Lecture 2: Statistical Inference

Jelle Zuidema

Institute for Logic, Language and Computation
University of Amsterdam, The Netherlands

Unsupervised Language Learning 2014
MSc Artificial Intelligence / MSc Logic
University of Amsterdam

A very brief tour of statistical learning

Bayes' Rule

$$P(G|D) = \frac{P(D|G)\,P(G)}{P(D)}$$

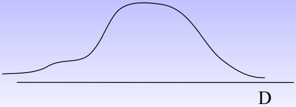
Bayes' Rule

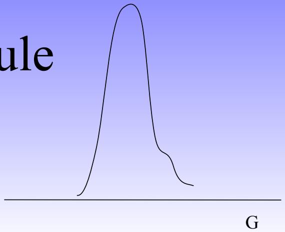
likelihood *prior*

$$\underset{\textit{posterior}}{P(G|D)} = \frac{P(D|G)\,P(G)}{P(D)}$$

probability of data

Bayes' Rule

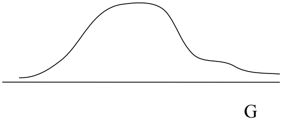




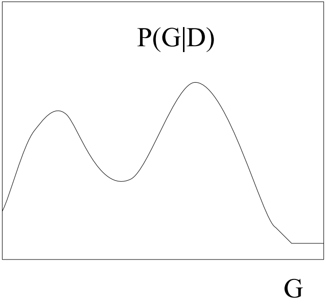
likelihood *prior*

$$P(G|D) = \frac{P(D|G)\,P(G)}{P(D)}$$

posterior *probability of data*

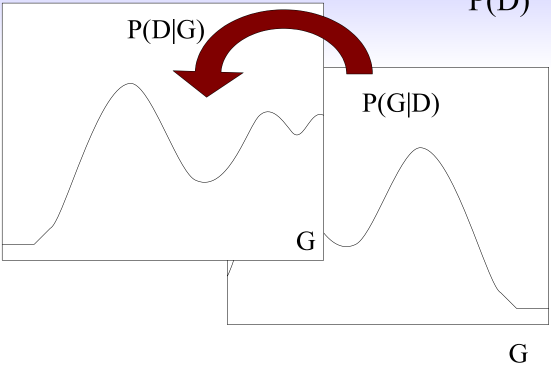


Statistical inference



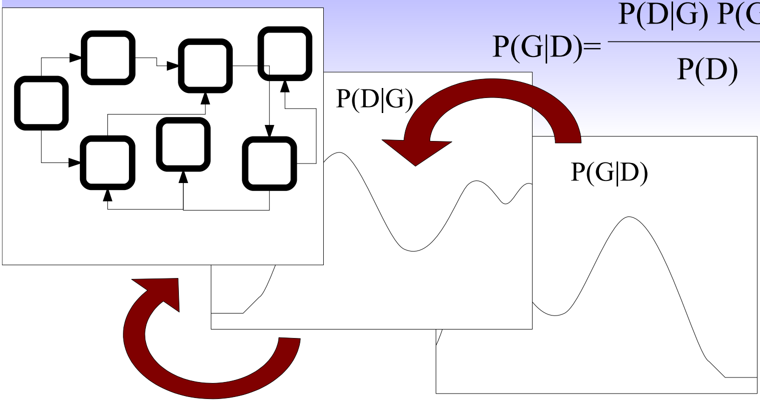
Statistical inference

$$P(G|D) = \frac{P(D|G)\,P(G)}{P(D)}$$

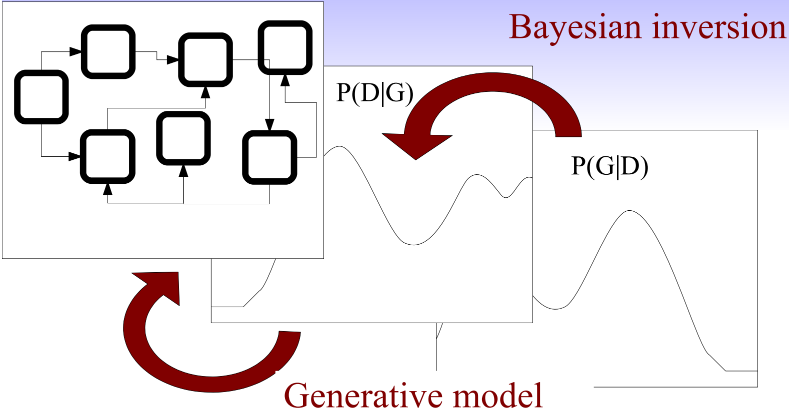


Statistical inference

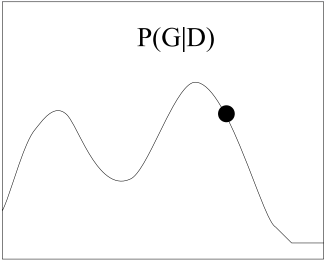
$$P(G|D) = \frac{P(D|G)\,P(G)}{P(D)}$$



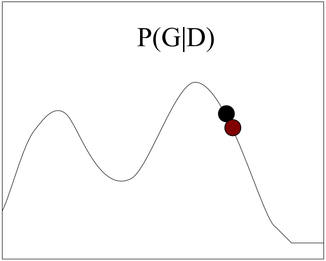
Statistical inference



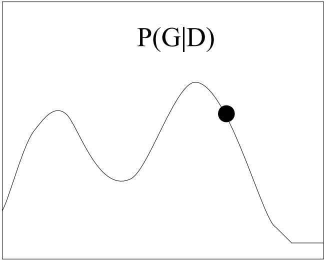
Stochastic hillclimbing



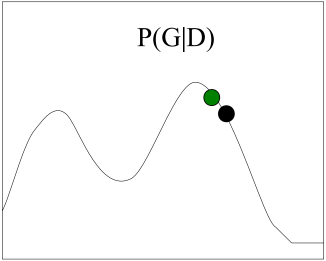
Stochastic hillclimbing



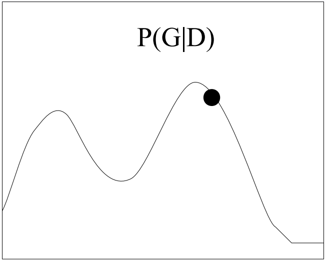
Stochastic hillclimbing



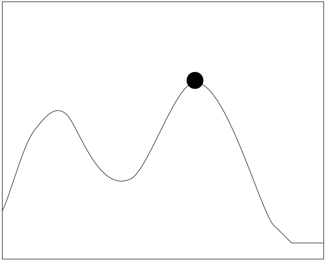
Stochastic hillclimbing



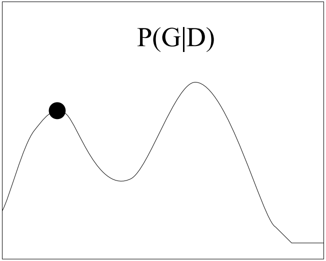
Stochastic hillclimbing



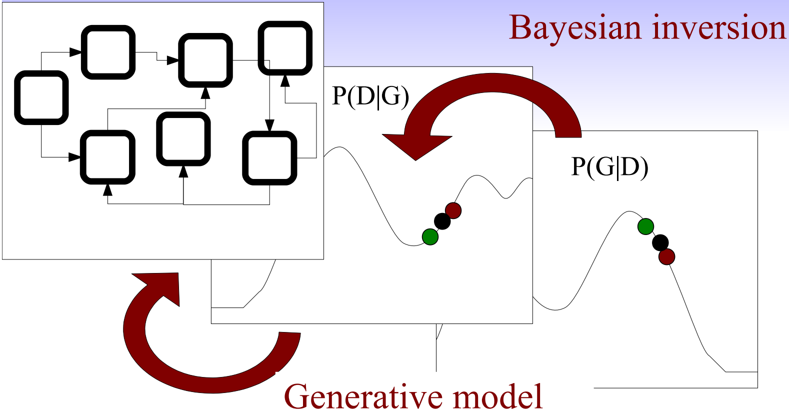
Stochastic hillclimbing



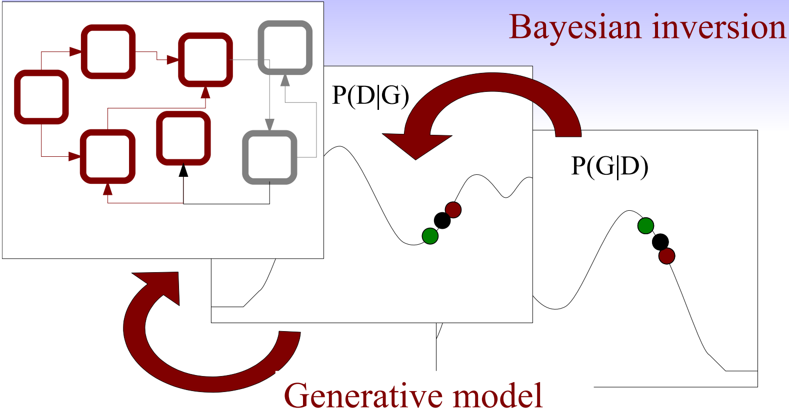
Local optimum



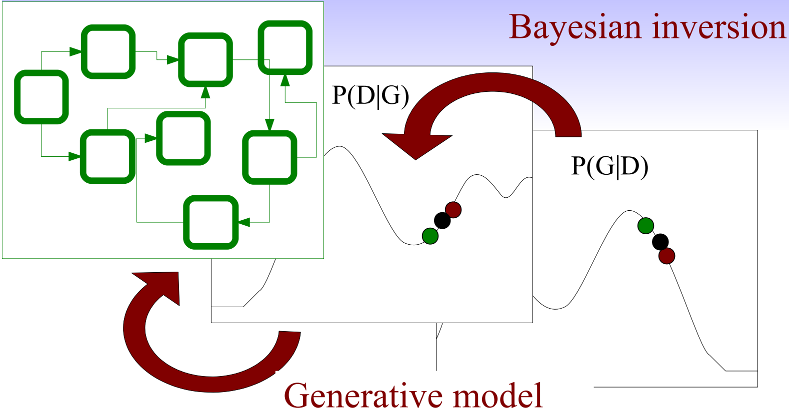
Statistical inference



Statistical inference



Statistical inference



Maximum Likelihood (ML) Hypothesis:

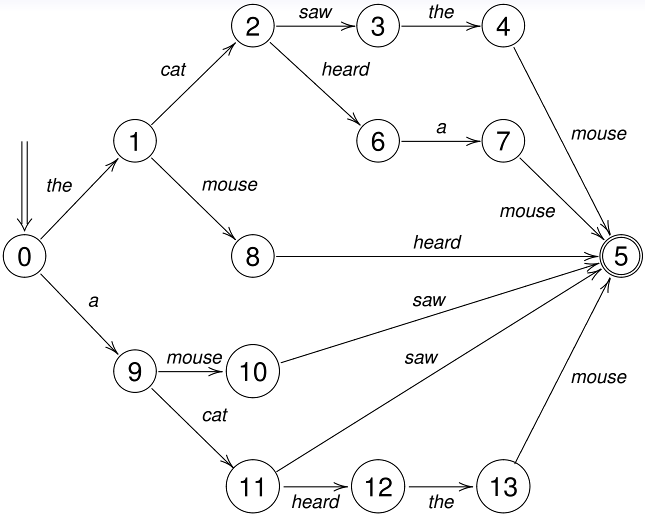
$$\begin{aligned} \text{argmax}_h \quad & P(\text{hypothesis}|\text{data}) \approx \\ \text{argmax}_h \quad & P(\text{data}|\text{hypothesis}) \end{aligned}$$

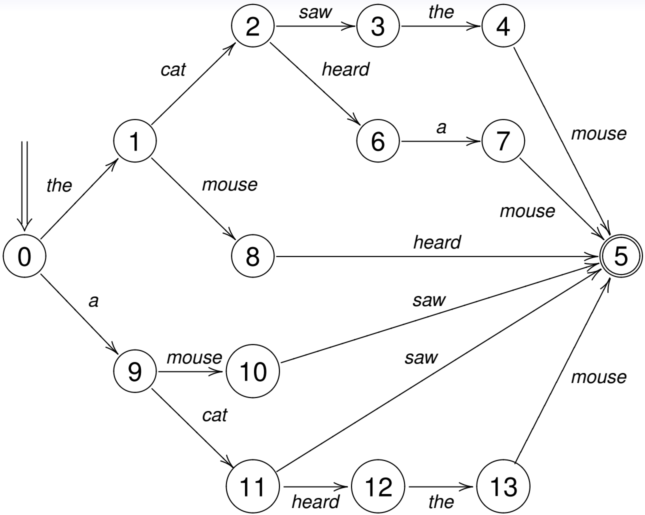
Bayesian Maximum A Posteriori (MAP) Hypothesis:

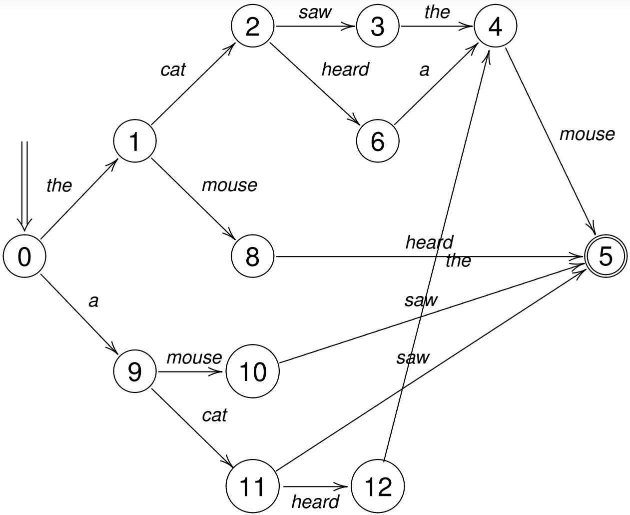
$$\begin{aligned} \text{argmax}_h \quad & P(\text{hypothesis}|\text{data}) = \\ \text{argmax}_h \quad & \frac{P(\text{data}|\text{hypothesis})P(\text{hypothesis})}{P(\text{data}} \end{aligned}$$

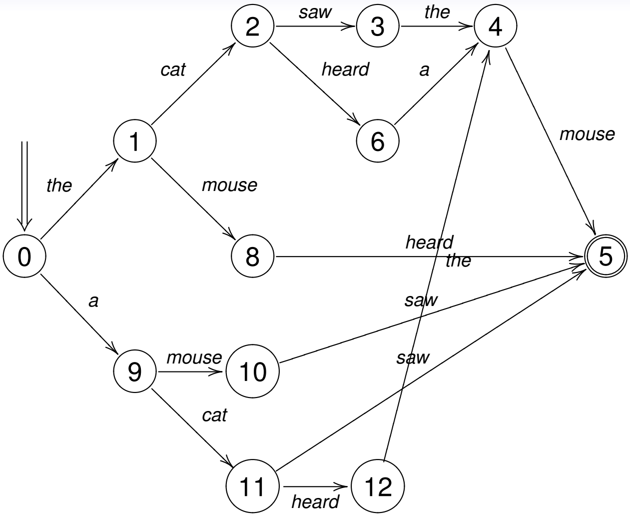
-
    - The cat saw the mouse.
    - The cat heard a mouse.
    - The mouse heard.
    - A mouse saw.
    - A cat saw.
    - A cat heard the mouse.

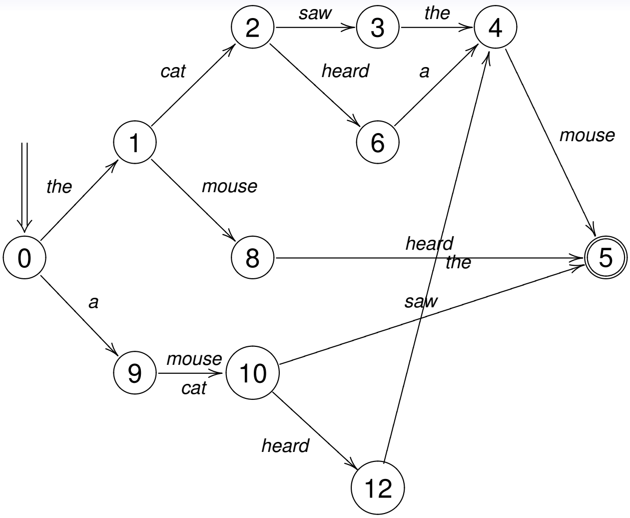
(Langley & Stromsten, 2000)

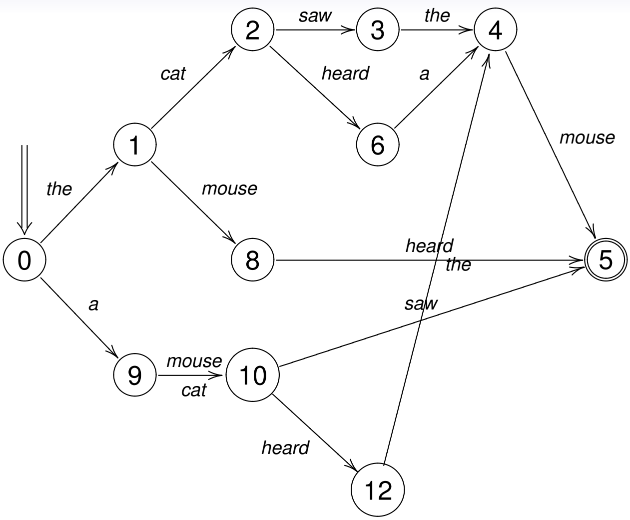

0
1
2
3
4
5
6
7
8
9
10
11
12
13
the
a
cat
mouse
saw
the
heard
a
mouse
mouse
heard
saw
saw
mouse
mouse
cat
heard
the

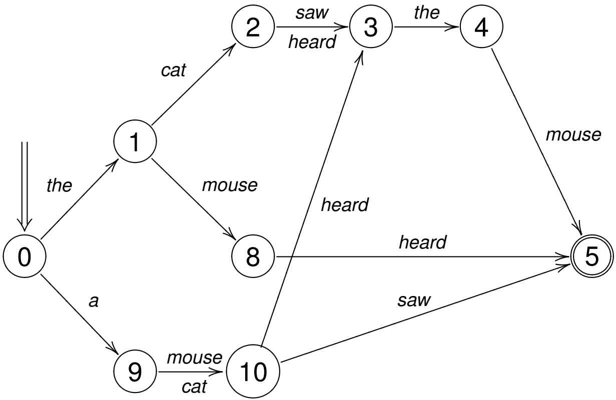

0
1
2
3
4
5
6
7
8
9
10
11
12
13
the
a
cat
mouse
saw
heard
the
a
mouse
mouse
heard
saw
saw
mouse
cat
heard
the
mouse

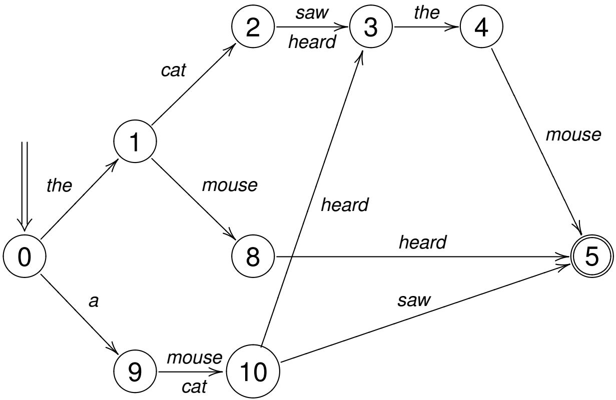

0
1
2
3
4
5
6
8
9
10
11
12
the
a
cat
mouse
saw
the
heard
a
mouse
heard
the
saw
mouse
cat
saw
heard

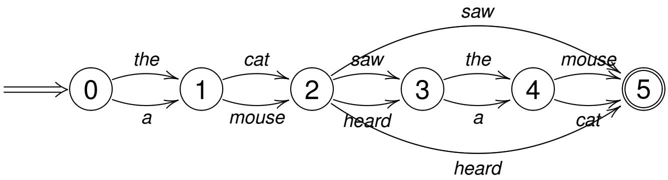

0
1
2
3
4
5
6
8
9
10
11
12
the
a
cat
mouse
saw
the
heard
a
mouse
heard
the
saw
mouse
cat
saw
heard

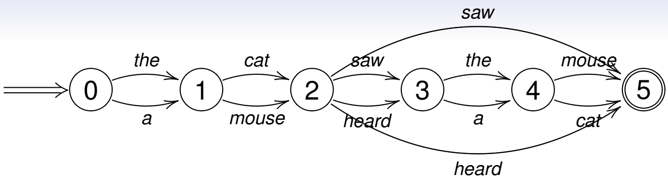

0
1
2
3
4
5
6
8
9
10
12
the
a
cat
mouse
saw
the
heard
a
mouse
heard
the
saw
mouse
cat
heard


0
1
2
3
4
5
6
8
9
10
12
the
a
cat
mouse
saw
heard
the
a
mouse
heard
the
mouse
cat
saw
heard


0
1
2
3
4
5
8
9
10
the
a
cat
mouse
saw
heard
the
mouse
heard
heard
saw
mouse
cat

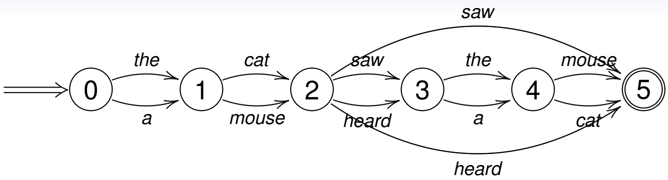

2
3
4
saw
heard
the
cat
1
mouse
heard
mouse
the
0
8
heard
5
a
saw
9
mouse
cat
10


saw
the
cat
saw
the
mouse
0
1
2
3
4
5
a
mouse
heard
a
cat
heard



0→the 1
0→a 1
1→cat 2
1→mouse 2
2→saw
2→heard
2→saw 3
2→heard 3
3→the 4
3→a 4
4→cat
4→mouse

- Grammar Description Length: 32 symbols
- *Prior* Probability Grammar: 2^{-32}

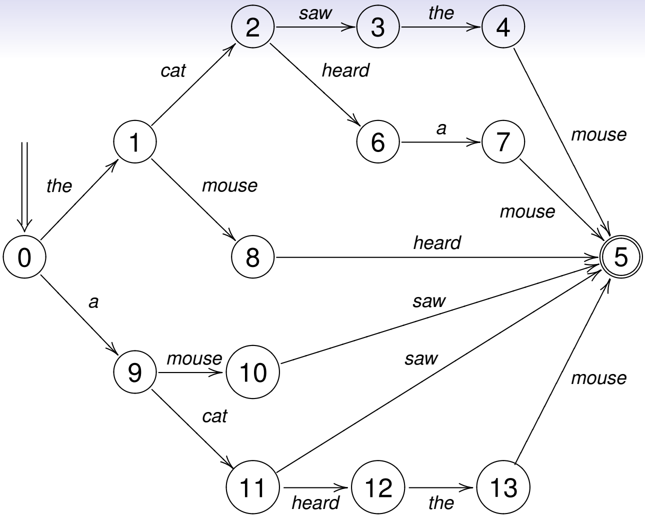


-
  - The cat saw the mouse.
    $1/2 \times 1/2 \times 1/4 \times 1/2 \times 1/2 = 1/64$
  - The cat heard a mouse. $1/64$
  - The mouse heard. $1/2 \times 1/2 \times 1/4 = 1/16$
  - A mouse saw. $1/16$
  - A cat saw. $1/16$
  - A cat heard the mouse. $1/64$

.... $\times$

Data Likelihood = 2^{-30}

(Langley & Stromsten, 2000)



- Data Description Length: 11
- Data Likelihood: 2^{-11}
- Grammar Description Length: 54
- Grammar Prior: 2^{-54}

Context-free Grammar

S $\rightarrow$NP VP	VP$\rightarrow$V
NP$\rightarrow$Art N	V $\rightarrow$saw
Art$\rightarrow$the	V $\rightarrow$heard
Art$\rightarrow$a	VP$\rightarrow$V NP
N $\rightarrow$cat	
N $\rightarrow$mouse	

- Data Description Length: 30
- Data Likelihood: 2^{-30}
- Grammar Description Length: 21
- Grammar Prior: 2^{-21}

Outline

Statistical Inference

Bayesian Model Merging

Implementation

Unsupervised induction

Unsupervised labeling

Slides derived from slides of Gideon Borensztajn.

PCFG induction by Bayesian Model Merging

- Goal: unsupervised induction of PCFG from flat text.
- Earlier methods (e.g., 1990) use parameter search (and EM).
- In Bayesian Model Merging (BMM) (Stolcke, 1994) it is assumed that neither parameters, nor structure are known.
- Stochastic hill-climbing search through space of possible grammars.
- Maximizing posterior probability (rather than likelihood of data).

PCFG induction by Bayesian Model Merging

- Goal: unsupervised induction of PCFG from flat text.
- Earlier methods (e.g., 1990) use parameter search (and EM).
- In Bayesian Model Merging (BMM) (Stolcke, 1994) it is assumed that neither parameters, nor structure are known.
- Stochastic hill-climbing search through space of possible grammars.
- Maximizing posterior probability (rather than likelihood of data).

PCFG induction by Bayesian Model Merging

- Goal: unsupervised induction of PCFG from flat text.
- Earlier methods (e.g., 1990) use parameter search (and EM).
- In Bayesian Model Merging (BMM) (Stolcke, 1994) it is assumed that neither parameters, nor structure are known.
- Stochastic hill-climbing search through space of possible grammars.
- Maximizing posterior probability (rather than likelihood of data).

PCFG induction by Bayesian Model Merging

- Goal: unsupervised induction of PCFG from flat text.
- Earlier methods (e.g., 1990) use parameter search (and EM).
- In Bayesian Model Merging (BMM) (Stolcke, 1994) it is assumed that neither parameters, nor structure are known.
- Stochastic hill-climbing search through space of possible grammars.
- Maximizing posterior probability (rather than likelihood of data).

PCFG induction by Bayesian Model Merging

- Goal: unsupervised induction of PCFG from flat text.
- Earlier methods (e.g., 1990) use parameter search (and EM).
- In Bayesian Model Merging (BMM) (Stolcke, 1994) it is assumed that neither parameters, nor structure are known.
- Stochastic hill-climbing search through space of possible grammars.
- Maximizing posterior probability (rather than likelihood of data).

Merging and chunking in BMM

In BMM, two learning operators (successor functions) involved in hill-climbing search: merging and chunking operators

- The *merging* operator creates generalizations by forming disjunctive groups (categories) of patterns that occur in the same contexts. It replaces two existing non-terminals X_1 and X_2 with a single new non-terminal Y.
- The *chunking* operator concatenates or chunks repeating patterns. It takes a sequence of two nonterminals X_1 and X_2 and creates a new nonterminal Y that expands to $X_1 X_2$.

Merging and chunking in BMM

In BMM, two learning operators (successor functions) involved in hill-climbing search: merging and chunking operators

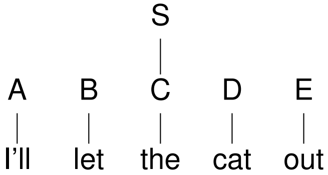
- The *merging* operator creates generalizations by forming disjunctive groups (categories) of patterns that occur in the same contexts. It replaces two existing non-terminals X_1 and X_2 with a single new non-terminal Y.
- The *chunking* operator concatenates or chunks repeating patterns. It takes a sequence of two nonterminals X_1 and X_2 and creates a new nonterminal Y that expands to $X_1 X_2$.

Steps in BMM

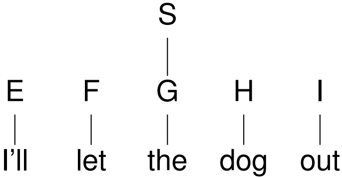
- Initialization of grammar by incorporating sentences as flat rules.
- Iterated merging and chunking in alternating phases.

The merge/chunk that scores best on an evaluation function is selected.

I'll let the cat out

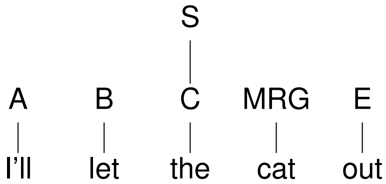


I'll let the dog out

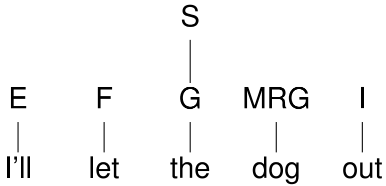


Initialization of grammar by incorporating sentences as flat rules

I'll let the cat out

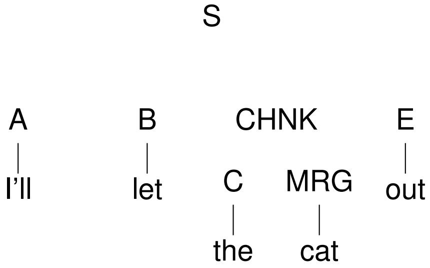


I'll let the dog out

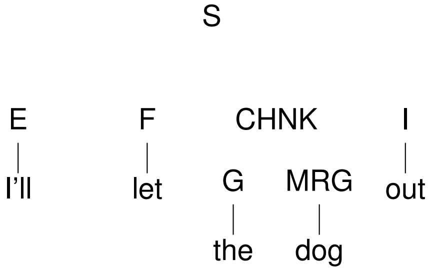


merge(D,H)

I'll let the cat out



I'll let the dog out



chunk(C,MRG), *chunk*(G,MRG)

Bayesian learning and MAP hypothesis

Maximum a Posteriori (MAP) hypothesis, M_{MAP} is the hypothesis that maximizes the posterior probability (with given data). With Bayes Law:

$$
\begin{aligned}
M_{MAP} \equiv argmax_M P(M|X) &= argmax_M \frac{P(X|M) \cdot P(M)}{P(X)} = \\
&= argmax_M P(X|M) \cdot P(M)
\end{aligned}
$$

$P(X|M)$ is likelihood
$P(M)$ is the prior probability, or ‘prior’.

Bayesian learning and MAP hypothesis

Maximum a Posteriori (MAP) hypothesis, M_{MAP} is the hypothesis that maximizes the posterior probability (with given data). With Bayes Law:

$$\begin{aligned} M_{MAP} \equiv argmax_M P(M|X) &= argmax_M \frac{P(X|M) \cdot P(M)}{P(X)} = \\ &= argmax_M P(X|M) \cdot P(M) \end{aligned}$$

$P(X|M)$ is likelihood
$P(M)$ is the prior probability, or 'prior'.

Bayesian learning and MAP hypothesis - ctd

- Prior is constructed such that it expresses the designer's a priori preferences for the model: this is a probabilistic form of bias (e.g. Occam's Razor).
- The maximization of $P(X|M) \cdot P(M)$ is equivalent to minimizing

$$-logP(M) - logP(X|M) \approx GDL + DDL = DL \tag{1}$$

- In information theory: estimation by *Minimum Description Length* (MDL).
  - Grammar Description Length (GDL): the length needed to encode the model
  - Data Description Length (DDL): the bits needed to describe the data given the model.

Bayesian learning and MAP hypothesis - ctd

- Prior is constructed such that it expresses the designer's a priori preferences for the model: this is a probabilistic form of bias (e.g. Occam's Razor).
- The maximization of $P(X|M) \cdot P(M)$ is equivalent to minimizing

$$-logP(M) - logP(X|M) \approx GDL + DDL = DL \tag{1}$$

- In information theory: estimation by *Minimum Description Length* (MDL).
  - Grammar Description Length (*GDL*): the length needed to encode the model
  - Data Description Length (*DDL*): the bits needed to describe the data given the model.

Bayesian learning and MAP hypothesis - ctd

- Prior is constructed such that it expresses the designer's a priori preferences for the model: this is a probabilistic form of bias (e.g. Occam's Razor).
- The maximization of $P(X|M) \cdot P(M)$ is equivalent to minimizing

$$-logP(M) - logP(X|M) \approx GDL + DDL = DL \tag{1}$$

- In information theory: estimation by *Minimum Description Length* (MDL).
  - Grammar Description Length (GDL): the length needed to encode the model
  - Data Description Length (DDL): the bits needed to describe the data given the model.

Bayesian learning and MAP hypothesis - ctd

- Prior is constructed such that it expresses the designer's a priori preferences for the model: this is a probabilistic form of bias (e.g. Occam's Razor).
- The maximization of $P(X|M) \cdot P(M)$ is equivalent to minimizing

$$-logP(M) - logP(X|M) \approx GDL + DDL = DL \qquad (1)$$

- In information theory: estimation by *Minimum Description Length* (MDL).
  - Grammar Description Length (*GDL*): the length needed to encode the model
  - Data Description Length (*DDL*): the bits needed to describe the data given the model.

Bayesian learning and MAP hypothesis - ctd

- Prior is constructed such that it expresses the designer's a priori preferences for the model: this is a probabilistic form of bias (e.g. Occam's Razor).
- The maximization of $P(X|M) \cdot P(M)$ is equivalent to minimizing

$$-logP(M) - logP(X|M) \approx GDL + DDL = DL \tag{1}$$

- In information theory: estimation by *Minimum Description Length* (MDL).
  - Grammar Description Length (*GDL*): the length needed to encode the model
  - Data Description Length (*DDL*): the bits needed to describe the data given the model.

Priors

The prior can be decomposed into a structure prior and a parameter prior:

$$P(M) = P(M_S) \cdot P(\Theta_M | M_S) \tag{2}$$

- The *structure prior* $P(M_S)$ is the inverse exponential of the code length of the model: $logP(M_S) = -DL(M_S)$. I.e., the minimal number of bits required to transmit the grammar.
- In the simplest case, parameter prior ignored (i.e. uniform).

Priors

The prior can be decomposed into a structure prior and a parameter prior:

$$P(M) = P(M_S) \cdot P(\Theta_M | M_S) \tag{2}$$

- The *structure prior* $P(M_S)$ is the inverse exponential of the code length of the model: $logP(M_S) = -DL(M_S)$. I.e., the minimal number of bits required to transmit the grammar.
- In the simplest case, parameter prior ignored (i.e. uniform).

e-Grids

- Because of computational complexity not feasible to apply the Stolcke algorithm on real natural language corpora
- e-Grids (Petasis et al., 2004): reduces the complexity of the computations by forecasting description length change from merge or chunk operation → no need to construct a grammar for every search operator.
- complexity of chunk reduced from $O(N^2)$ to $O(N)$
  complexity of merge reduced from $O(N^3)$ to $O(N^2)$
  (where N is the number of non-terminals)

e-Grids

- Because of computational complexity not feasible to apply the Stolcke algorithm on real natural language corpora
- e-Grids (Petasis et al., 2004): reduces the complexity of the computations by forecasting description length change from merge or chunk operation → no need to construct a grammar for every search operator.
- complexity of chunk reduced from $O(N^2)$ to $O(N)$
  complexity of merge reduced from $O(N^3)$ to $O(N^2)$
  (where N is the number of non-terminals)

e-Grids

- Because of computational complexity not feasible to apply the Stolcke algorithm on real natural language corpora
- e-Grids (Petasis et al., 2004): reduces the complexity of the computations by forecasting description length change from merge or chunk operation $\rightarrow$ no need to construct a grammar for every search operator.
- complexity of chunk reduced from $O(N^2)$ to $O(N)$
  complexity of merge reduced from $O(N^3)$ to $O(N^2)$
  (where N is the number of non-terminals)

Evaluation

OVIS: a treebank containing 10040 annotated Dutch sentences from a public transport information system.
WSJ10-POSTAGS: 7422 POSTAG sequences of length $<=$ 10 extracted from Wall Street Journal (WSJ) (Klein & Manning, 2002).

	OVIS	**WSJ10**
Sentences	10040	7422
Sentences $>$ 1 **word**	6892	7263
Words	31697	52089
Av. sentence length	4.60	7.17
Vocabulary	946	35
Av. token frequency	33.5	1488

Experimental results

Evalutation using a version of PARSEVAL (Klein & Manning, 2005). (Poisson, $\mu = 2.5$)

OVIS	**UP**	**UR**	**F**
R-B	68.91	66.30	67.58
BMM	71.70	66.56	69.03

Experimental results (ctd)

WSJ-10	UP	UR	F
CCM (Klein & Manning, 2002)	64.2	81.6	71.9
DMV+CCM (Klein & Manning, 2004)	69.3	88.0	77.6
U-DOP (Bod, 2006)	70.8	88.2	78.5
R-B	70.0	55.12	61.68
BMM	**57.57**	**42.65**	**49.00**

- For WSJ-10 the scores are disappointing, compared to previous work on unsupervised grammar induction, and even to right branching (R-B).
- influence of parameter settings is minor.
- Follow-up experiments to explain disappointing results.

Experimental results (ctd)

WSJ-10	UP	UR	F
CCM (Klein & Manning, 2002)	64.2	81.6	71.9
DMV+CCM (Klein & Manning, 2004)	69.3	88.0	77.6
U-DOP (Bod, 2006)	70.8	88.2	78.5
R-B	70.0	55.12	61.68
BMM	**57.57**	**42.65**	**49.00**

- For WSJ-10 the scores are disappointing, compared to previous work on unsupervised grammar induction, and even to right branching (R-B).
- influence of parameter settings is minor.
- Follow-up experiments to explain disappointing results.

Experimental results (ctd)

WSJ-10	UP	UR	F
CCM (Klein & Manning, 2002)	64.2	81.6	71.9
DMV+CCM (Klein & Manning, 2004)	69.3	88.0	77.6
U-DOP (Bod, 2006)	70.8	88.2	78.5
R-B	70.0	55.12	61.68
BMM	**57.57**	**42.65**	**49.00**

- For WSJ-10 the scores are disappointing, compared to previous work on unsupervised grammar induction, and even to right branching (R-B).
- influence of parameter settings is minor.
- Follow-up experiments to explain disappointing results.

Experimental results (ctd)

WSJ-10	**UP**	**UR**	**F**
CCM (Klein & Manning, 2002)	64.2	81.6	71.9
DMV+CCM (Klein & Manning, 2004)	69.3	88.0	77.6
U-DOP (Bod, 2006)	70.8	88.2	78.5
R-B	70.0	55.12	61.68
BMM	**57.57**	**42.65**	**49.00**

- For WSJ-10 the scores are disappointing, compared to previous work on unsupervised grammar induction, and even to right branching (R-B).
- influence of parameter settings is minor.
- Follow-up experiments to explain disappointing results.

Is failure due to objective function or search?

- Is the treebank grammar indeed a minimum of the objective function?
- Test: BMM algorithm was initialized with the treebank grammar.
- Priors: Poisson ($\mu = 3.0$) and Dirichlet. Similar results for alternative priors.

	DL	GDL	DDL	UP	UR	F
Treebank Initial	292515	66067	226448	90.10	88.64	89.36
Final	275807	40557	235250	64.31	74.78	69.15

Conclusion: BMM can still optimize the description length when initialized with the treebank grammar → treebank grammar is not even a local minimum of the objective function.

Is failure due to objective function or search?

- Is the treebank grammar indeed a minimum of the objective function?
- Test: BMM algorithm was initialized with the treebank grammar.
- Priors: Poisson ($\mu = 3.0$) and Dirichlet. Similar results for alternative priors.

	DL	GDL	DDL	UP	UR	F
Treebank Initial	292515	66067	226448	90.10	88.64	89.36
Final	275807	40557	235250	64.31	74.78	69.15

Conclusion: BMM can still optimize the description length when initialized with the treebank grammar → treebank grammar is not even a local minimum of the objective function.

Is failure due to objective function or search?

- Is the treebank grammar indeed a minimum of the objective function?
- Test: BMM algorithm was initialized with the treebank grammar.
- Priors: Poisson ($\mu = 3.0$) and Dirichlet. Similar results for alternative priors.

	DL	GDL	DDL	UP	UR	F
Treebank Initial	292515	66067	226448	90.10	88.64	89.36
Final	275807	40557	235250	64.31	74.78	69.15

Conclusion: BMM can still optimize the description length when initialized with the treebank grammar $\rightarrow$ treebank grammar is not even a local minimum of the objective function.

Is failure due to objective function or search?

- Is the treebank grammar indeed a minimum of the objective function?
- Test: BMM algorithm was initialized with the treebank grammar.
- Priors: Poisson ($\mu = 3.0$) and Dirichlet. Similar results for alternative priors.

	DL	**GDL**	**DDL**	**UP**	**UR**	**F**
Treebank Initial	292515	66067	226448	90.10	88.64	89.36
Final	275807	40557	235250	64.31	74.78	69.15

Conclusion: BMM can still optimize the description length when initialized with the treebank grammar $\rightarrow$ treebank grammar is not even a local minimum of the objective function.

Is failure due to objective function or search?

- Is the treebank grammar indeed a minimum of the objective function?
- Test: BMM algorithm was initialized with the treebank grammar.
- Priors: Poisson ($\mu = 3.0$) and Dirichlet. Similar results for alternative priors.

	DL	GDL	DDL	UP	UR	F
Treebank Initial	292515	66067	226448	90.10	88.64	89.36
Final	275807	40557	235250	64.31	74.78	69.15

Conclusion: BMM can still optimize the description length when initialized with the treebank grammar $\rightarrow$ treebank grammar is not even a local minimum of the objective function.

Temporal dynamics of BMM

Merges and chunks of OVIS and WSJ are initially linguistically relevant, but soon after fail to be so. Is this behavior reflected in the PARSEVAL scores?

After 10 out of 60 chunks the F-score reaches maximum, At the same time, DL continues decreasing monotonically → objective function gives no good stopping criterion!

Temporal dynamics of BMM

Merges and chunks of OVIS and WSJ are initially linguistically relevant, but soon after fail to be so. Is this behavior reflected in the PARSEVAL scores?

After 10 out of 60 chunks the F-score reaches maximum, At the same time, DL continues decreasing monotonically → objective function gives no good stopping criterion!

Temporal dynamics of BMM

Merges and chunks of OVIS and WSJ are initially linguistically relevant, but soon after fail to be so. Is this behavior reflected in the PARSEVAL scores?
After 10 out of 60 chunks the F-score reaches maximum, At the same time, DL continues decreasing monotonically $\rightarrow$ objective function gives no good stopping criterion!

Conclusions on BMM

- Contrary to received wisdom (Klein, 2005, Clark, 2001) BMM can be evaluated on large corpora.
- Although good results on artificial grammars, BMM (still) disappointing on real languages.
- Probably not a search problem, but objective function doesn't fit natural languages
- Too much noise: merging errors are carried over to the chunking phase and vice versa, causing a snow ball effect. Better for single algorithm not to deal at the same time both with bracketing and with labeling
  $\longrightarrow$ new approach: unsupervised labeling with BMM.

Conclusions on BMM

- Contrary to received wisdom (Klein, 2005, Clark, 2001) BMM can be evaluated on large corpora.
- Although good results on artificial grammars, BMM (still) disappointing on real languages.
- Probably not a search problem, but objective function doesn't fit natural languages
- Too much noise: merging errors are carried over to the chunking phase and vice versa, causing a snow ball effect. Better for single algorithm not to deal at the same time both with bracketing and with labeling
  $\longrightarrow$ new approach: unsupervised labeling with BMM.

Conclusions on BMM

- Contrary to received wisdom (Klein, 2005, Clark, 2001) BMM can be evaluated on large corpora.
- Although good results on artificial grammars, BMM (still) disappointing on real languages.
- Probably not a search problem, but objective function doesn't fit natural languages
- Too much noise: merging errors are carried over to the chunking phase and vice versa, causing a snow ball effect. Better for single algorithm not to deal at the same time both with bracketing and with labeling
  $\rightarrow$ new approach: unsupervised labeling with BMM.

Conclusions on BMM

- Contrary to received wisdom (Klein, 2005, Clark, 2001) BMM can be evaluated on large corpora.
- Although good results on artificial grammars, BMM (still) disappointing on real languages.
- Probably not a search problem, but objective function doesn't fit natural languages
- Too much noise: merging errors are carried over to the chunking phase and vice versa, causing a snow ball effect. Better for single algorithm not to deal at the same time both with bracketing and with labeling
  $\rightarrow$ new approach: unsupervised labeling with BMM.

Unsupervised Labeling

- Semi-supervised induction in two stages:
  - use bracketed sentences from the treebank, or specialized unsupervised bracketing algorithm.
  - give the brackets as input to the BMM algorithm adapted for unsupervised label induction.
- Treebank bracketings of WSJ10 were used as input; for pilot experiments 5000 (declarative) POSTAG sequences were selected having S non-terminal as their root.
- We use the BMM algorithm as before, but without chunking. The initial grammar consists of the rules read off from the target bracketings, but with a unique label for every non-terminal. Non-terminals with the same descendants are given equal names.

Unsupervised Labeling

- Semi-supervised induction in two stages:
  - use bracketed sentences from the treebank, or specialized unsupervised bracketing algorithm.
  - give the brackets as input to the BMM algorithm adapted for unsupervised label induction.
- Treebank bracketings of WSJ10 were used as input; for pilot experiments 5000 (declarative) POSTAG sequences were selected having S non-terminal as their root.
- We use the BMM algorithm as before, but without chunking. The initial grammar consists of the rules read off from the target bracketings, but with a unique label for every non-terminal. Non-terminals with the same descendants are given equal names.

Unsupervised Labeling

- Semi-supervised induction in two stages:
  - use bracketed sentences from the treebank, or specialized unsupervised bracketing algorithm.
  - give the brackets as input to the BMM algorithm adapted for unsupervised label induction.
- Treebank bracketings of WSJ10 were used as input; for pilot experiments 5000 (declarative) POSTAG sequences were selected having S non-terminal as their root.
- We use the BMM algorithm as before, but without chunking. The initial grammar consists of the rules read off from the target bracketings, but with a unique label for every non-terminal. Non-terminals with the same descendants are given equal names.

Unsupervised Labeling

- Semi-supervised induction in two stages:
  - use bracketed sentences from the treebank, or specialized unsupervised bracketing algorithm.
  - give the brackets as input to the BMM algorithm adapted for unsupervised label induction.
- Treebank bracketings of WSJ10 were used as input; for pilot experiments 5000 (declarative) POSTAG sequences were selected having S non-terminal as their root.
- We use the BMM algorithm as before, but without chunking. The initial grammar consists of the rules read off from the target bracketings, but with a unique label for every non-terminal. Non-terminals with the same descendants are given equal names.

Unsupervised Labeling

- Semi-supervised induction in two stages:
  - use bracketed sentences from the treebank, or specialized unsupervised bracketing algorithm.
  - give the brackets as input to the BMM algorithm adapted for unsupervised label induction.
- Treebank bracketings of WSJ10 were used as input; for pilot experiments 5000 (declarative) POSTAG sequences were selected having S non-terminal as their root.
- We use the BMM algorithm as before, but without chunking. The initial grammar consists of the rules read off from the target bracketings, but with a unique label for every non-terminal. Non-terminals with the same descendants are given equal names.

Results on WSJ

Evaluation: greedy remapping of the experimental labels onto the treebank labels (Haghighi and Klein, 2006)
since # experimental labels $>>$ # treebank labels we also compute scores remapping the other way round.

	LP	LR	F
BMM (uniform)	87.3	31.9	46.7
BMM (3000)	76.3	84.3	80.1
BMM (5000)	75.3	82.3	78.7
Haghighi & Klein '06	64.8	78.7	71.1

- BMM performs better than state-of-the-art labeling algorithms.
- High F-scores on categories TOP, NP, and VP (77% of all brackets) are responsible for good result. F-scores on PP, ADVP, ADJP considerably less.
- Adaptation of e-Grids for PCFG ('non-uniform distribution') significantly improves the scores.

Results on WSJ

Evaluation: greedy remapping of the experimental labels onto the treebank labels (Haghighi and Klein, 2006)
since # experimental labels $>>$ # treebank labels we also compute scores remapping the other way round.

	LP	LR	F
BMM (uniform)	87.3	31.9	46.7
BMM (3000)	76.3	84.3	80.1
BMM (5000)	75.3	82.3	78.7
Haghighi & Klein '06	64.8	78.7	71.1

- BMM performs better than state-of-the-art labeling algorithms.
- High F-scores on categories TOP, NP, and VP (77% of all brackets) are responsible for good result. F-scores on PP, ADVP, ADJP considerably less.
- Adaptation of e-Grids for PCFG ('non-uniform distribution') significantly improves the scores.

Results on WSJ

Evaluation: greedy remapping of the experimental labels onto the treebank labels (Haghighi and Klein, 2006)
since # experimental labels $>>$ # treebank labels we also compute scores remapping the other way round.

	LP	**LR**	**F**
BMM (uniform)	87.3	31.9	46.7
BMM (3000)	76.3	84.3	80.1
BMM (5000)	75.3	82.3	78.7
Haghighi & Klein '06	64.8	78.7	71.1

- BMM performs better than state-of-the-art labeling algorithms.
- High F-scores on categories TOP, NP, and VP (77% of all brackets) are responsible for good result. F-scores on PP, ADVP, ADJP considerably less.
- Adaptation of e-Grids for PCFG ('non-uniform distribution') significantly improves the scores.

Results on WSJ

Evaluation: greedy remapping of the experimental labels onto the treebank labels (Haghighi and Klein, 2006)
since # experimental labels $>>$ # treebank labels we also compute scores remapping the other way round.

	LP	LR	F
BMM (uniform)	87.3	31.9	46.7
BMM (3000)	76.3	84.3	80.1
BMM (5000)	75.3	82.3	78.7
Haghighi & Klein '06	64.8	78.7	71.1

- BMM performs better than state-of-the-art labeling algorithms.
- High F-scores on categories TOP, NP, and VP (77% of all brackets) are responsible for good result. F-scores on PP, ADVP, ADJP considerably less.
- Adaptation of e-Grids for PCFG ('non-uniform distribution') significantly improves the scores.

Conclusions and future directions

- BMM performance worse than state-of-the-art on bracketing ...
- ... but better than state-of-the-art on labeling.
- bracketing and labeling processes should perhaps be separated, and there are cognitive arguments to do so.
- or we should think about a different generative model!

Conclusions and future directions

- BMM performance worse than state-of-the-art on bracketing ...
- ... but better than state-of-the-art on labeling.
- bracketing and labeling processes should perhaps be separated, and there are cognitive arguments to do so.
- or we should think about a different generative model!

Conclusions and future directions

- BMM performance worse than state-of-the-art on bracketing ...
- ... but better than state-of-the-art on labeling.
- bracketing and labeling processes should perhaps be separated, and there are cognitive arguments to do so.
- or we should think about a different generative model!

Conclusions and future directions

- BMM performance worse than state-of-the-art on bracketing ...
- ... but better than state-of-the-art on labeling.
- bracketing and labeling processes should perhaps be separated, and there are cognitive arguments to do so.
- or we should think about a different generative model!