Likelihood

$$\begin{aligned} J(\Theta) &= -\frac{1}{N} \log \sum_{d=1}^{N} \max_{\mathrm{z}^d} P(\mathrm{x}^d, \mathrm{z}^d) \\ &= -\frac{1}{N} \log \sum_{d=1}^{N} \max_{\mathrm{z}^d} \prod_{\mathrm{r} \in \mathcal{R}(\mathrm{x}^d, \mathrm{z}^d)} P(\mathrm{r}) \end{aligned}$$

- Optimized with the hard EM algorithm.
- Stronger performance in unsupervised parsing.
- Seen as a special case of entropy regularized model learning.

[Spitkovsky et al. 2010, Tu et al. 2012]

Improvements for MLE: Contrastive Estimation

Contrastive Estimation [Smith and Eisner, 2005a,b]

$$J(\Theta) = -\frac{1}{N} \log \sum_{d=1}^{N} \frac{\sum_{\mathsf{z}^d} s(\mathsf{x}, \mathsf{z})}{\sum_{\mathsf{x}' \in N(\mathsf{x}')} \sum_{\mathsf{z}^d} s(\mathsf{x}', \mathsf{z})}$$

where $s(\mathsf{x}, \mathsf{z})$ is the score of x and z.

Improvements for MLE: Contrastive Estimation

Contrastive Estimation [Smith and Eisner, 2005a,b]

$$J(\Theta) = -\frac{1}{N} \log \sum_{d=1}^{N} \frac{\sum_{\mathrm{z}^d} s(\mathrm{x}, \mathrm{z})}{\sum_{\mathrm{x}' \in N(\mathrm{x}')} \sum_{\mathrm{z}^d} s(\mathrm{x}', \mathrm{z})}$$

where $s(\mathrm{x}, \mathrm{z})$ is the score of x and z.

- The intuition is to assign higher weight to appeared samples and decrease the weight for neighborhood samples.
- Choice of neighborhood: linguistic knowledge.
- Examples: deleting words from x, transposing two words.

Improvements for MLE: Posterior Regularization

Basic idea:

- Uses constraints on posterior distribution to guide parameter learning.
- Knowledge of unlikely parses simplify learning.

Improvements for MLE: Posterior Regularization

Basic idea:

- Uses constraints on posterior distribution to guide parameter learning.
- Knowledge of unlikely parses simplify learning.

$$J'(\Theta, Q(\mathrm{z})) = J(\Theta) + \sum_{d=1}^{N} \mathrm{KL}(q(\mathrm{z}^d) || P(\mathrm{z}^d | \mathrm{x}^d; \Theta)) + \sum_{d=1}^{N} f(q(\mathrm{z}^d))$$

- The new term is only dependent on q.
- Only the E step is affected and modified. M step remains the same.

[Ganchev et al. 2010]

Different Forms of Posterior Constraints

- Entropy constraints [Tu and Honavar, 2012]

$$f(q(\mathrm{z})) = -\sum_{\mathrm{z}} q(\mathrm{z}) \log q(\mathrm{z})$$

- Linguistic constraints [Naseem et al., 2010]

$$f(q(\mathrm{z})) = E_{q(\mathrm{z})} \phi(\mathrm{x}, \mathrm{z})$$

- Sparsity constraints [Gillenwater et al., 2010]

$$f(q(\mathrm{z})) = ||E_{q(\mathrm{z})} \phi(\mathrm{x}, \mathrm{z})||_{\beta}$$

- Bounding recursion depth [Noji et al. 2016].

$\phi(\mathrm{x}, \mathrm{z})$ is a decomposed function, which enables effective learning!

Posterior Regularization #1: Entropy Constraints

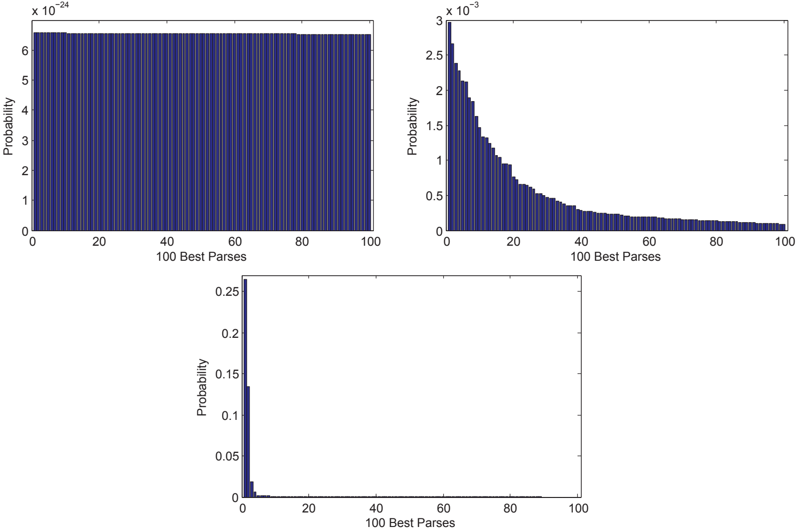
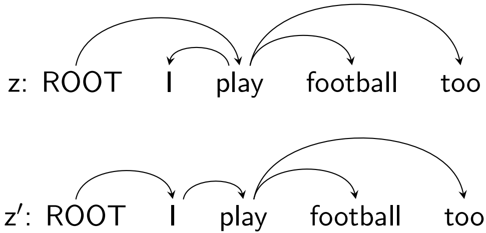
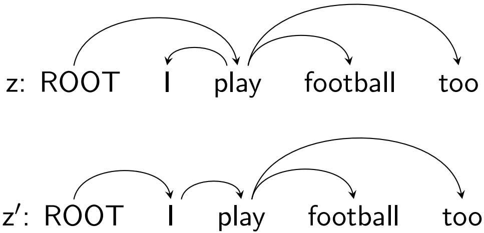


Figure 1: up-left/up-right/down: distribution from random grammar/EM learned model/supervisedly learned grammar, [Tu et al. 2012]

Posterior Regularization #2: Linguistic Constraints



Posterior Regularization #2: Linguistic Constraints



A set of predefined linguistic rules [Naseem et al. 2010], like:

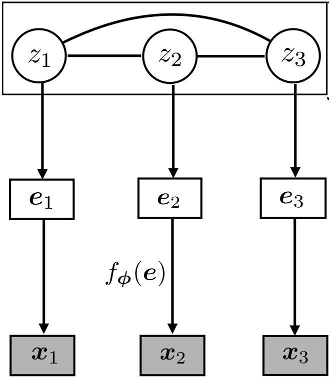
VERB $\rightarrow$ VERB	NOUN $\rightarrow$ NOUN
VERB $\rightarrow$ NOUN	NOUN $\rightarrow$ ADJ
VERB $\rightarrow$ PRON	NOUN $\rightarrow$ DET
VERB $\rightarrow$ ADV	NOUN $\rightarrow$ NUM
VERB $\rightarrow$ ADP	NOUN $\rightarrow$ CONJ
ADJ $\rightarrow$ ADV	ADP $\rightarrow$ NOUN

In this example, $\phi(\mathrm{x}, \mathrm{z}) = 2, \phi(\mathrm{x}, \mathrm{z}') = 1$

Improvements for Avoiding Local Minimum

- Deterministic annealing [Smith and Eisner, 2004]: start with a concave objective and gradually move to the actual non-concave objective.
- Structural annealing [Smith and Eisner, 2006]: gradually decrease structural biases.
- Curriculum learning [Spitkovsky et al. 2010; Tu et al. 2011]: start learning from short sentences; gradually increase training sentence length limit.
- Switching between different objectives [Spitkovsky et al. 2013].
- Treating different learning algorithms and configurations as modules and connecting them to form a network [Spitkovsky et al., 2013].
- Gibbs sampling [Johnson et al., 2007]: may incorporate constraints and biases, e.g., depth-bound [Jin et al., 2018a,b], subtree reducibility [Mareček and Žabokrtský, 2012; Mareček and Straka, 2013].

Flow-based Models: Improve Syntax with Semantics



- Jointly learn discrete syntactic structure and continuous word representations (semantic).
- Latent embedding e, pretrained embeddings x and invertible function f_{ϕ}(e).
- Training with normalizing flow helps induce better parsers.

[He et al. 2018, Jin et al. 2019]

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.
- A typical generative model in unsupervised dependency parsing: DMV

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.
- A typical generative model in unsupervised dependency parsing: DMV
- Three different kinds of representations: table-based, feature-based and neural based methods.

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.
- A typical generative model in unsupervised dependency parsing: DMV
- Three different kinds of representations: table-based, feature-based and neural based methods.
- EM algorithm is one of standard approach to learn the grammar parameter, which MLE objectives.

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.
- A typical generative model in unsupervised dependency parsing: DMV
- Three different kinds of representations: table-based, feature-based and neural based methods.
- EM algorithm is one of standard approach to learn the grammar parameter, which MLE objectives.
- Recently, directly gradient descent is more popular.

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.
- A typical generative model in unsupervised dependency parsing: DMV
- Three different kinds of representations: table-based, feature-based and neural based methods.
- EM algorithm is one of standard approach to learn the grammar parameter, which MLE objectives.
- Recently, directly gradient descent is more popular.
- Many improvements for MLE: viterbi likelihood, MAP estimation, contrastive estimation.

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.
- A typical generative model in unsupervised dependency parsing: DMV
- Three different kinds of representations: table-based, feature-based and neural based methods.
- EM algorithm is one of standard approach to learn the grammar parameter, which MLE objectives.
- Recently, directly gradient descent is more popular.
- Many improvements for MLE: viterbi likelihood, MAP estimation, contrastive estimation.
- Posterior regularization is a useful approach to encode knowledge.

Summary on Parameter Learning of Generative Approaches

- Structure learning is hard. Parameter learning is much easier and draws more attention.
- A typical generative model in unsupervised dependency parsing: DMV
- Three different kinds of representations: table-based, feature-based and neural based methods.
- EM algorithm is one of standard approach to learn the grammar parameter, which MLE objectives.
- Recently, directly gradient descent is more popular.
- Many improvements for MLE: viterbi likelihood, MAP estimation, contrastive estimation.
- Posterior regularization is a useful approach to encode knowledge.
- Many technologies can improve avoiding local minimum.