

# Online Learning, Perceptrons and Syntactic Analysis

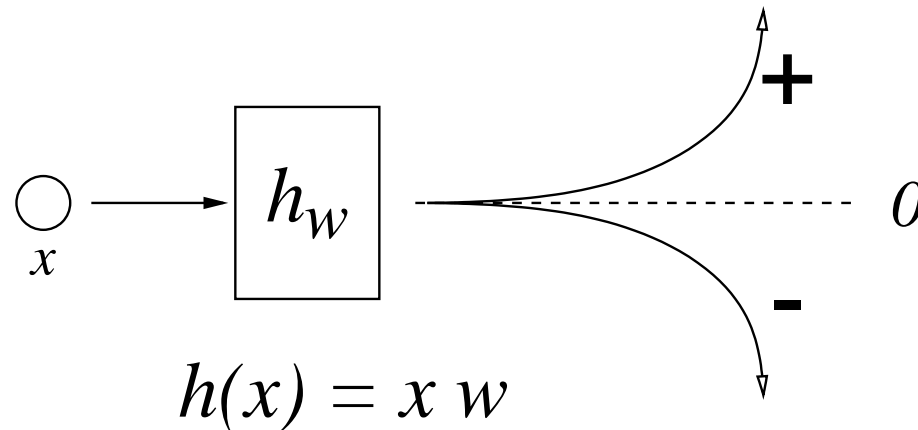
Xavier Carreras and Lluís Màrquez  
Universitat Politècnica de Catalunya

Hondarribiako Workshopa  
February 2004

# Mr. Perceptron (Rosenblatt, 1957)

A simple, traditional online learning algorithm for binary classification.

Works with linear hiperplanes:

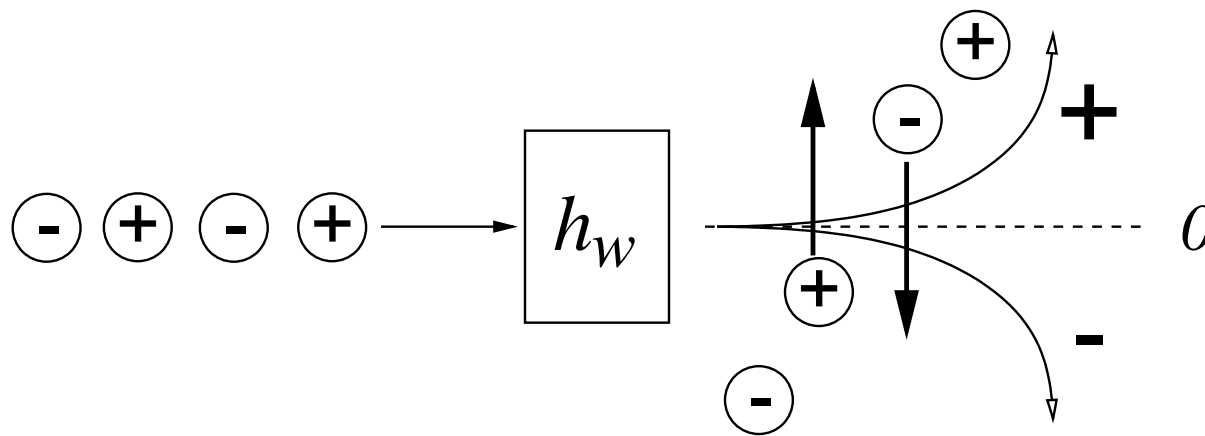


# Perceptron Learning: Binary Classification

Algorithm:

Visit examples  $(x, y)$ ;  $x \in \mathbb{R}^n$ ,  $y \in \{+1, -1\}$ :

1.  $\hat{y} = h_w(x)$
2. if  $\hat{y} \neq y$  then  $w = w + yx$

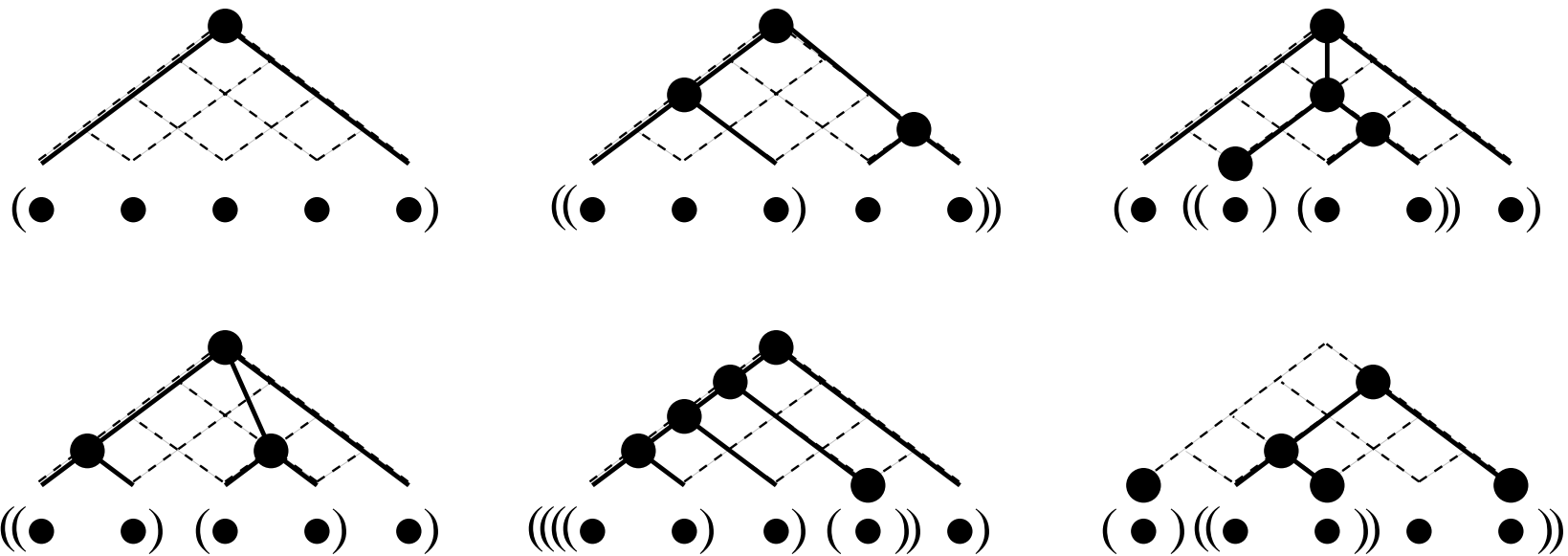


# Syntactic Analysis

- Chunking
- Clause Identification
- Full Parsing
- . . . also Named Entities, PoS Tagging, Semantic Roles

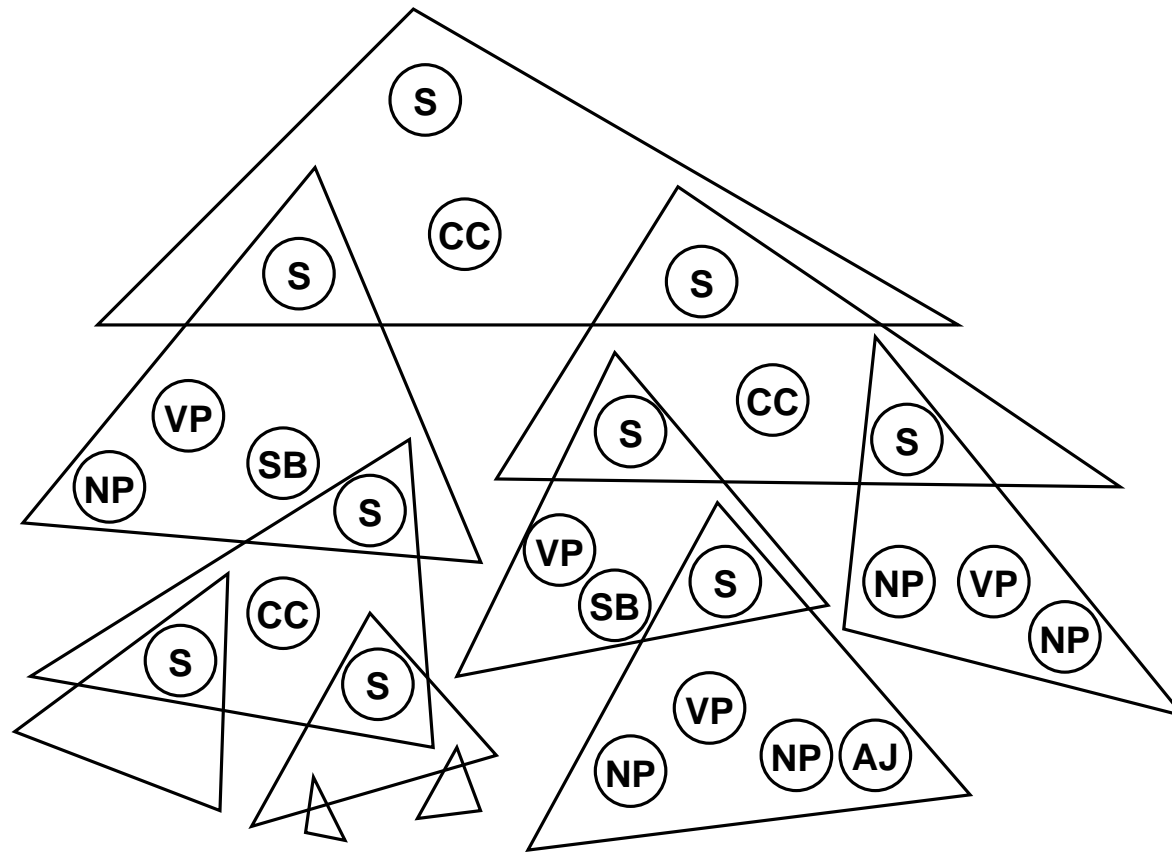


# Observation 1: Huge Output Space



Output space is  $\sim O(2^{n^2})$ : Parsing strategy required.

## Observation 2: Recursive Structures



Desirable to put learning in high-order level.

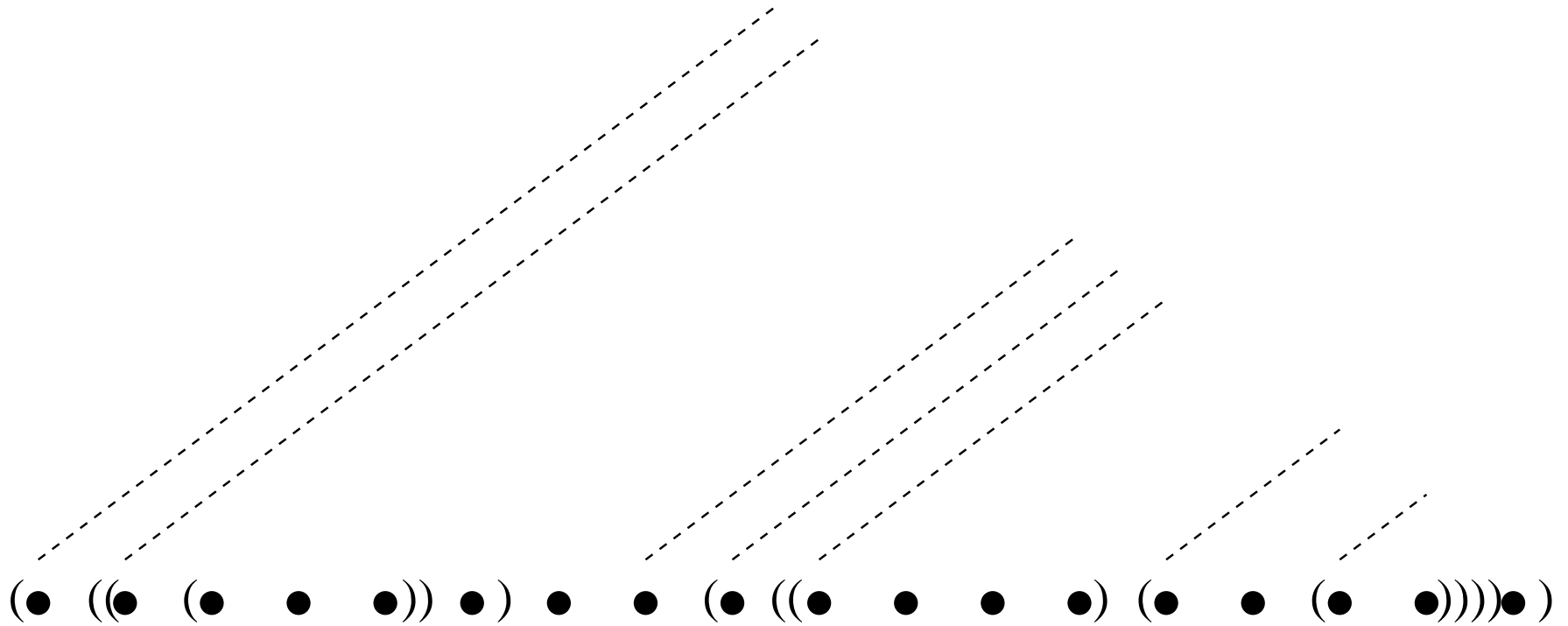
# Filtering-Ranking Strategy (i)

1. Generate phrase candidates with Start-End functions.
2. Score each phrase candidate.
3. Build the hierarchy with best phrases.

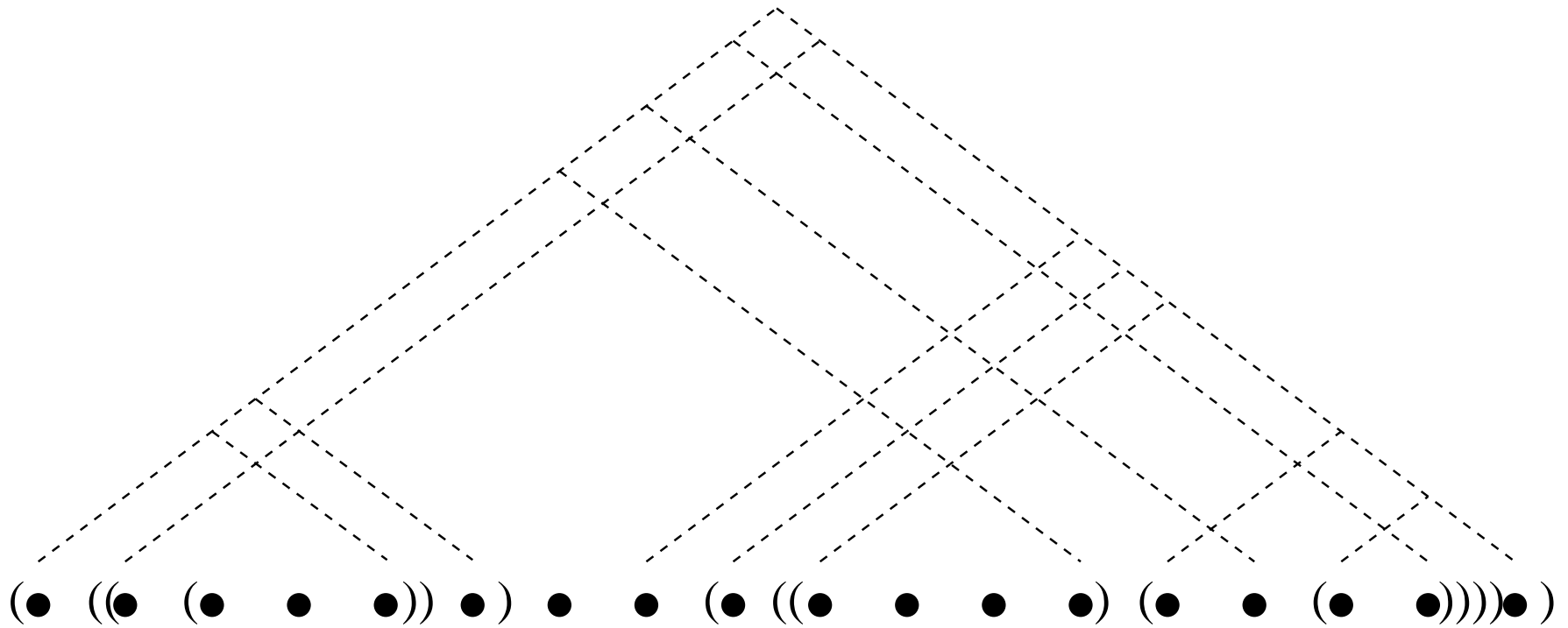
# Filtering-Ranking Strategy (ii)

(• ((• (• • •)) •) • • (• ((• • • •) (• • (• •)))•))

# Filtering-Ranking Strategy (ii)



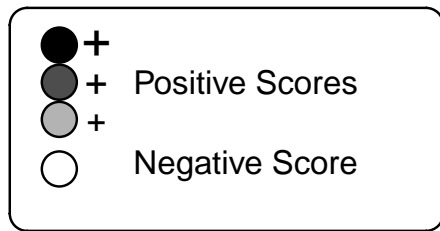
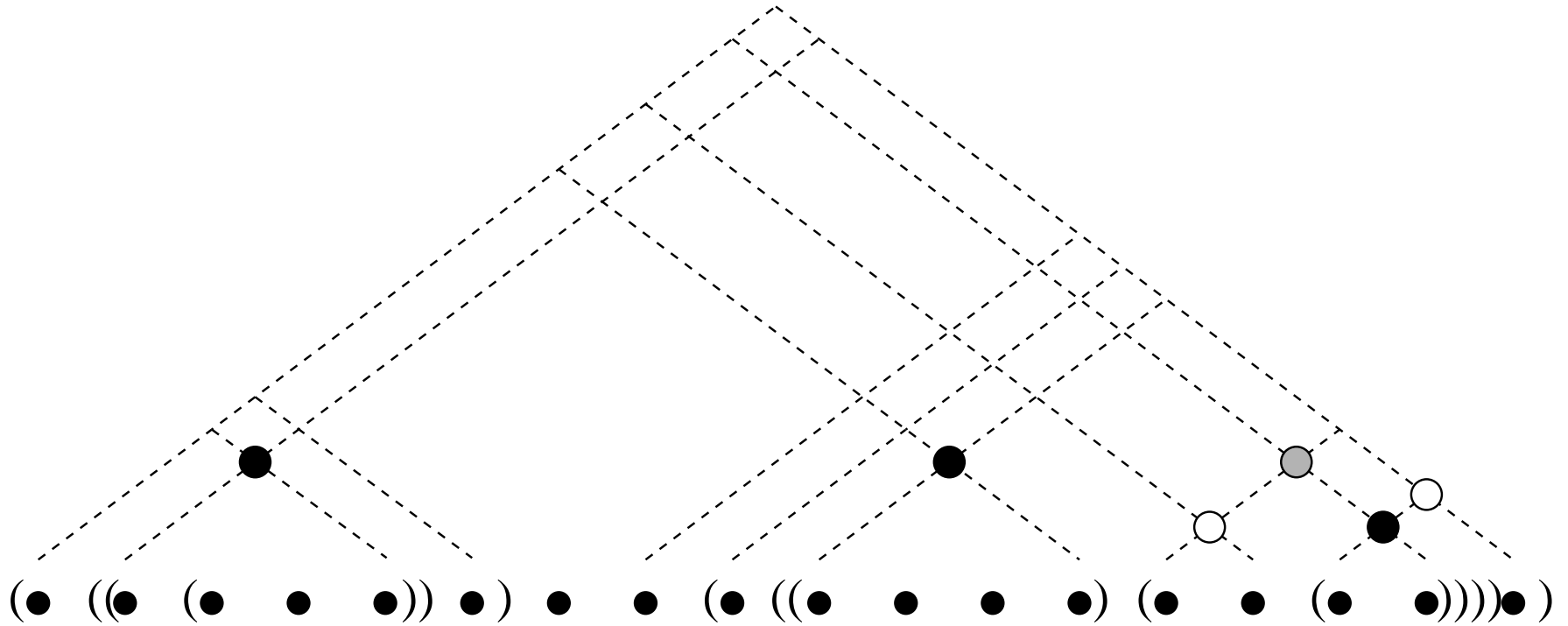
# Filtering-Ranking Strategy (ii)



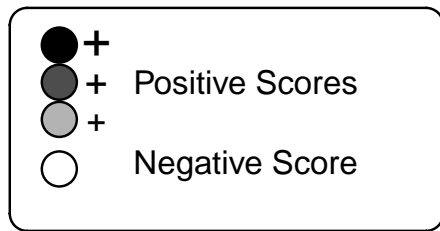
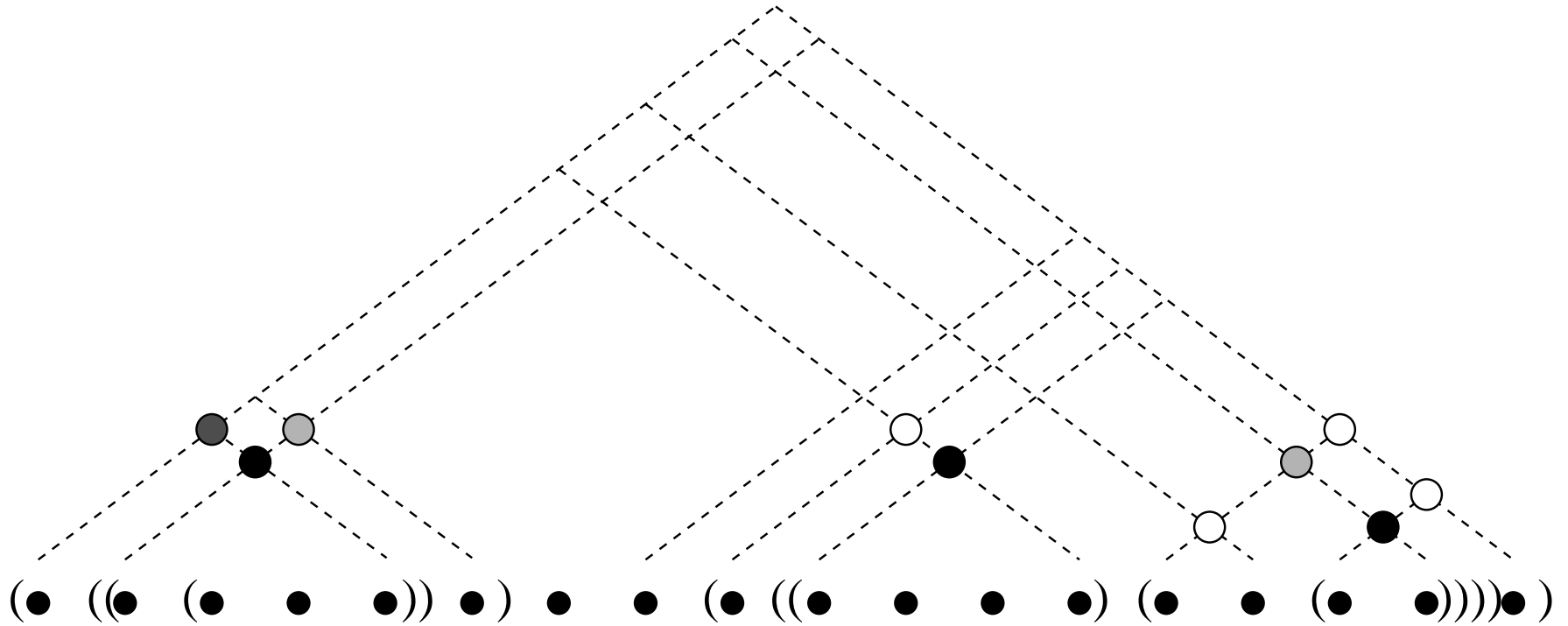




# Filtering-Ranking Strategy (ii)

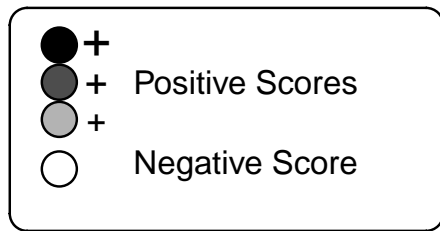
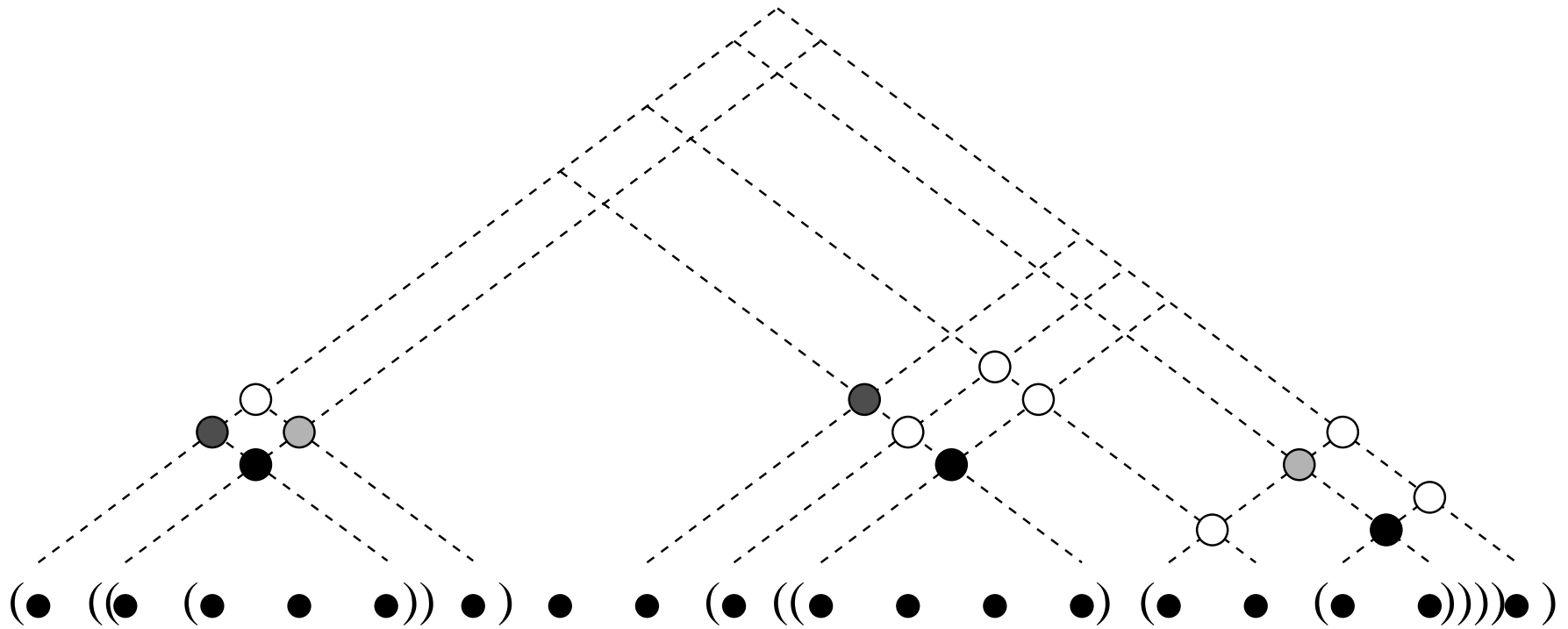


# Filtering-Ranking Strategy (ii)





# Filtering-Ranking Strategy (ii)



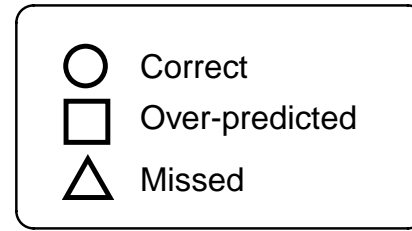
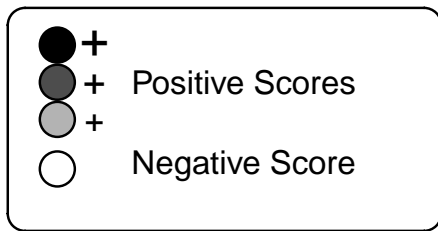
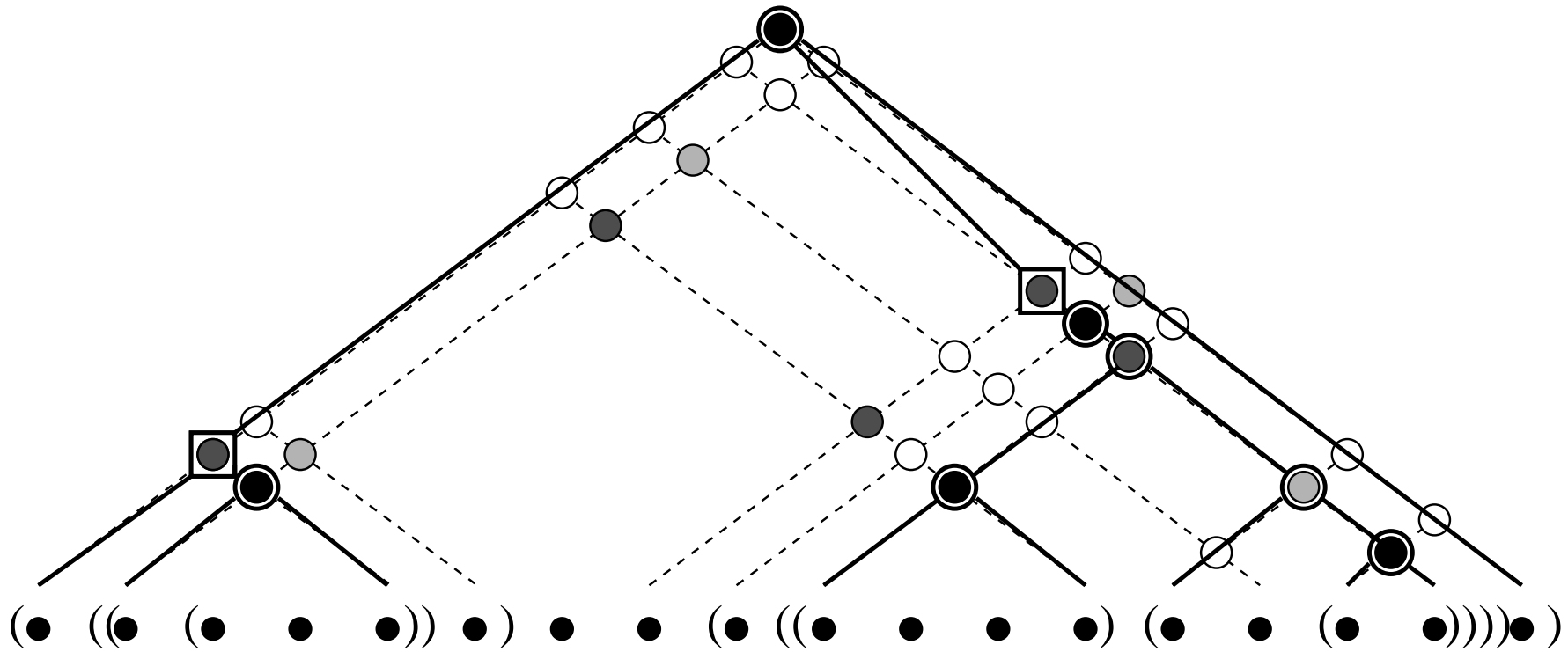




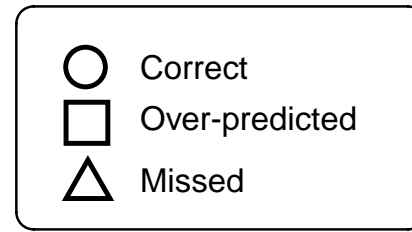
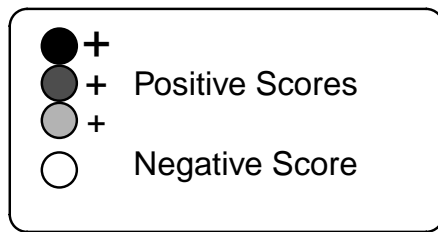
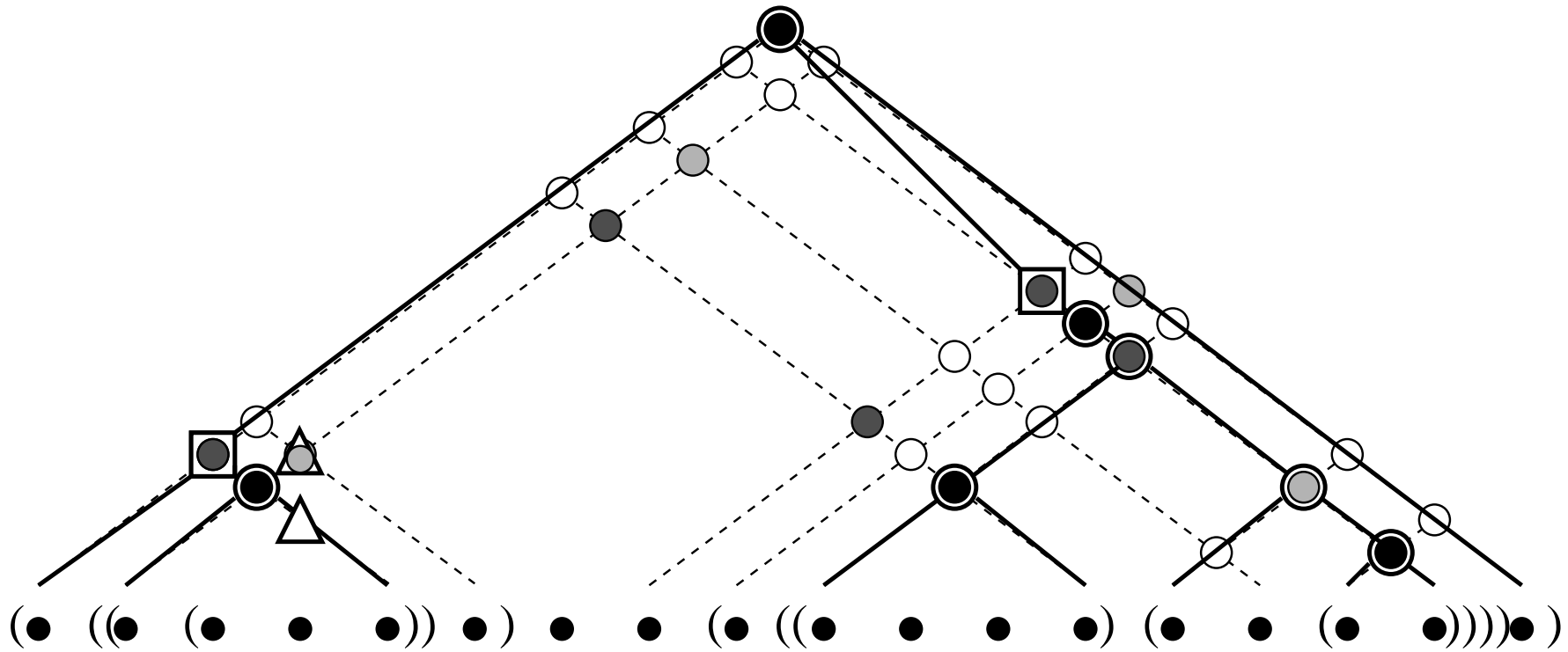




# Filtering-Ranking Strategy (ii)



# Filtering-Ranking Strategy (ii)



## Filtering-Ranking Strategy (iii)

$\mathcal{Y}$ : **solution space**, i.e. set of all coherent phrase sets.

$\mathcal{Y}_{SE}$ : **practical solution space**, filtered at word level.

$$R(x) = \arg \max_{y \in \mathcal{Y}_{SE}} \sum_{(s,e) \in y} \text{score}(s, e, x, y')$$

$$\mathcal{Y}_{SE} = \{y \in \mathcal{Y} \mid \forall (s, e) \in y \text{ start}(s) \wedge \text{end}(e)\}$$

- Sequential case:  $O(n^2)$  Dynamic Prog. search
- Hierarchical case:  $O(n^3)$  Dynamic Prog. search

# Learning with Perceptrons

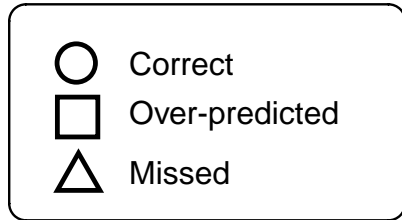
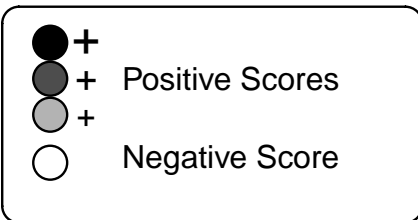
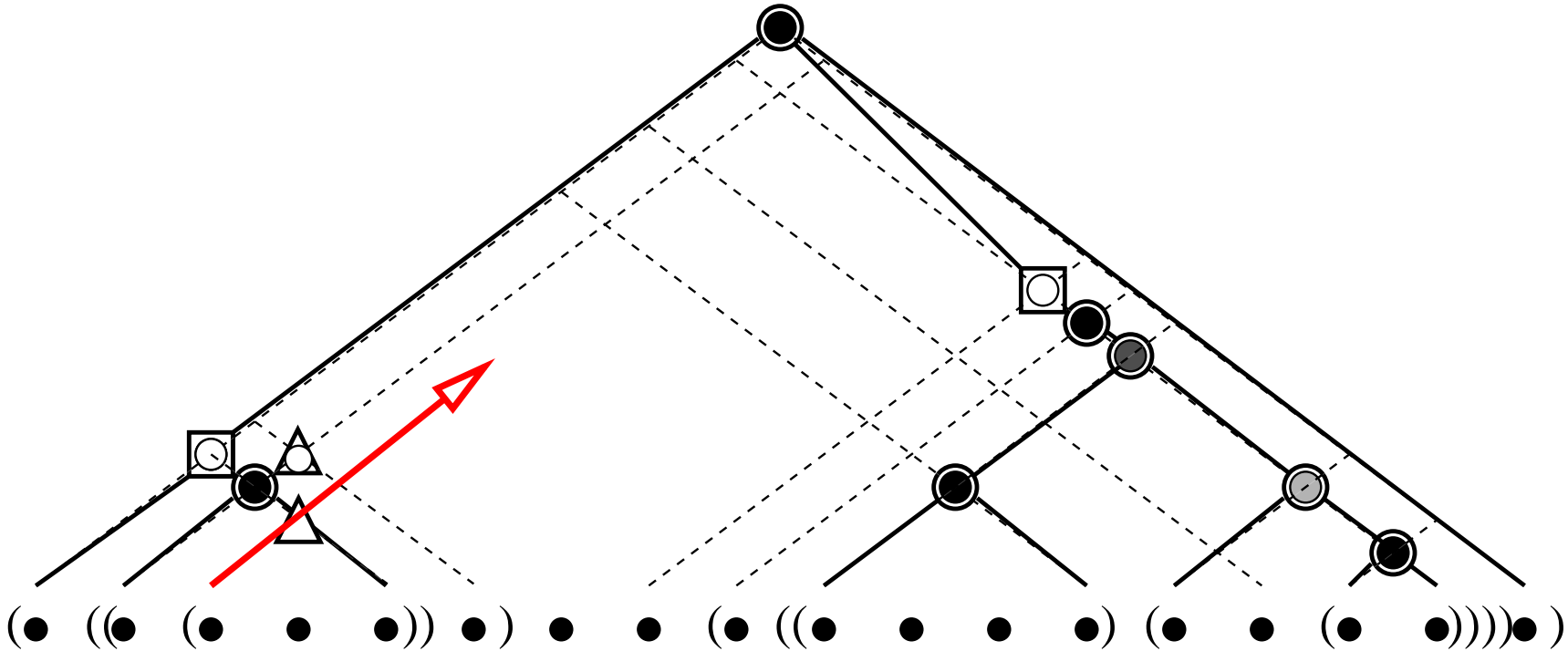
- All functions are learned together, while visiting online training sentences.
- Algorithm: Given a sentence:
  1. Predict the phrase hierarchy.
  2. Mistake-driven approach: Identify errors and provide feedback.

We consider only errors at global level:

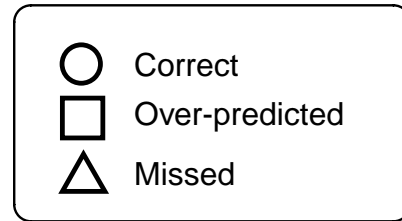
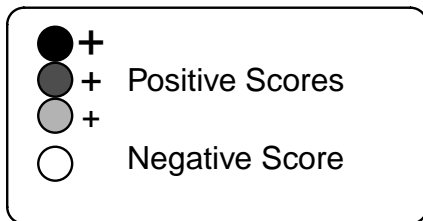
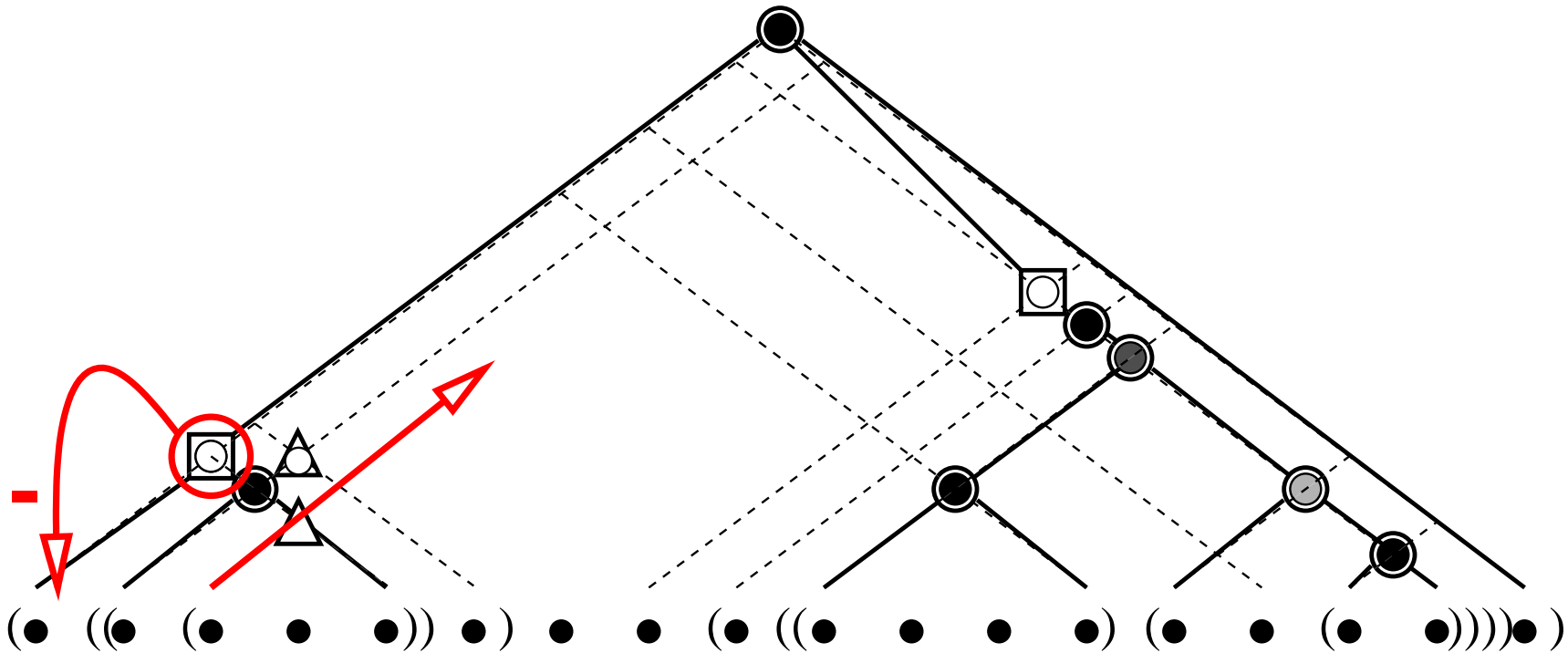
    - ★ Missed Phrases
    - ★ Over-predicted Phrases



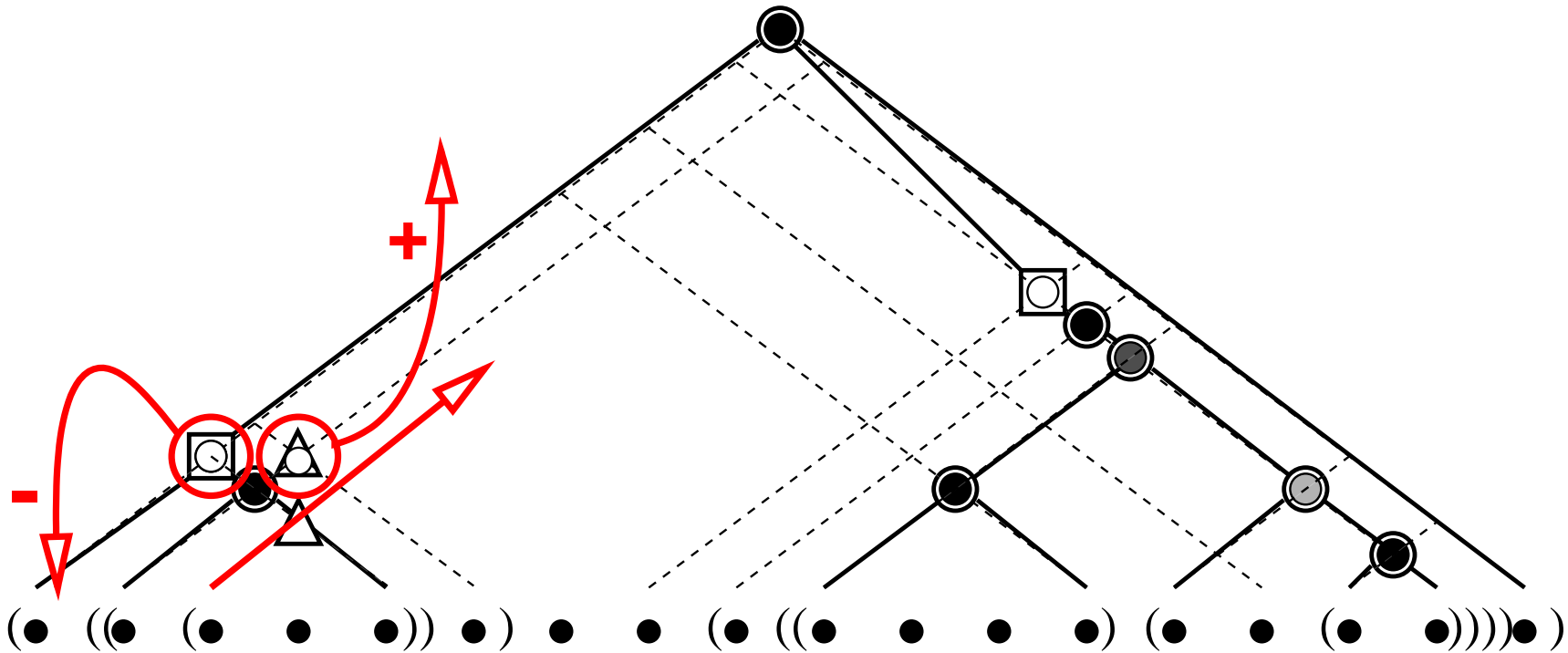
# Learning Feedback



# Learning Feedback



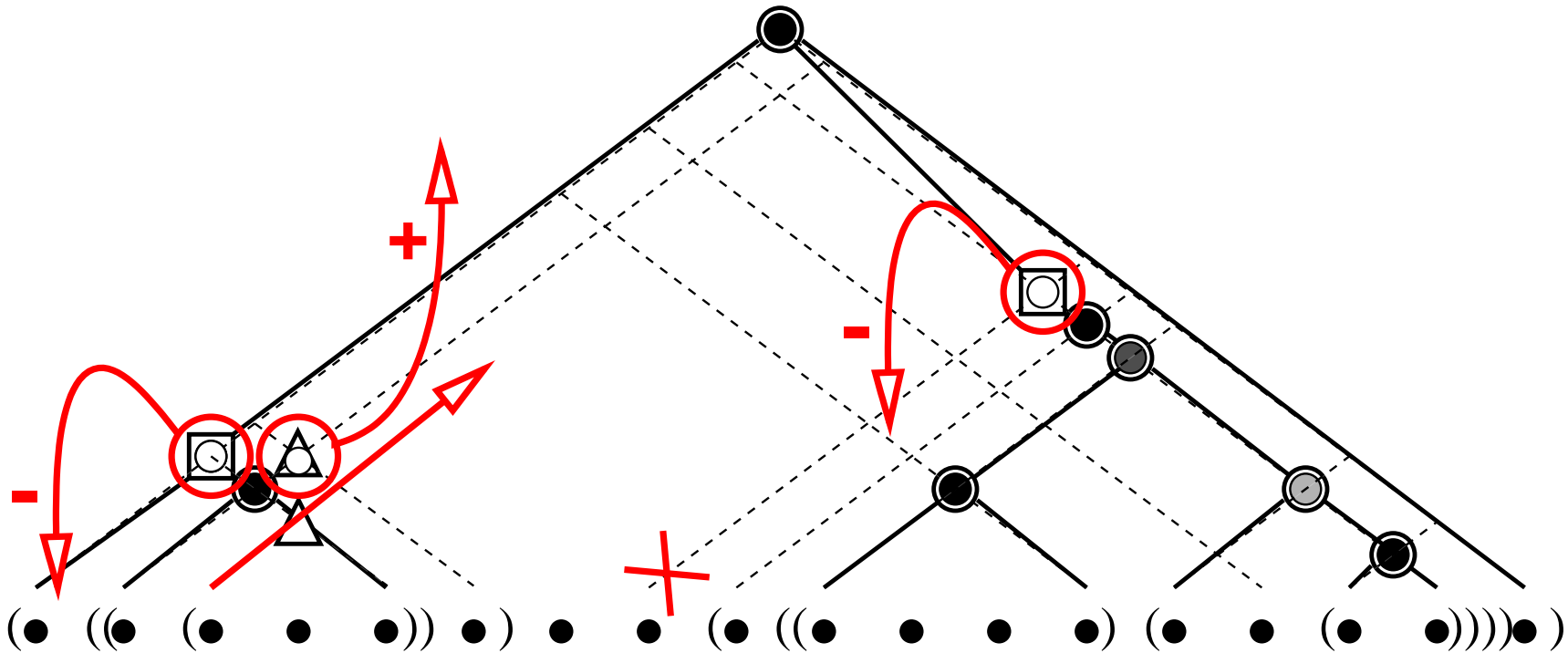
# Learning Feedback



●	+
●	+ Positive Scores
●	+ Negative Score
○	Negative Score

○	Correct
□	Over-predicted
△	Missed

# Learning Feedback



●+	+
●+	Positive Scores
●+	+
○	Negative Score

○	Correct
□	Over-predicted
△	Missed

# On Representation

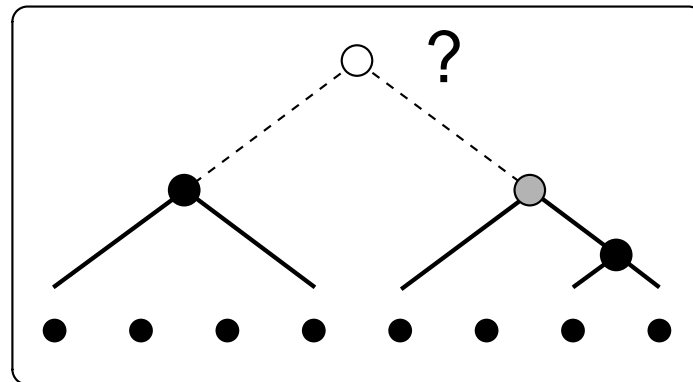
Each learning instance (words, phrases) and its context is represented in a designed feature space.

A key point is that the feature space provides to perceptrons enough expressivity for:

- Making Start words, or End words, distinct from other words, via window-based features.
- Making competing phrases distinct, via high-order features.

# High-Order Features

When visiting a phrase, the internal structure is already computed:



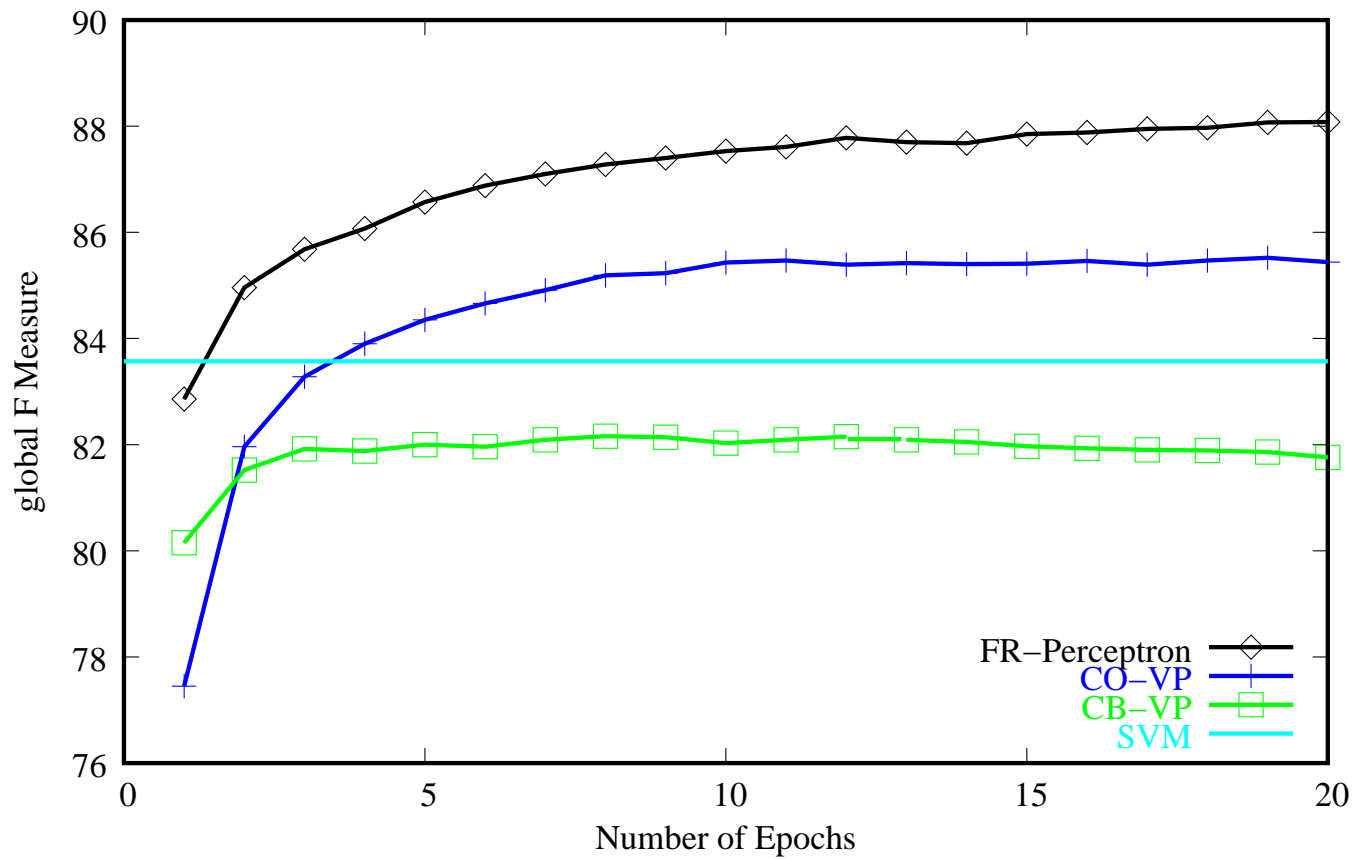
Exploitation through patterns and constraints:

- Linguistically-motivated, grammar-based.
- Kernels, ie. exhaustive exploration of the structure.

# Final Architecture

- Exploration, incrementality, local decisions, inference.
- Learning at high order (beyond words).
- Include partially build solutions in predictions.
- Online approach (vs. batch). Allows to:
  - ★ Learn all components together, capturing interactions.
  - ★ Model the functions so as to optimize its behavior within the parser, e.g. as filters and rankers.

# Experiments on Clause Identification



# Results on Clause Identification CoNLL-2001

	T	prec.	recall	$F_{\beta=1}$
CB-VP	8	82.22	78.09	80.10
SVM	-	83.19	80.00	81.57
CO-VP	19	89.25	77.62	83.03
FR-Perceptron	20	88.17	<b>82.10</b>	<b>85.03</b>
AdaBoost (1)	-	<b>90.18</b>	78.11	83.71

(1) (Carreras, Màrquez, Punyakanok and Roth, ECML'02)

# Results on Chunking CoNLL-2000

	technique	prec.	recall	$F_1$
(Zhang 02)	Winnnow ++	94.28	94.07	94.17
(Kudo & M. 01)	SVM voting	93.89	93.92	93.91
(Kudo & M. 01)	SVM single	n/a	n/a	93.85
<b>FRP-Chunker</b>	<b>F&amp;R VP</b>	94.20	93.38	93.79
(Zhang 02)	Winnnow	93.54	93.60	93.57
<b>BI-Chunker</b>	<b>greedy VP</b>	92.83	92.21	92.52

# Final Views

- Flexible learning architecture for recovering structure:
  - ★ The parsing strategy defines the dependencies to be exploited.
  - ★ With Perceptron, the parser functions are easily adapted to work within the parser.
  - ★ External (linguistic) knowledge welcome!
- But . . .
  - ★ Programming: all modules have to work together.
  - ★ Lots, LOTS! of instances, features, inputs, outputs.
  - ★ High computing power needed.

# Current and Future works

- Partial Parsing, all-in-one: PoS+Chunks+Clauses
- Full Parsing, w/wo grammar guidance
- Propositional Analysis and Semantic Roles
- Alignment and Machine Translation

# References

- X. Carreras and L. Márquez, *Phrase Recognition by Filtering and Ranking with Perceptrons*, Recent Advances in NLP, 2004.
- M. Collins, *Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms*, in Proceedings of the EMNLP-2002, 2002.
- Y. Freund and R. Schapire. *Large Margin Classification Using the Perceptron Algorithm*, Machine Learning 37, 1999.
- V. Punyakanok and D. Roth. *The Use of Classifiers in Sequential Inference*, in Proceedings of NIPS-15, 2001.

**Eskerrik Asko!**