German-Japan Workshop
5 July, 2003

Word Dependency Parsing with Support Vector Machines

Yuji Matsumoto (NAIST) **and** **Hiroyasu Yamada** (JAIST)

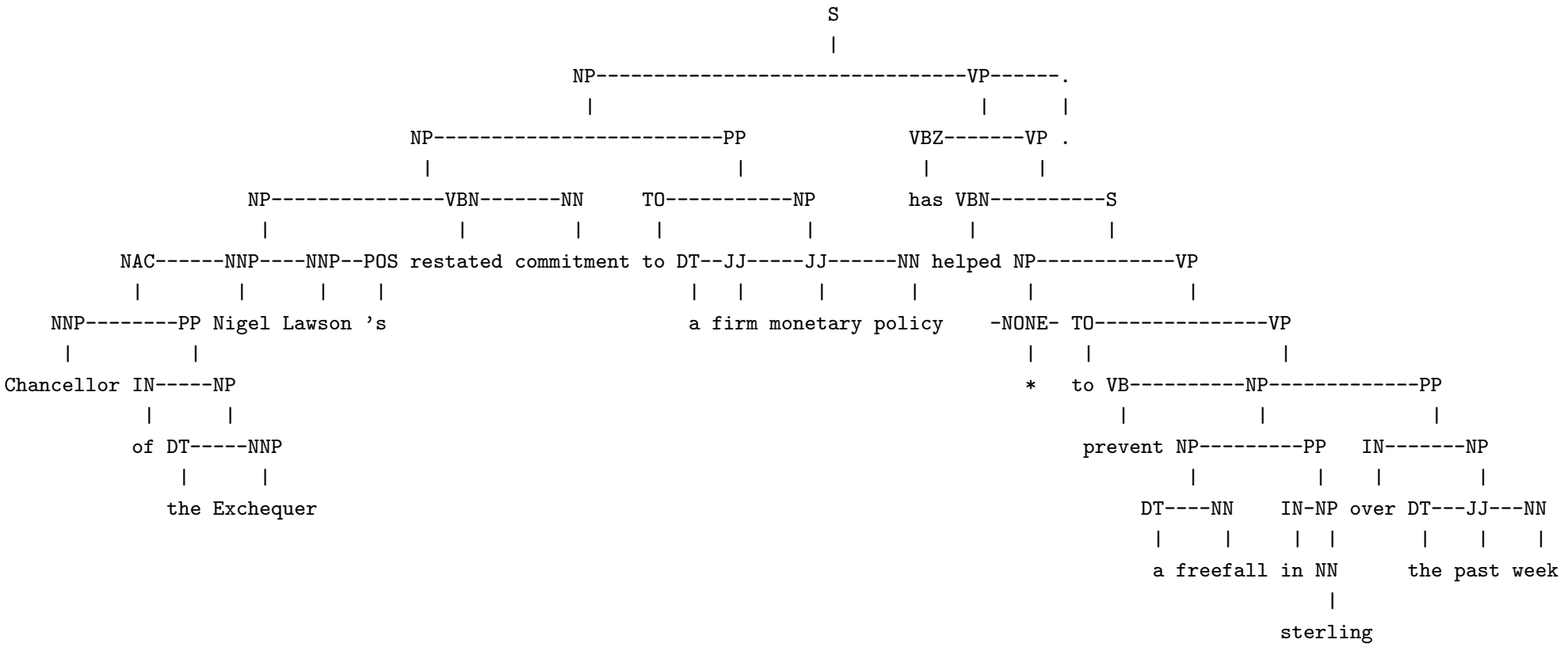
Statistical Parsing of English

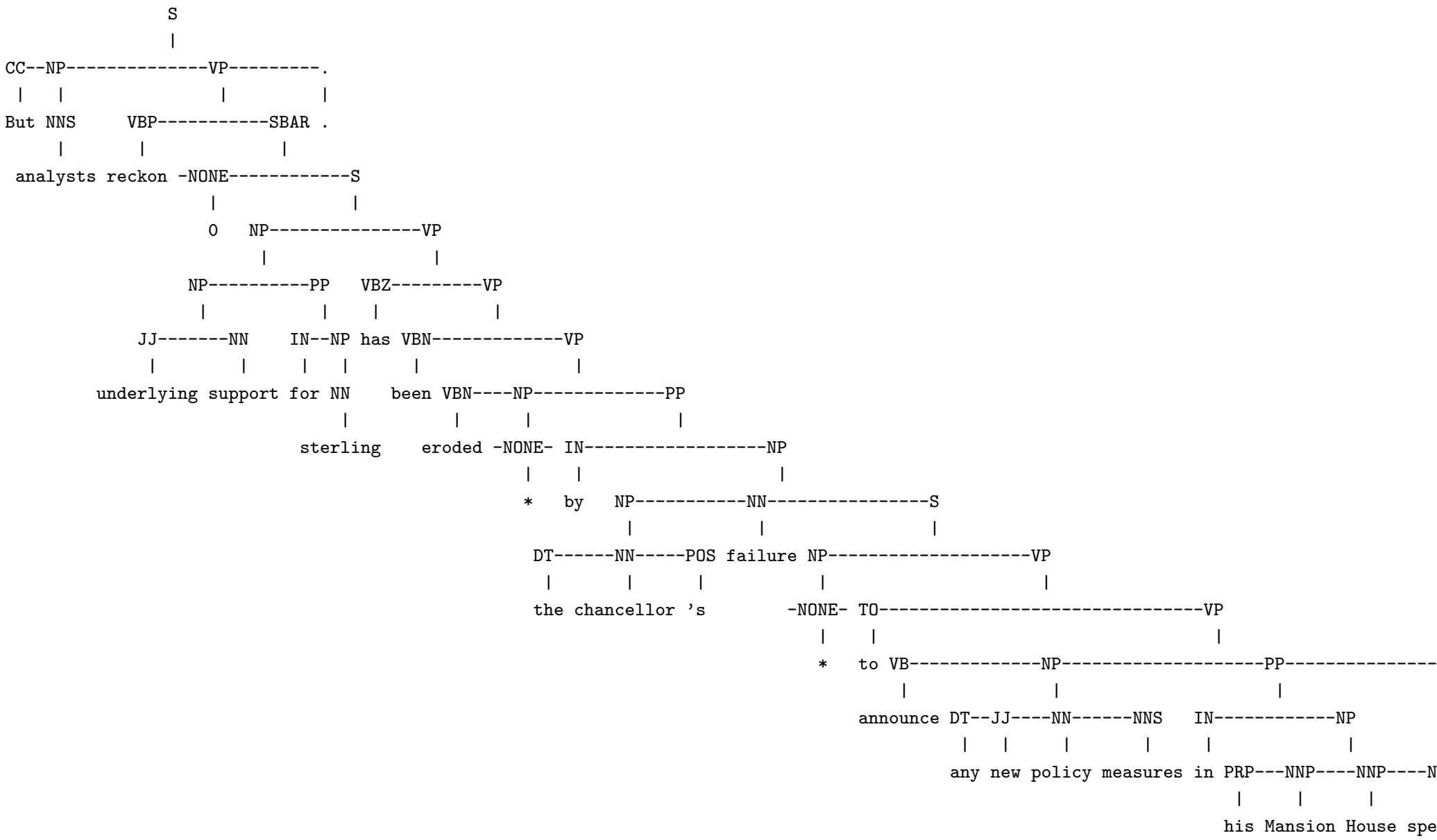
- Early 90's: Probabilistic CFGs (Inside-Outside Algorithm)
- From mid-90's: Machine Learning based Parsing
  - Decision Tree Parsing[Magerman 95]
  - Generative Lexicalised Models [Collins 96,97,99]
  - Maximum Entropy Parsing [Ratnaparkhi 97]
  - PCFG + Maximum Entropy[Charniak 99,00]
  - Reranking by Tree Kernel [Collins 02]
- All parsers learn from and produce Penn-style Phrase Structure Trees

Penn Treebank annotation

- Phrase Structure Trees with more than a hundred types of phrase labels
- Relatively flat phrase structure rules
- Quite a few types of empty catetories (`-NONE-`)
- $\Rightarrow$ Annotators must have a competent knowledge of English Grammars

Typecal Penn Treebank Trees

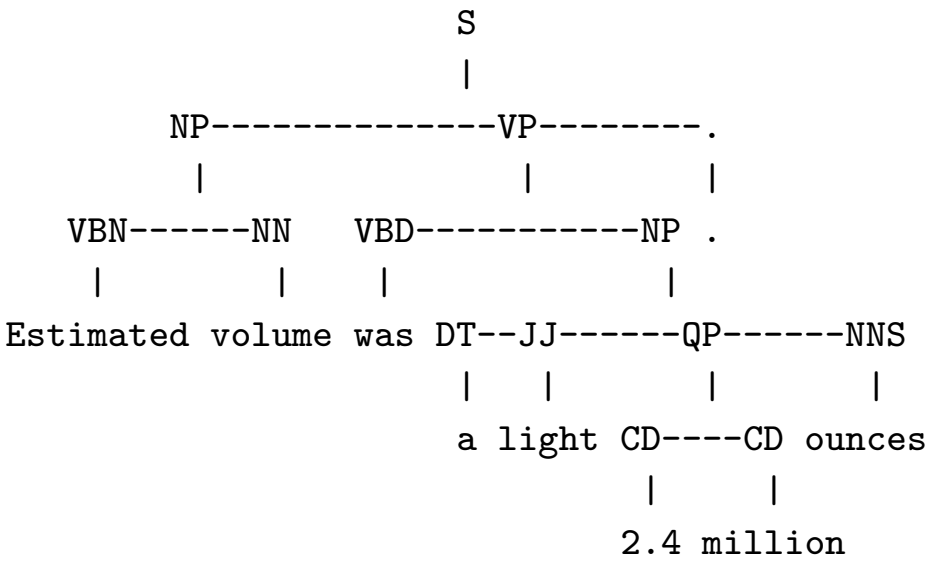


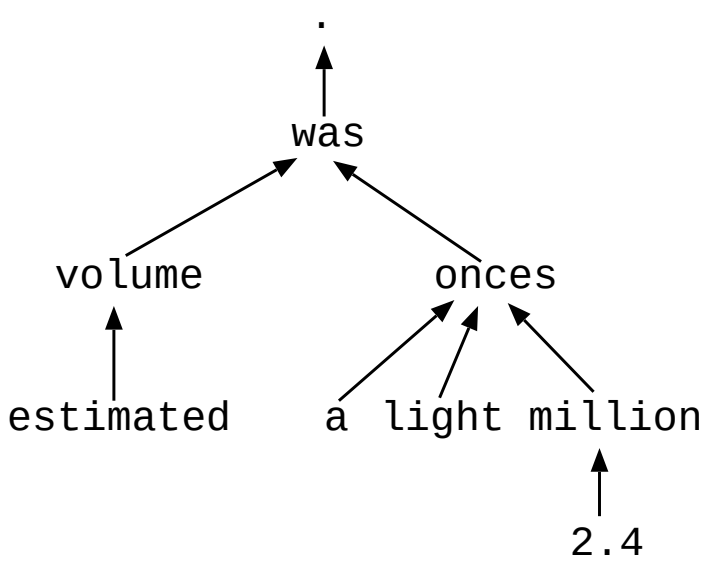

S
CC NP VP .
But NNS VBP SBAR .
analysts reckon -NONE- S
0 NP VP
NP PP VBZ VP
JJ NN IN NP has VBN VP
underlying support for NN been VBN NP PP
sterling eroded -NONE- IN NP
* by NP NN S
DT NN POS failure NP VP
the chancellor 's -NONE- TO VP
* to VB NP PP
announce DT JJ NN NNS IN NP
any new policy measures in PRP NNP NNP N
his Mansion House spe

Word Dependency Parsing

- Word dependency does not heavily rely on specific grammar formalisms
- Easily transformable from phrase structure trees
  - Only if the heads of phrase structure rules are defined
- Efficient and practical in constructing traning data
  - Word dependencies are much easier to understand than phrase structures
  - More intuitive for English native speakers
- Learning and parsing algorithms may be much simpler

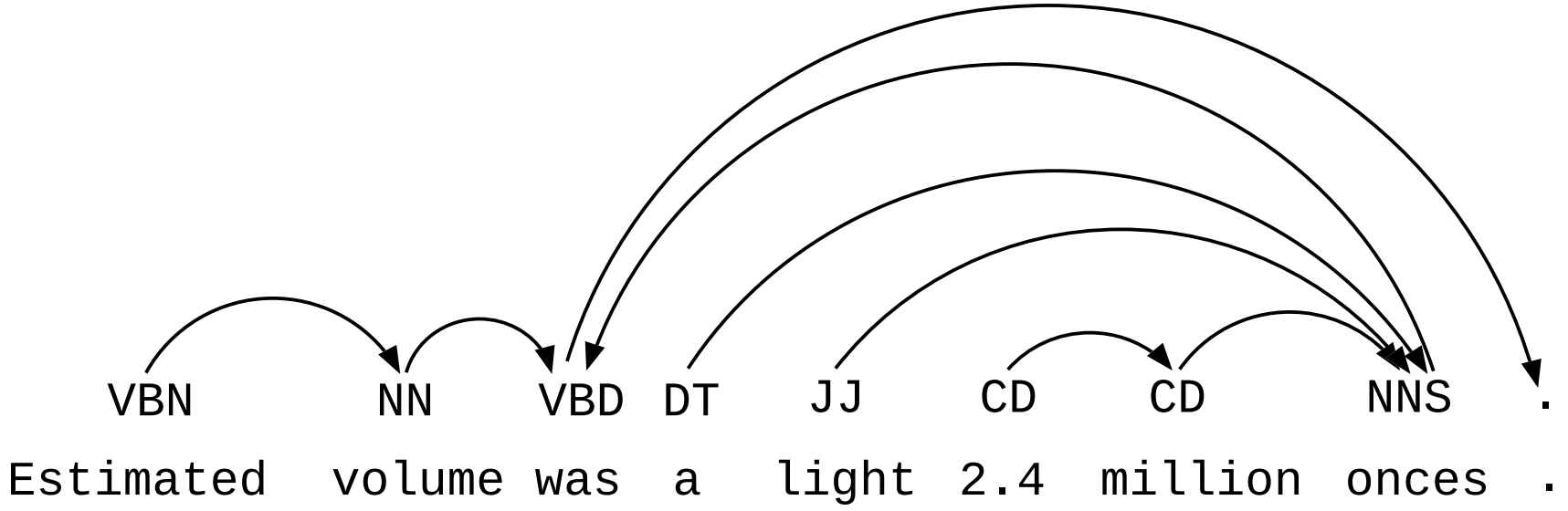
Phrase Structure Tree and Dependency Tree





(Penn Treebank)

Flattened Representation of Dependency Tree



Purposes and Claims of this Research

- It is not efficient to annotate sentences in expert domains in Penn-style phrase sturucture.
- Some IE or knowledge acquisition tasks rely on syntactic sturucture, not only on bag-of-words or N-gram statistics. The essential syntactic strucure required is not phrase structure but word dependency structure.
- Syntactic annotation to Medline abstracts: Dependency structure is more comprehensible to medical/biological scientists.
- Statistical parsing may be easier for dependency parsing.

Deterministic Dependency Parsing (Three action model)

Right: Between two adjacent nodes (words), the left node modifies to the right node, and dependency relation is allocated.

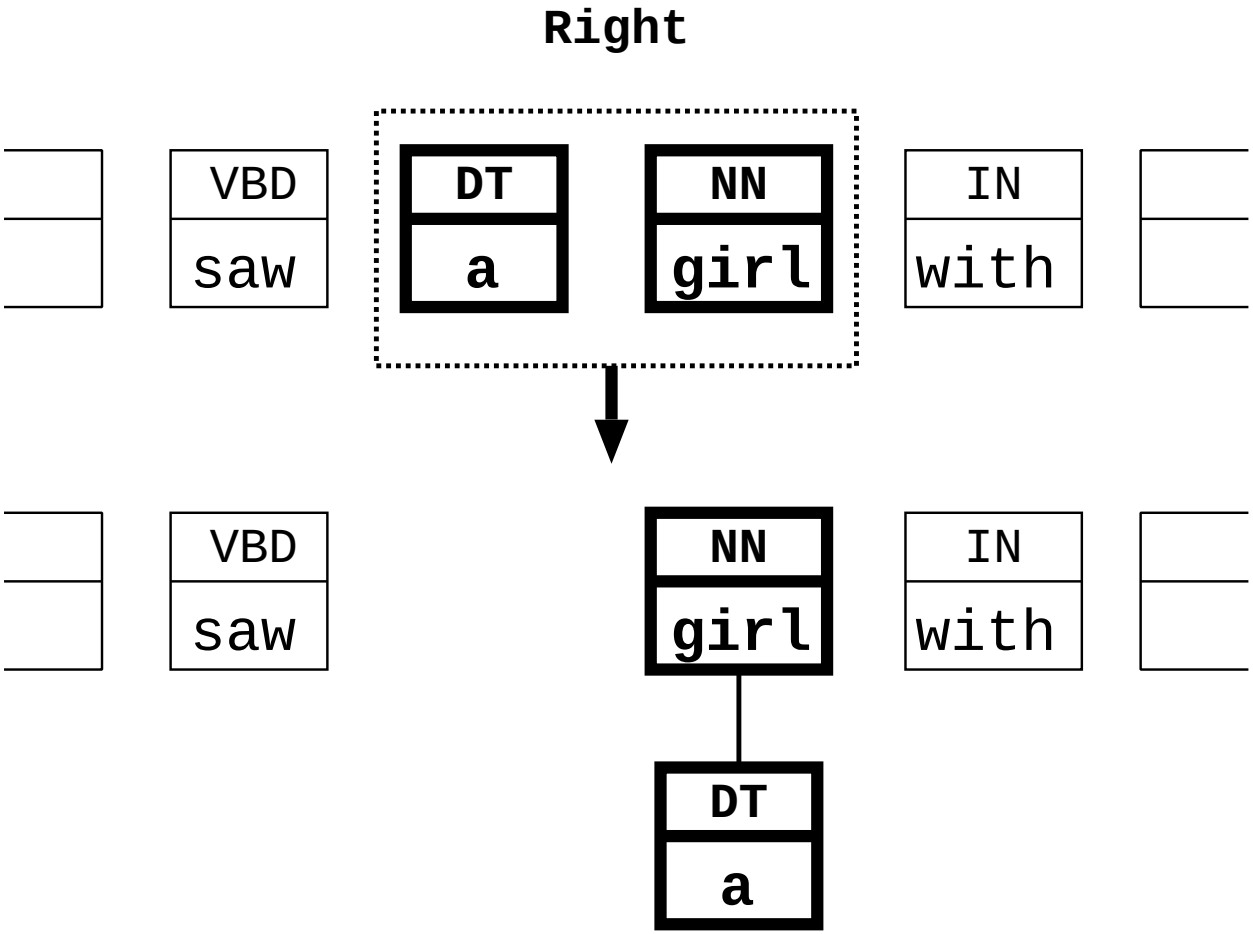
Left: Between two adjacent nodes (words), the right node modifies to the left node, and dependency relation is allocated.

Shift: Not to construct any dependency tree and the focus position is is moved to the right (or to the left).
There are two cases in this action:

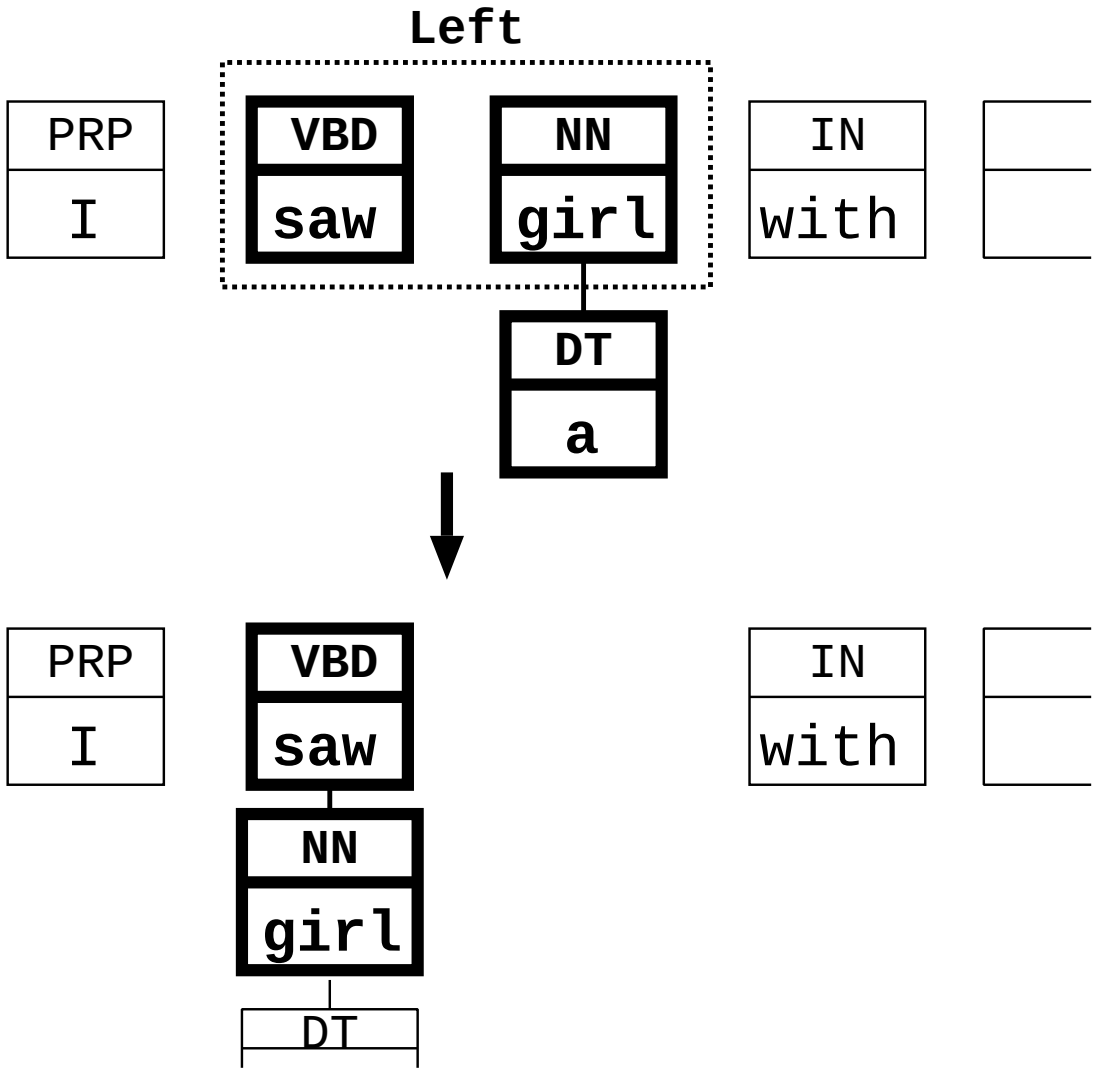
1. There is no dependency relation between the two words.
2. There is a dependency relation between the two words, but the action shouldn't be applied now.

The direction of parsing may be from left to right, or from rihgt to left.

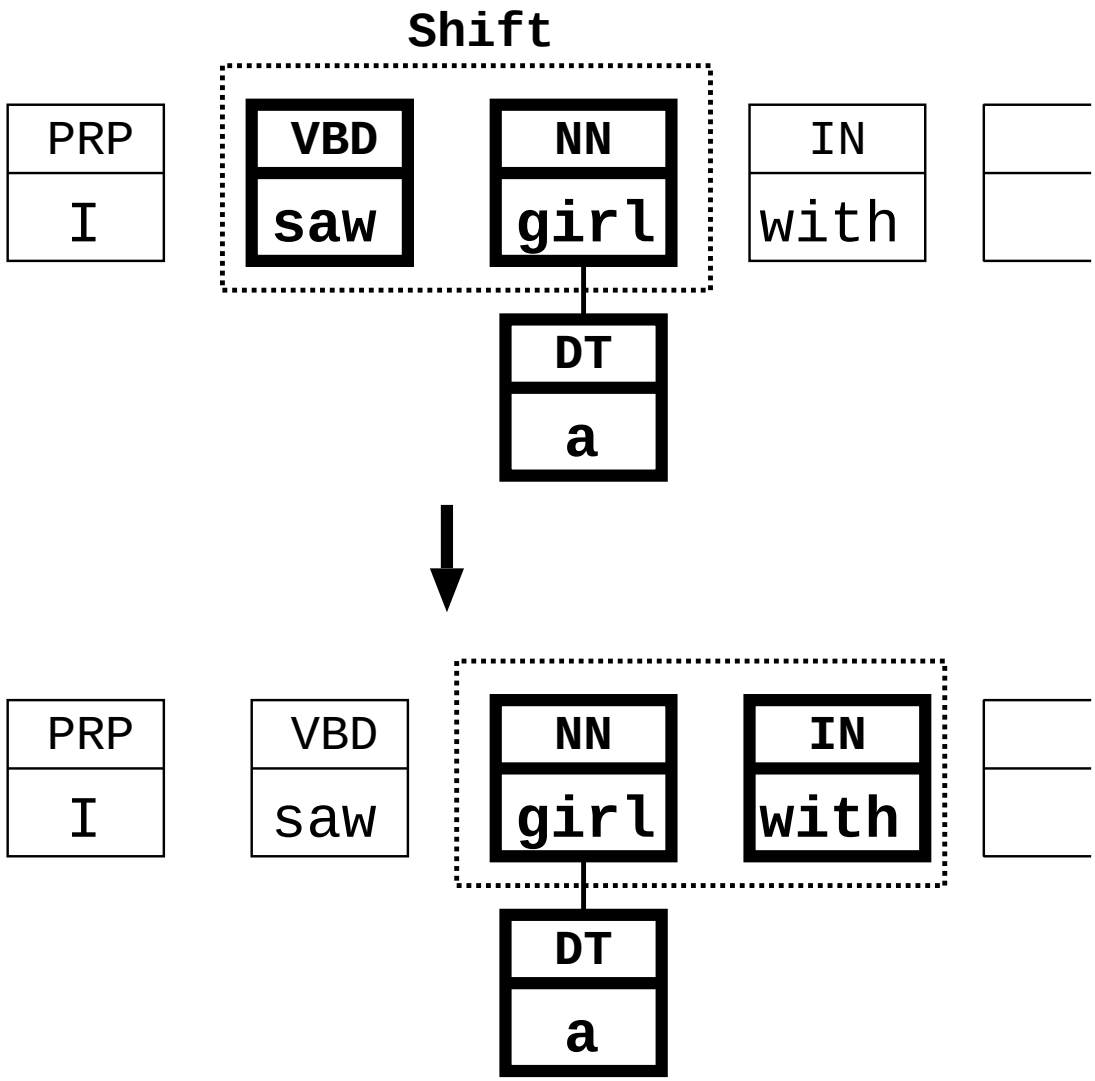
Right Action



Left Action


Left
PRP
I
VBD
saw
NN
girl
IN
with
DT
a
PRP
I
VBD
saw
IN
with
NN
girl
DT

Shift (when parsed from left to right)



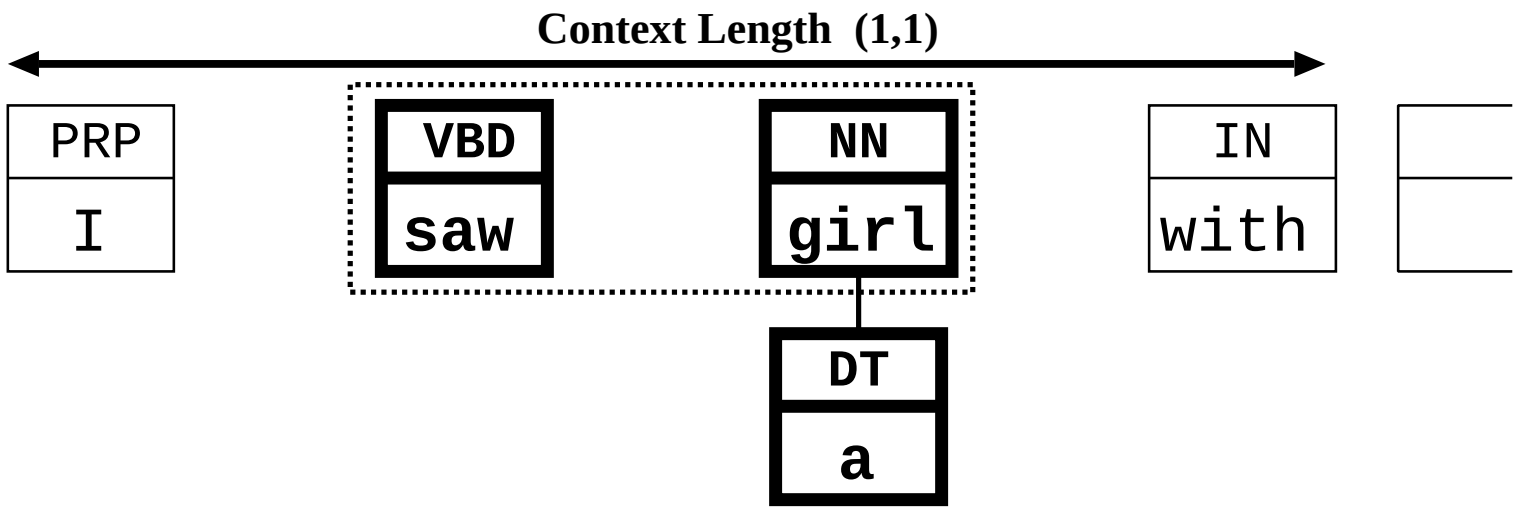
Learning for Dependency Parsing

- Training examples are collected through parsing the annotated sentences. (An example means a pair of configuration and the action class)
- Support Vector Machines are constructed by *pair-wise* method between action classes.
  (Examples are grouped according to the POS tag at the focus position. Non-grouping case is also tested.)

Execution of Dependency Parsing

- The focus point moves either from the beginning or from the end of the sentence.
- Pair-wise SVMs are applied to the configuration at the focus point, and the action class of the winner is selected and applied.
- The focus point moves only when "Shift action" is selected.
- When the focus point reaches at the end or the beginning of the sentence, the process is repeated until there remains a single word.
- The test data (Penn Treebank: Section 23) are POS tagged by an SVM-based POS tagger [Nakagawa, Kudo, Matsumoto 02]

Configuration Defining Features




```
-1.lex.I     0-.lex.saw   0+.lex.girl   +1.lex.with
-1.pos.PRP   0-.pos.VBD   0+.pos.NN     +1.pos.IN
                          0+.pos(L).DT
                          0+.lex(L).a
```

The Features

features		values
static	pos	POS tags
	lex	word forms
dynamic	pos(L)	POS tag of the word that modifies from the left
	lex(L)	the word that modifies from the left
	pos(R)	POS tag of the word that modifies from the right
	lex(R)	the word that modifies from the left
position		$-2,\ -1,\ 0-,\ 0+,\ +1,\ +2$

Evaluation

Traning and Test Data:

- Penn Treebank trees are transformed into word dependency trees. (Head rules [Collins 99] are used to identify dependency relations)
- Traning data: section 02 to 21, Test data: section 23

Evaluation measures:

$$\text{Dependency Accuracy} = \frac{\text{number of correct dependency relations}}{\text{total number of dependency relations}}$$

$$\text{Root Accuracy} = \frac{\text{number of correct root nodes}}{\text{the total number of sentences}}$$

$$\text{Complete Rate} = \frac{\text{number complete parsed sentences}}{\text{the total number of sentences}}$$

Experiments

The following aspects are evaluated.

- The degrees of the polynomial kernels used in SVMs
- The effect of context length
- The effect of parsing direction
- The effect of dynamic features (information of children)
- Comparison with related work

Degrees of Polynomial Kernels

The degree of d : Taking account of combination of d features.

context length: (2, 2)

	d : $(\mathbf{x}' \cdot \mathbf{x}'' + 1)^d$			
	1	2	3	4
Dep. Acc.	0.854	**0.900**	0.897	0.886
Root Acc.	0.811	**0.896**	0.894	0.875
Comp. Rate	0.261	**0.379**	0.368	0.346

Effect of Context Length

kernel function: $(\mathbf{x}' \cdot \mathbf{x}'' + 1)^2$

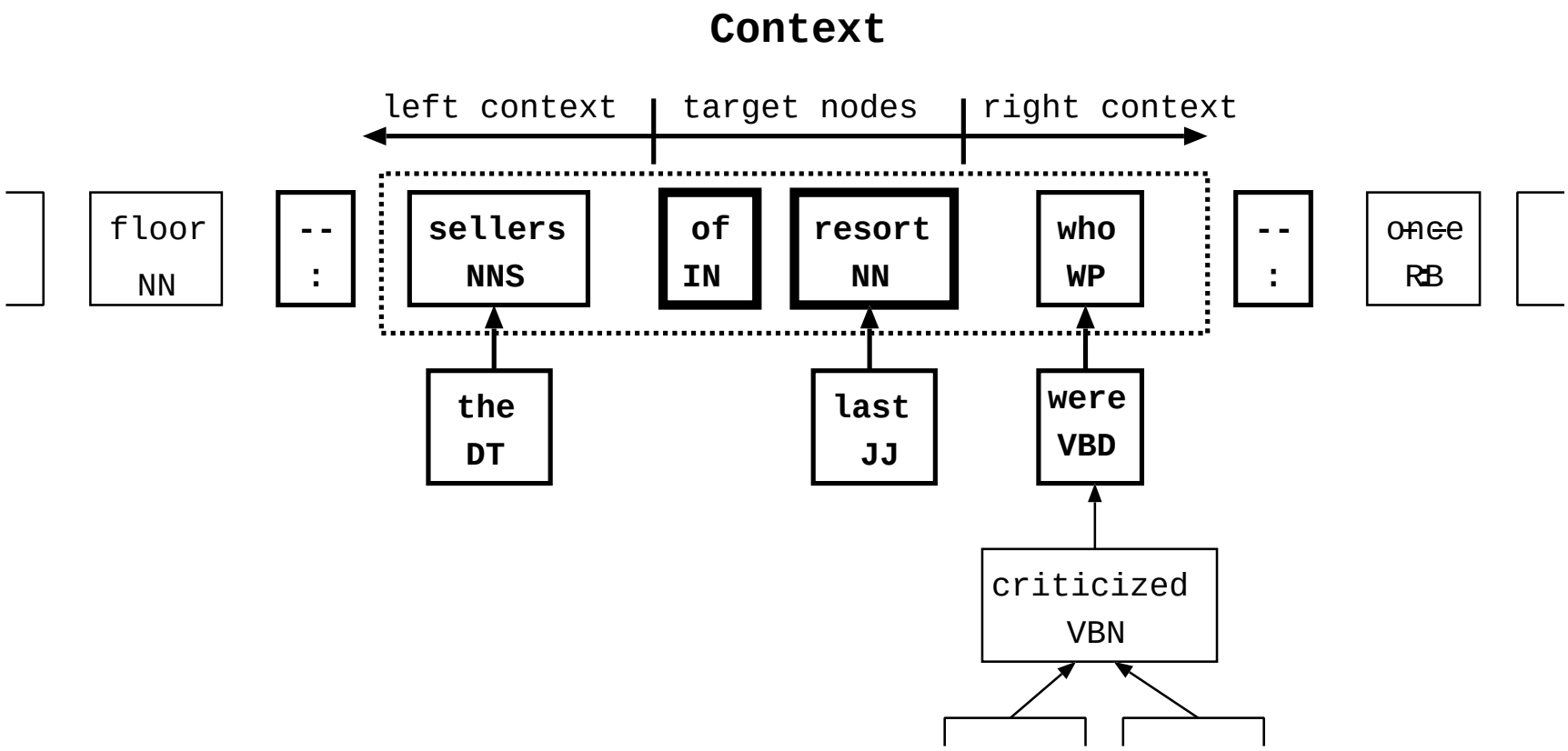
	(l, r): *context length*			
	(2, 2)	(2, 3)	(2, 4)	(2, 5)
Dep. Acc.	0.900	**0.903**	**0.903**	0.901
Root Acc.	0.896	0.911	**0.916**	0.913
Comp. Rate	0.379	0.382	0.384	0.375
	(3, 2)	(3, 3)	(3, 4)	(3,5)
Dep. Acc.	0.898	0.902	0.900	0.897
Root Acc.	0.897	0.915	0.912	0.909
Comp. Rate	0.373	**0.387**	0.373	0.366

Effect of Parsing Direction

context = (2, 2):

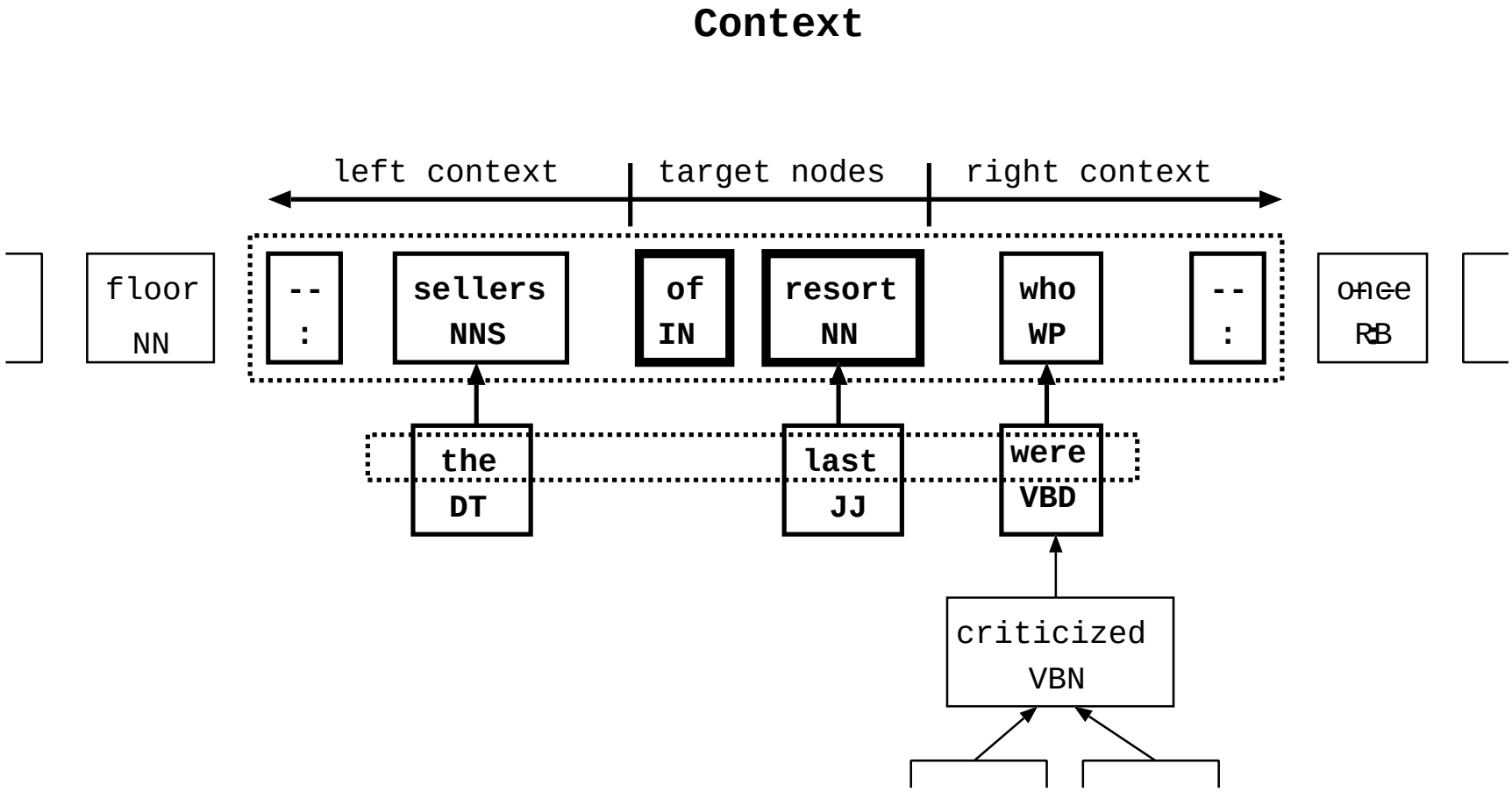
	forward	backward
Dep. Acc.	0.887	0.900
Root Acc.	0.835	0.896
Comp. Rate	0.361	0.379

Effect of Child Features (1/4)



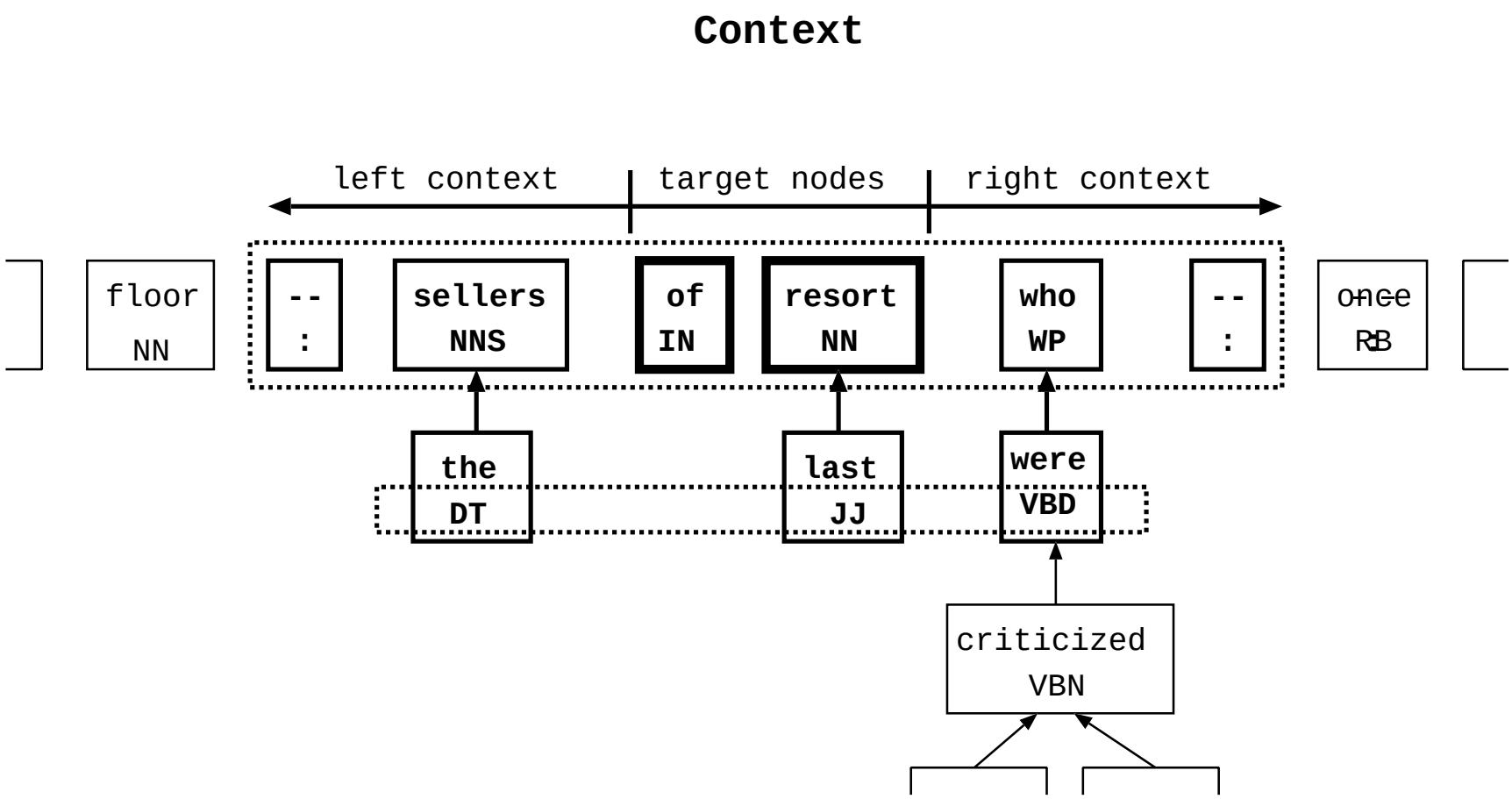
No use of child features

Effect of Child Features (2/4)



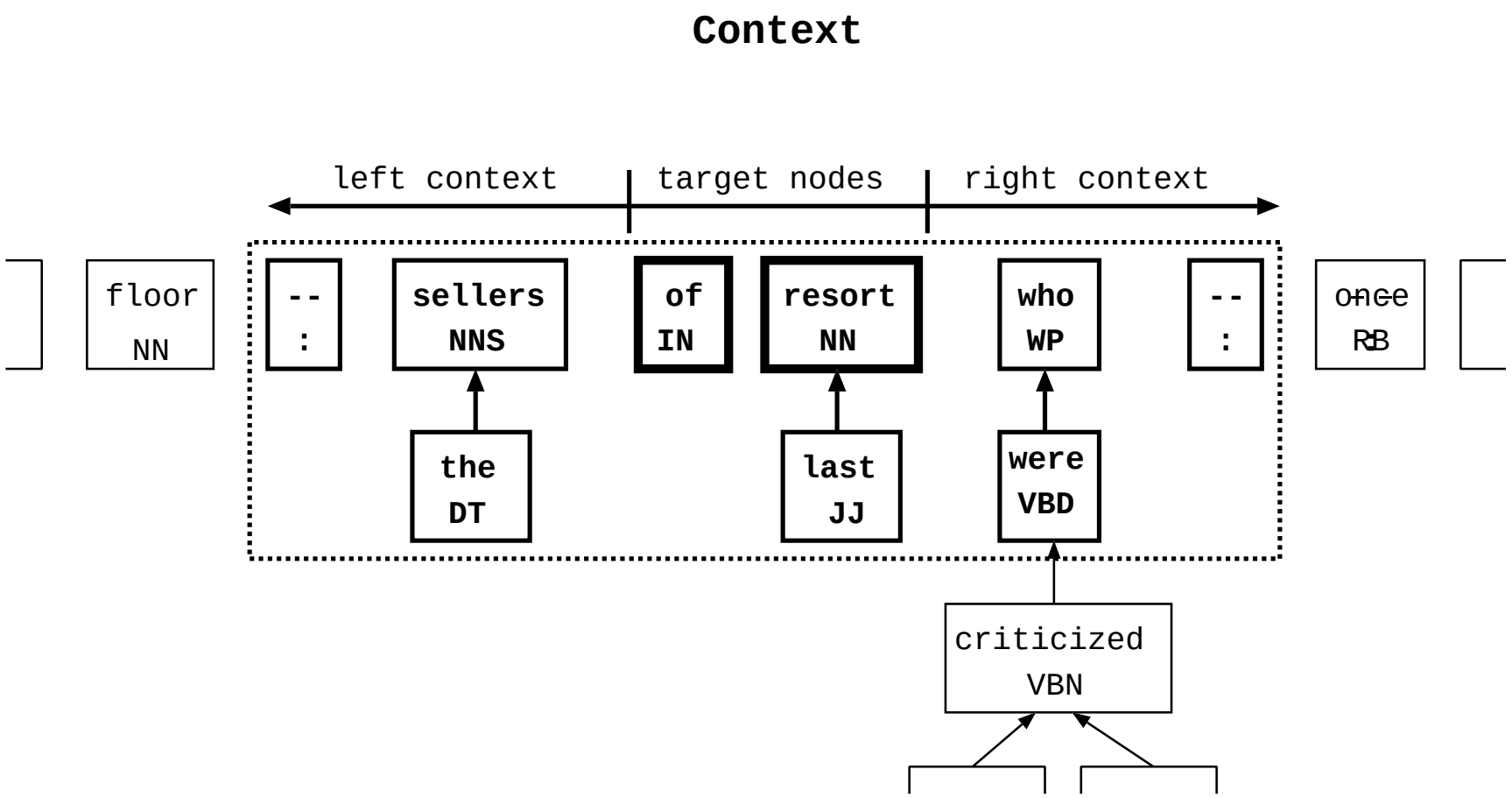
Use of only word of child node.

Effect of Child Features (3/4)



Use of only POS tags of child node.

Effect of Child Features (4/4)



Use of POS tags and word of child node.

Effect of Child Features (Summary)

context length: (2, 4), kernel function: $(\mathbf{x}' \cdot \mathbf{x}'' + 1)^2$

	no children	only word	only POS	all
Dep. Acc.	0.890	0.901	0.902	**0.903**
Root Acc.	0.882	0.915	0.912	**0.916**
Comp. Rate	0.348	0.383	0.374	**0.384**

Dynamic use of dependency relation is effective.

Comparison with Related Work

Charniak parser (Charniak 00) :

⇒ Based on probabilistic CFG and maximum entropy models.

(POS tagged simultaneously)

Collins parser (Collins 97) :

⇒ Based on three kinds of probabilistic generative models.

(POS tagged by Nakagawa's tagger)

⇓

The outputs are converted from phrase structure trees into dependency trees, then evaluated.

context length:(2, 4), kernel function: $(\mathbf{x}' \cdot \mathbf{x}'' + 1)^2$, all of features.

	Charniak	Collins' models			Our parser
		1	2	3	(no grouping)
Dep.	**0.921**	0.912	0.914	0.915	0.903 (0.910)
Root	**0.952**	0.950	0.951	**0.952**	0.915 (0.917)
Comp.	**0.452**	0.406	0.431	0.433	0.382 (0.422)
Leaf	**0.943**	0.936	0.936	0.937	0.937 (–)

Over 90% accuracy without phrase structure information

References

- Eugene Charniak, "A Maximum-Entropy-Inspired Parser," 1st Meeting of the North American Chapter of the ACL, pp.132-139, May 2000.
- Michael Collins, "Three Generative, Lexicalised Models for Statistical Parsing," 35th Annual Meeting of the ACL and 8th Conference of the European Chapter of the ACL, pp.16-23, July 1997.
- Tetsuji Nakagawa, Taku Kudoh, Yuji Matsumoto, "Revision Learning and its Application to Part-of-Speech Tagging," 40th Annual Meeting of the ACL, pp.497-504, July 2002.
- Hiroyasu Yamada, Yuji Matsumoto, "Statistical Dependency Analysis with Support Vector Machines," International Workshop on Parsing Technology, April 2003.