Machine Learning

Kernel Engineering

Alessandro Moschitti

Department of information and communication technology
University of Trento
Email: moschitti@dit.unitn.it

Kernel Function Definition

Def. 2.26 *A kernel is a function k, such that* $\forall\ \vec{x}, \vec{z} \in X$

$$k(\vec{x}, \vec{z}) = \phi(\vec{x}) \cdot \phi(\vec{z})$$

where ϕ is a mapping from X to an (inner product) feature space.

- Kernel function are mappings such as

$$\vec{x} \in \Re^n, \qquad \vec{\phi}(\vec{x}) = (\phi_1(\vec{x}), \phi_2(\vec{x}), \ldots, \phi_m(\vec{x})) \in \Re^m$$

Valid Kernels

Def. B.11 *Eigen Values*
Given a matrix $\boldsymbol{A} \in \mathbb{R}^m \times \mathbb{R}^n$, *an egeinvalue* λ *and an egeinvector* $\vec{x} \in \mathbb{R}^n - \{\vec{0}\}$ *are such that*

$$\boldsymbol{A}\vec{x} = \lambda\vec{x}$$

Def. B.12 *Symmetric Matrix*
A square matrix $\boldsymbol{A} \in \mathbb{R}^n \times \mathbb{R}^n$ *is symmetric iff* $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$ *for* $i \neq j$ $i = 1, .., m$ *and* $j = 1, .., n$, *i.e. iff* $\boldsymbol{A} = \boldsymbol{A}'$.

Def. B.13 *Positive (Semi-) definite Matrix*
A square matrix $\boldsymbol{A} \in \mathbb{R}^n \times \mathbb{R}^n$ *is said to be positive (semi-) definite if its eigenvalues are all positive (non-negative).*

Valid Kernels cont'd

Proposition 2.27 *(Mercer's conditions)*
Let X be a finite input space with $K(\vec{x}, \vec{z})$ a symmetric function on X. Then $K(\vec{x}, \vec{z})$ is a kernel function if and only if the matrix

$$k(\vec{x}, \vec{z}) = \phi(\vec{x}) \cdot \phi(\vec{z})$$

is positive semi-definite (has non-negative eigenvalues).

- If the matrix is positive semi-definite then we can find a mapping ϕ implementing the kernel function

Valid Kernel operations

- $k(x,z) = k_1(x,z)+k_2(x,z)$
- $k(x,z) = k_1(x,z)*k_2(x,z)$
- $k(x,z) = \alpha\, k_1(x,z)$
- $k(x,z) = f(x)f(z)$
- $k(x,z) = k_1(\phi(x),\phi(z))$
- $k(x,z) = x'Bz$

Basic Kernels for unstructured data

- Linear Kernel
  - Your features
- Polynomial Kernel
  - Feature conjunctions
- Lexical kernel
  - similarity between your features

Kernels for structured data

- String Kernel
  - Character sequences
  - Word sequences
  - Spectrum kernel
- Tree kernels
  - Subtree,
  - Subset Tree,
  - Partial Tree kernels
    - Applies properties of string kernels

Kernel Engineering for Language Applications

- Basic Combinations
- *Canonical Mappings*, e.g. object transformations
- *Merging of Kernels*

Kernel Combinations an example

K_p^3 polynomial kernel of flat features

K_{Tree} Tree kernel

- Kernel Combinations:

$$K_{Tree+P} = \gamma \times K_{Tree} + K_p^3, \qquad K_{Tree \times P} = K_{Tree} \times K_p^3$$

$$K_{Tree+P} = \gamma \times \frac{K_{Tree}}{\left|K_{Tree}\right|} + \frac{K_p^3}{\left|K_p^3\right|}, \qquad K_{Tree \times P} = \frac{K_{Tree} \times K_p^3}{\left|K_{Tree}\right| \times \left|K_p^3\right|}$$

Object Transformation [CLJ 2008]

- $$K(O_1,O_2)=\phi(O_1)\cdot\phi(O_2)=\phi_E(\phi_M(O_1))\cdot\phi_E(\phi_M(O_2))$$
  $$=\phi_E(S_1)\cdot\phi_E(S_2)=K_E(S_1,S_2)$$
- ***Canonical Mapping***, $\phi_M()$
  - object transformation,
  - e. g. a syntactic parse tree, into a verb subcategorization frame tree.
- ***Feature Extraction***, $\phi_E()$
  - maps the canonical structure in all its fragments
  - different fragment spaces, e. g. ST, SST and PT.

Predicate Argument Classification

- In an event:
  - target words describe relation among different entities
  - the participants are often seen as predicate's arguments.
- Example:

  Paul gives a talk in Rome

Predicate Argument Classification

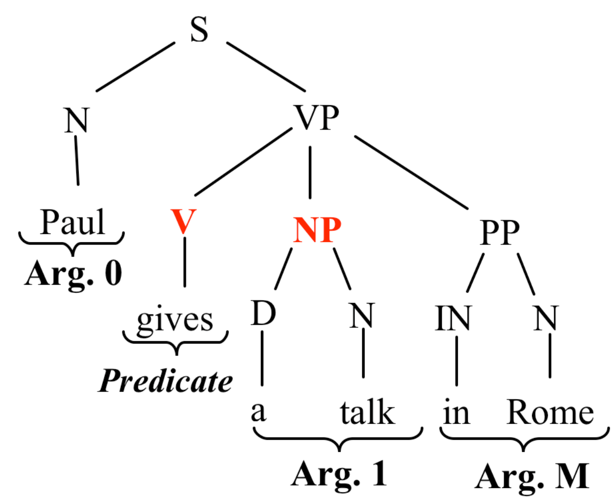
- In an event:
  - target words describe relation among different entities
  - the participants are often seen as predicate's arguments.
- Example:

  [$_{Arg0}$ Paul] [$_{predicate}$ gives] [$_{Arg1}$ a talk] [$_{ArgM}$ in Rome]

Predicate-Argument Feature Representation

Given a sentence, a predicate *p*:

1. Derive the sentence parse tree
2. For each node pair <N_p,N_x>
   a. Extract a feature representation set *F*
   b. If N_x exactly covers the Arg-*i*, *F* is one of its positive examples
   c. *F* is a negative example otherwise



Vector Representation for the linear kernel

Phrase Type

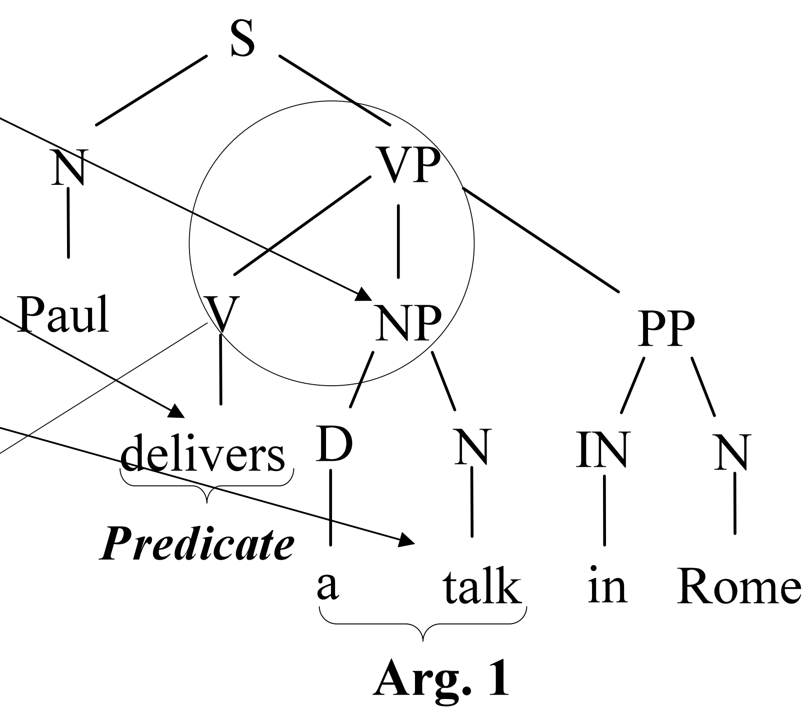
Predicate Word

Head Word

Parse Tree Path

Position Right

Voice Active

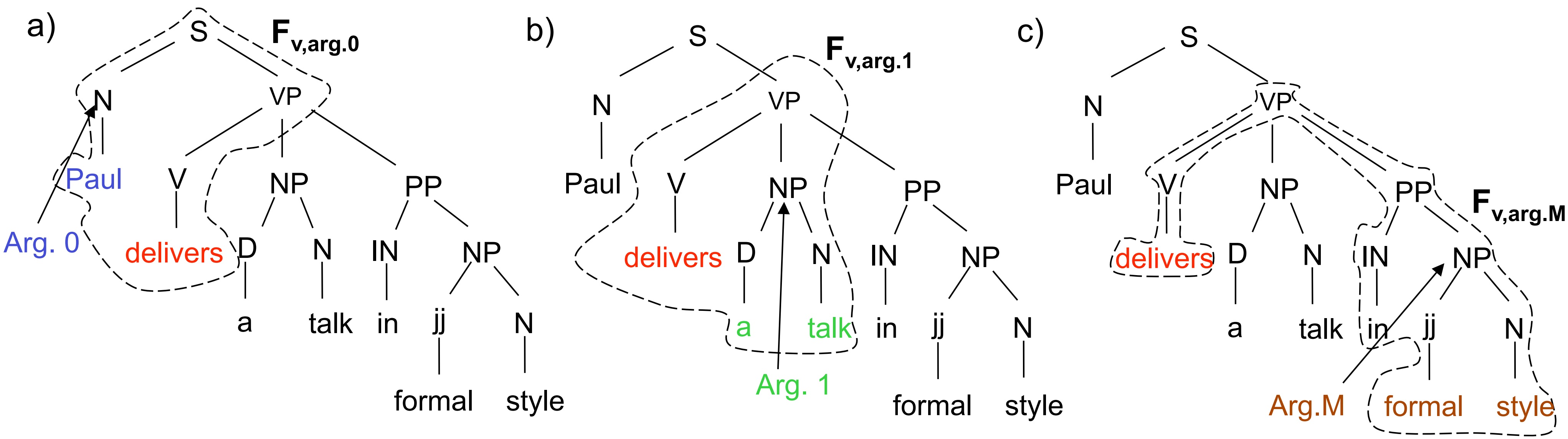


Kernel Engineering: Tree Tailoring

PAT Kernel [Moschitti, 2004]

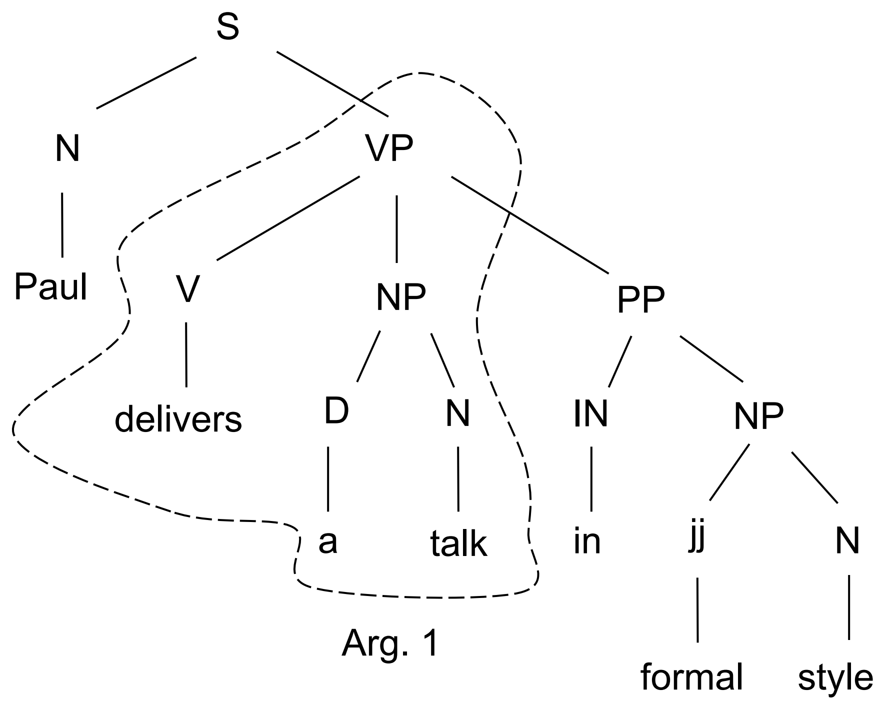
- Given the sentence:

[$_{Arg0}$ Paul] [$_{predicate}$ delivers] [$_{Arg1}$ a talk] [$_{ArgM}$ in formal Style]

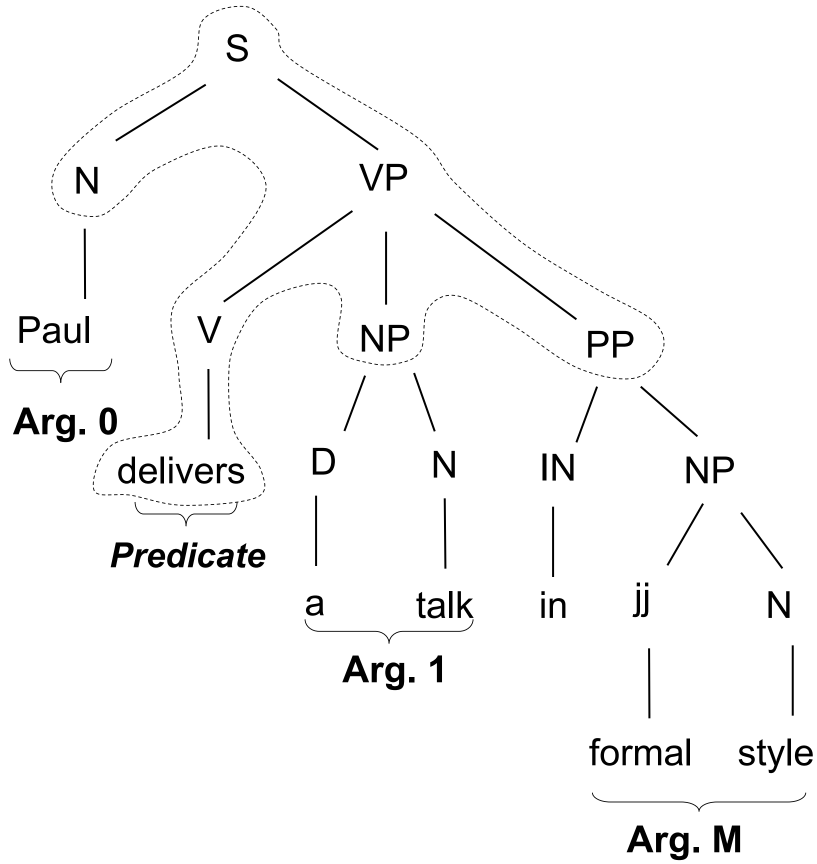


- These are Semantic Structures

In other words we consider…



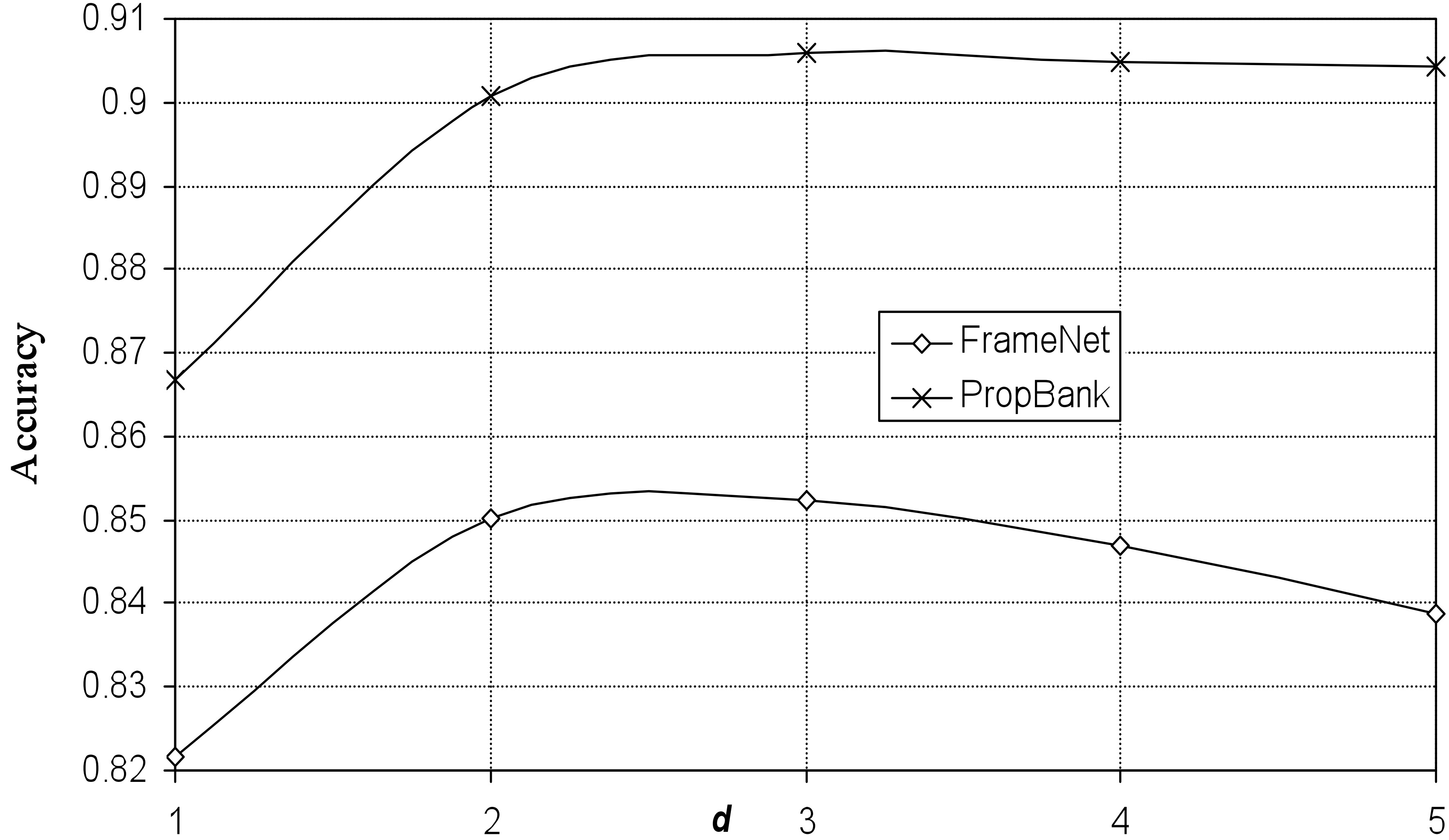
Sub-Categorization Kernel (SCF) [Moschitti, 2004]



Experiments on Gold Standard Trees

- PropBank and PennTree bank
  - about 53,700 sentences
  - Sections from 2 to 21 train., 23 test., 1 and 22 dev.
  - Arguments from Arg0 to Arg5, ArgA and ArgM for a total of 122,774 and 7,359
- FrameNet and Collins' automatic trees
  - 24,558 sentences from the 40 frames of Senseval 3
  - 18 roles (same names are mapped together)
  - Only verbs
  - 70% for training and 30% for testing

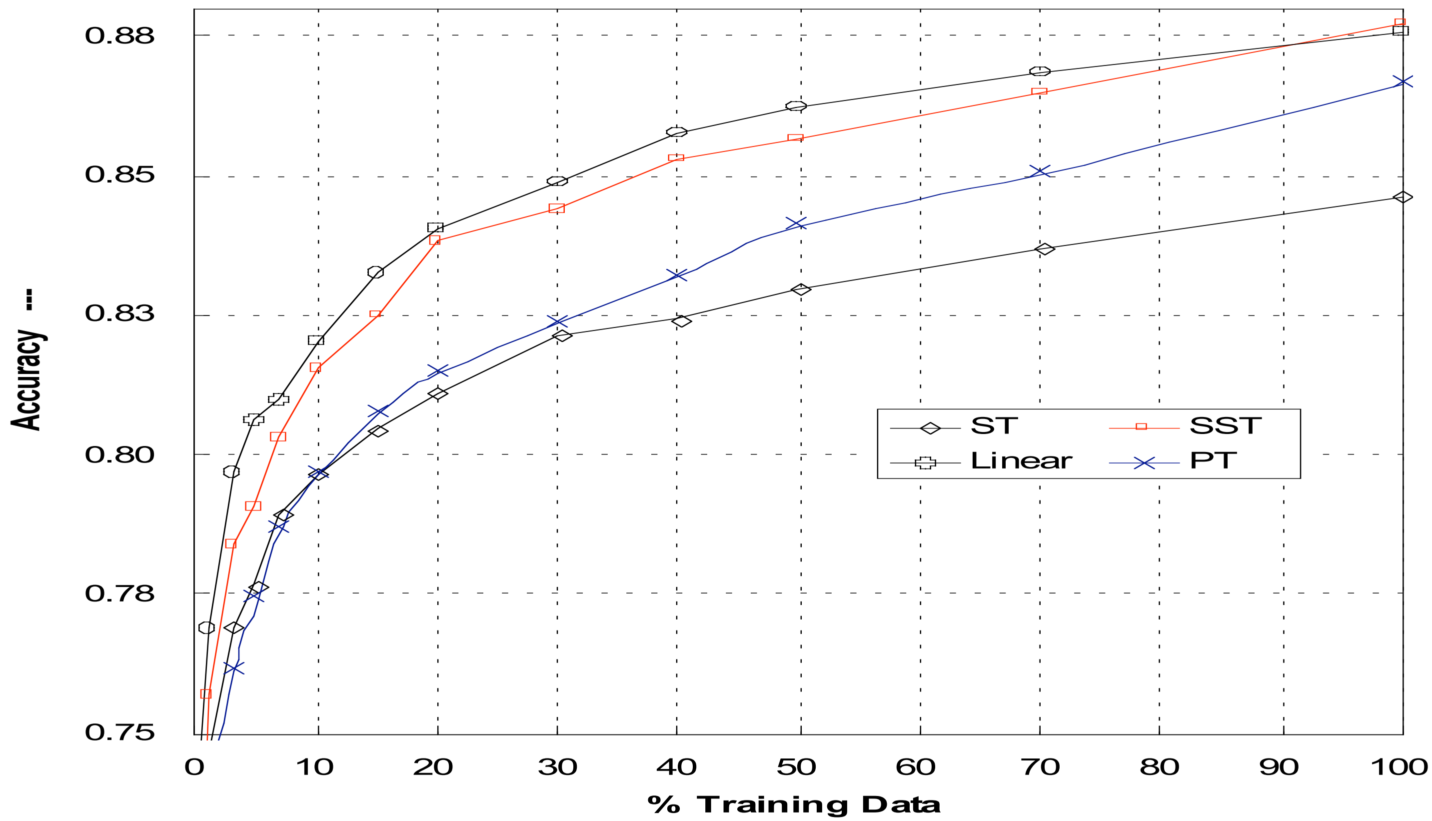
Argument Classification with Poly Kernel



PropBank Results

Args	P3	PAT	PAT+P	PAT×P	SCF+P	SCF×P
Arg0	90.8	88.3	92.6	90.5	94.6	94.7
Arg1	91.1	87.4	91.9	91.2	92.9	94.1
Arg2	80.0	68.5	77.5	74.7	77.4	82.0
Arg3	57.9	56.5	55.6	49.7	56.2	56.4
Arg4	70.5	68.7	71.2	62.7	69.6	71.1
ArgM	95.4	94.1	96.2	96.2	96.1	96.3
Global Accuracy	**90.5**	**88.7**	**91.3**	**90.4**	**92.4**	**93.2**

Argument Classification on PAT using different Tree Fragment Extractor [Moschitti, ECML 2006]



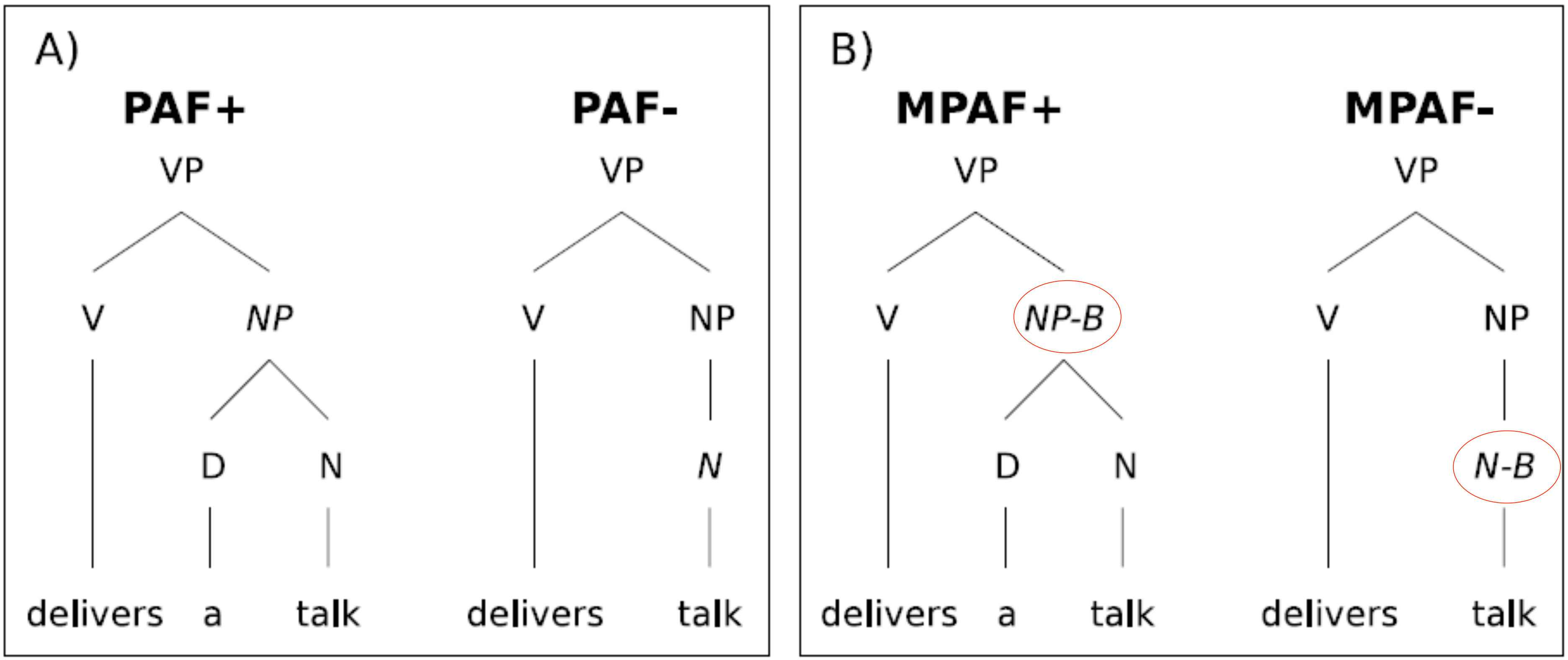
FrameNet Results

Roles	P3	PAF	PAF+P	PAF×P	SCF+P	SCF×P
agent	92.0	88.5	91.7	91.3	93.1	93.9
cause	59.7	16.1	41.6	27.7	42.6	57.3
degree	74.9	68.6	71.4	57.8	68.5	60.9
depictive	52.6	29.7	51.0	28.6	46.8	37.6
duration	45.8	52.1	40.9	29.0	31.8	41.8
goal	85.9	78.6	85.3	82.8	84.0	85.3
instrument	67.9	46.8	62.8	55.8	59.6	64.1
manner	81.0	81.9	81.2	78.6	77.8	77.8
Global Acc. (18 roles)	85.2	79.5	84.6	81.6	83.8	84.2

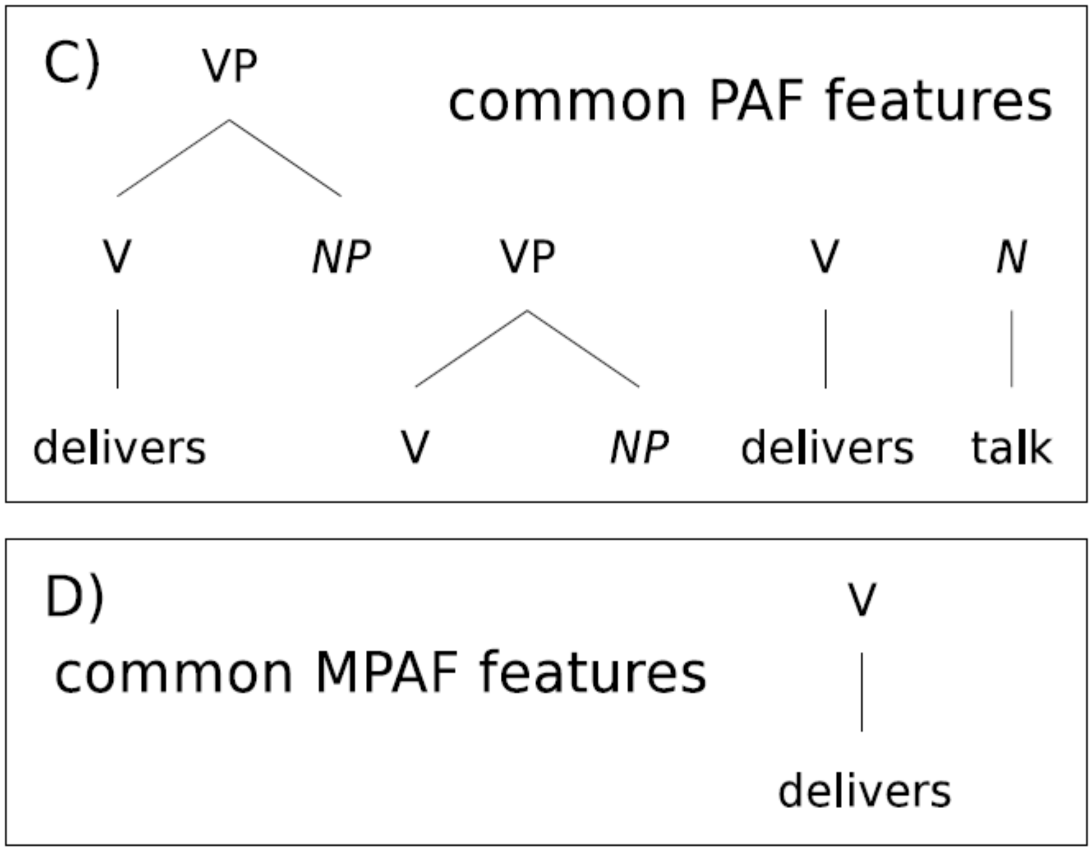
- ProbBank arguments vs. Semantic Roles

Kernel Engineering: Node marking [Moschitti et al, CLJ 2008]

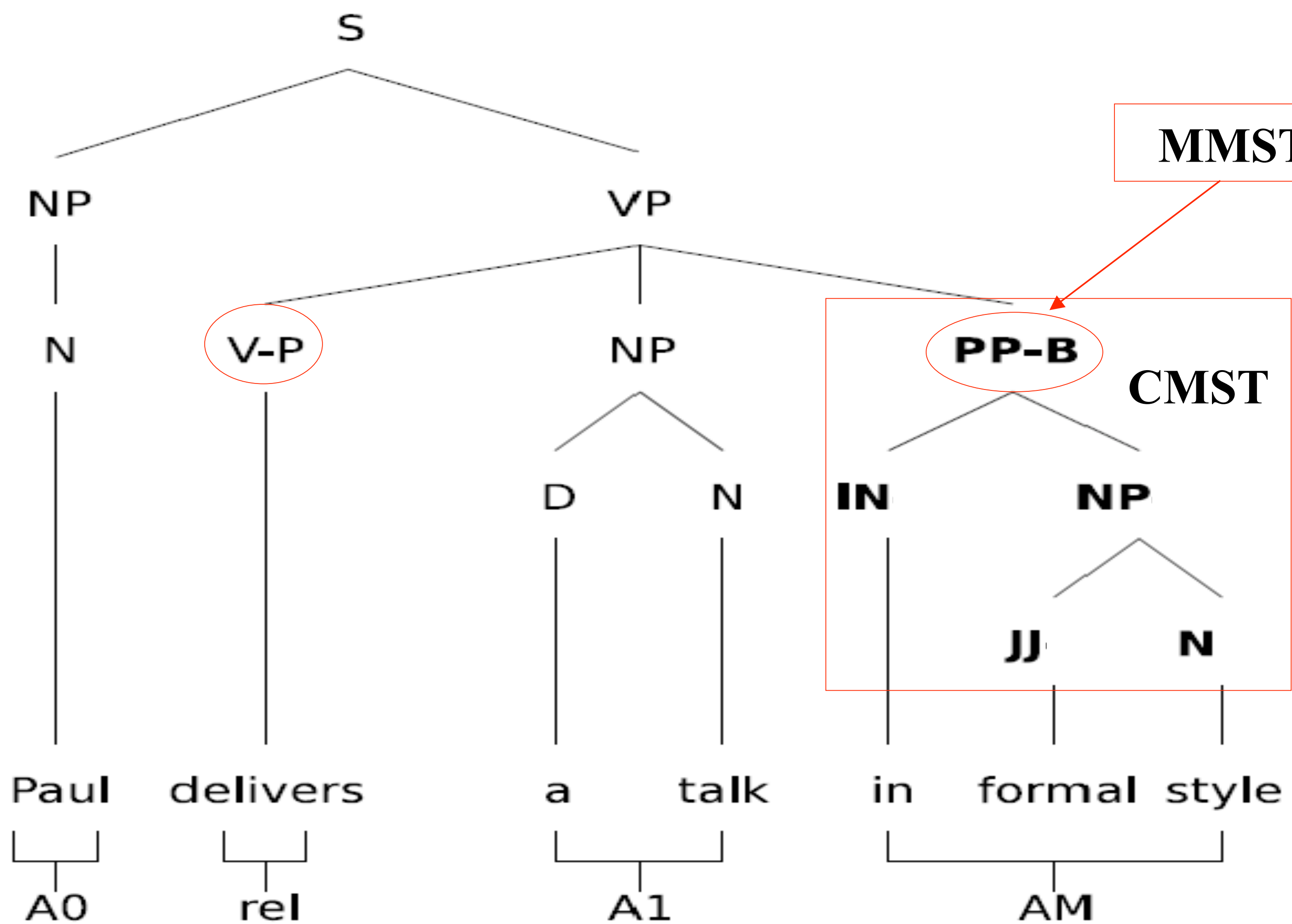
Marking Boundary nodes



Node Marking Effect



Different tailoring and marking



Experiments

- PropBank and PennTree bank
  - about 53,700 sentences
  - Charniak trees from CoNLL 2005
- Boundary detection:
  - Section 2 training
  - Section 24 testing
  - PAF and MPAF

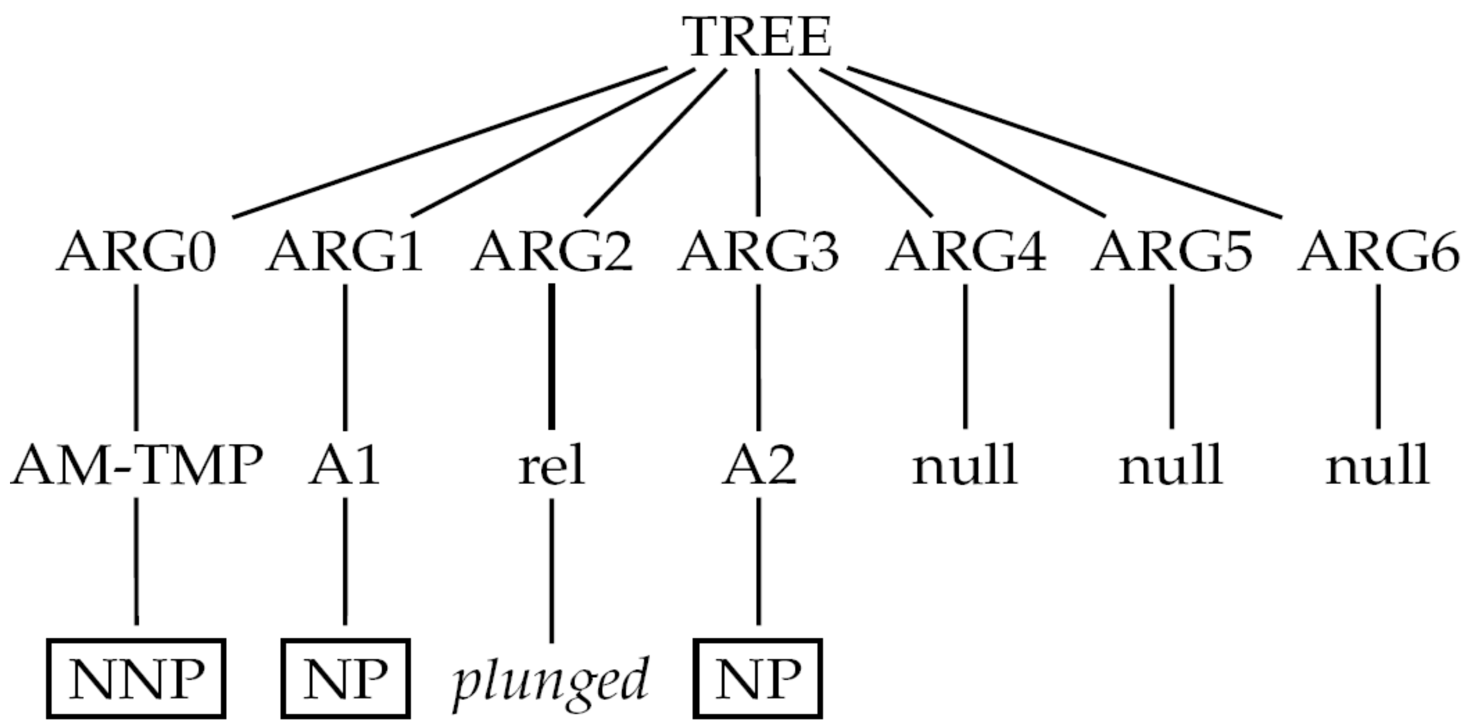
Number of examples/nodes of Section 2

	Section 2			Section 24		
Nodes	pos	neg	tot	pos	neg	tot
Internal	11,847	71,126	82,973	7,525	50,123	57,648
Pre-terminal	894	114,052	114,946	709	80,366	81,075
Both	12,741	185,178	197,919	8,234	130,489	138,723

Predicate Argument Feature (PAF) vs. Marked PAF (MPAF) [ACL-ws-2005]

Tagging strategy	CPU_{time}	F1
PAF	5,179.18	75.24
MPAF	3,131.56	82.07

Other Canonical mappings: Semantic structures for re-ranking [Moschitti et al., CoNLL06]



Question Classification

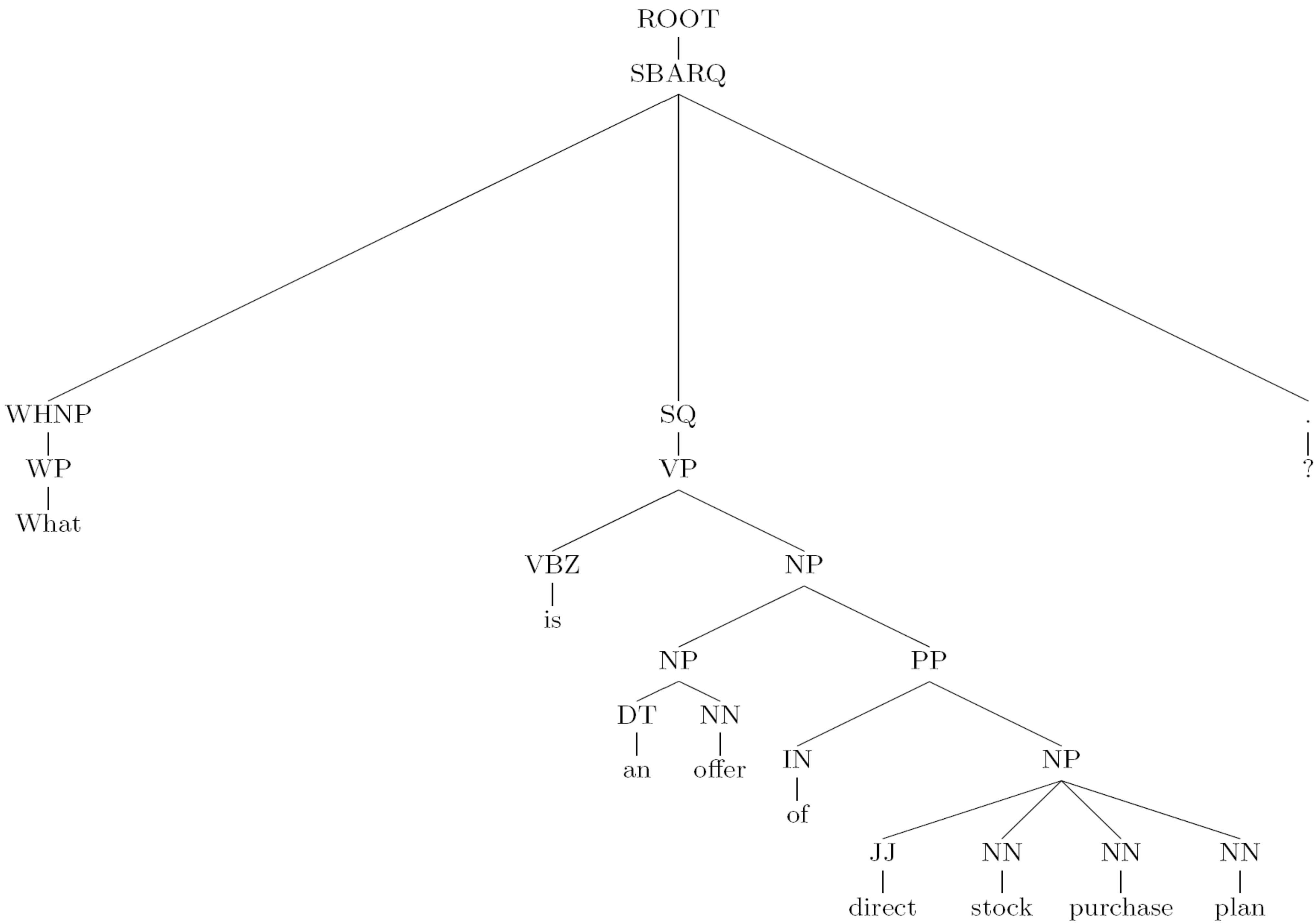
Question Taxonomy

- **Definition**: What does HTML stand for?
- **Description**: What's the final line in the Edgar Allan Poe poem "The Raven"?
- **Entity**: What foods can cause allergic reaction in people?
- **Human**: Who won the Nobel Peace Prize in 1992?
- **Location**: Where is the Statue of Liberty?
- **Manner**: How did Bob Marley die?
- **Numeric**: When was Martin Luther King Jr. born?
- **Organization**: What company makes Bentley cars?

Question Classifier based on Tree Kernels

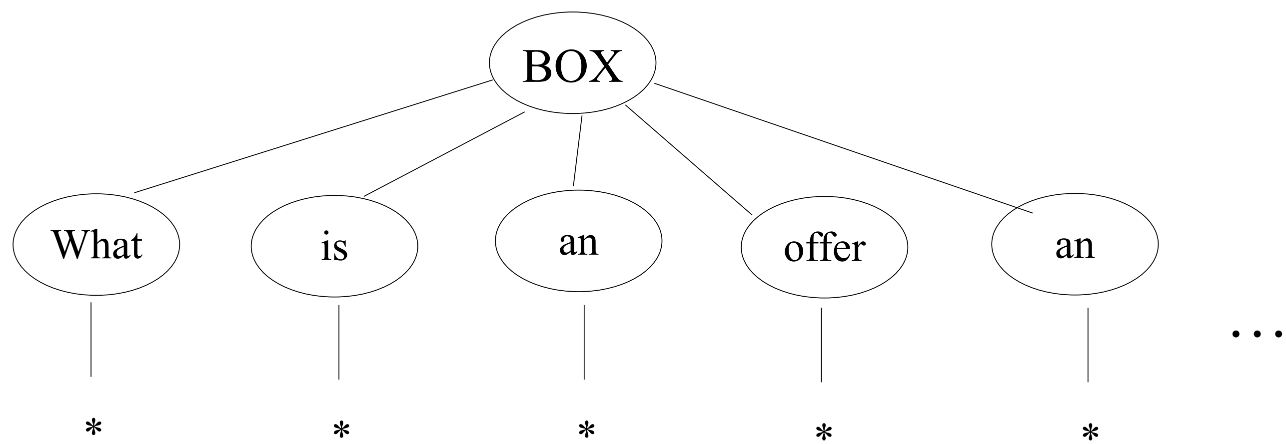
- Question dataset (http://l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/) [Lin and Roth, 2005])
  - Distributed on 6 categories: Abbreviations, Descriptions, Entity, Human, Location, and Numeric.
- Fixed split 5500 training and 500 test questions
- Cross-validation (10-folds)
- Using the whole question parse trees
  - Constituent parsing
  - Example

“**What is an offer of direct stock purchase plan ?**”


ROOT
SBARQ
WHNP
WP
What
SQ
VP
VBZ
is
NP
NP
DT
an
NN
offer
PP
IN
of
NP
JJ
direct
NN
stock
NN
purchase
NN
plan
.
?

Kernels

- BOW, POS are obtained with a simple tree, e.g.



- PT (parse tree)
- PAS (predicate argument structure)

Question classification

Features	Accuracy (UIUC)	Accuracy (c.v.)
PT	90.4	84.8±1.4
BOW	90.6	84.7±1.4
PAS	34.2	43.0±2.2
POS	26.4	32.4±2.5
PT+BOW	**91.8**	**86.1±1.3**
PT+BOW+POS	91.8	84.7±1.7
PAS+BOW	90.0	82.1±1.5
PAS+BOW+POS	88.8	81.0±1.7

Merging of Kernels

[Bloehdorn and Moschitti, ECIR and CIKM 2007]:

- Syntactic/Semantic Tree Kernel
- Kernel Combinations
- Experiments

Efficient evaluation of the scalar product

$$\vec{x} \cdot \vec{z} = \phi(T_x) \cdot \phi(T_z) = K(T_x, T_z) =$$

$$= \sum_{n_x \in T_x} \sum_{n_z \in T_z} \Delta(n_x, n_z)$$

Efficient evaluation of the scalar product

$$\vec{x} \cdot \vec{z} = \phi(T_x) \cdot \phi(T_z) = K(T_x, T_z) =$$

$$= \sum_{n_x \in T_x} \sum_{n_z \in T_z} \Delta(n_x, n_z)$$

- [Collins and Duffy, ACL 2002] evaluate Δ in $O(n^2)$:

$$\Delta(n_x, n_z) = 0, \ \text{ if the productions are different else}$$

$$\Delta(n_x, n_z) = 1, \ \text{ if pre - terminals else}$$

$$\Delta(n_x, n_z) = \prod_{j=1}^{nc(n_x)} (1 + \Delta(ch(n_x, j), ch(n_z, j)))$$

Merging of Kernels

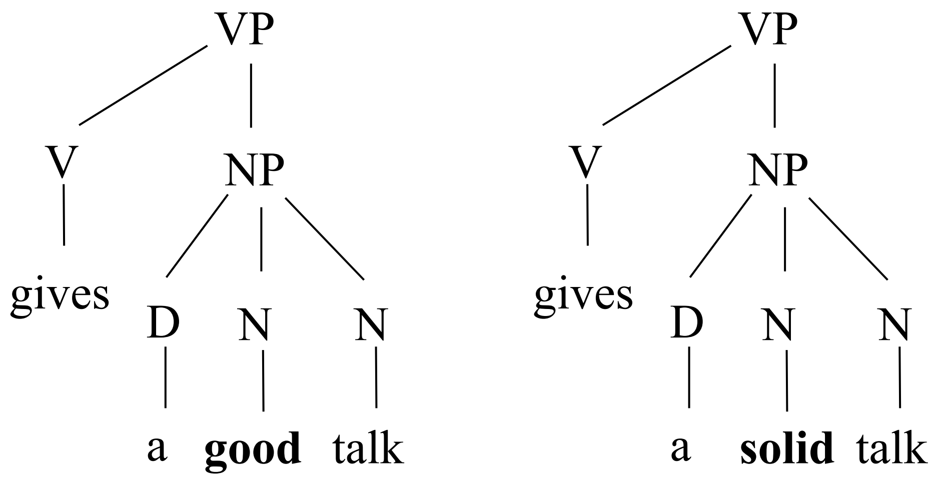
Definition 4 (Tree Fragment Similarity Kernel). *For two tree fragments $f_1, f_2 \in \mathcal{F}$, we define the Tree Fragment Similarity Kernel as*[4]*:*

$$\kappa_{\mathcal{F}}(f_1, f_2) = comp(f_1, f_2) \prod_{t=1}^{nt(f_1)} \kappa_S(f_1(t), f_2(t))$$

$$\kappa_T(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$$

where $\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} \sum_{j=1}^{|\mathcal{F}|} I_i(n_1) I_j(n_2) \kappa_{\mathcal{F}}(f_i, f_j)$.

Merging of Kernels



$$\kappa_T(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$$

$$where\ \Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} \sum_{j=1}^{|\mathcal{F}|} I_i(n_1) I_j(n_2) \kappa_{\mathcal{F}}(f_i, f_j).$$

Delta Evaluation is very simple

0. if n_1 and n_2 are pre-terminals and $label(n_1) = label(n_2)$ then $\Delta(n_1, n_2) = \lambda \kappa_{\mathcal{S}}(ch^1_{n_1}, ch^1_{n_2})$,

1. if the productions at n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$;

2. $\Delta(n_1, n_2) = \lambda$,

3. $\Delta(n_1, n_2) = \lambda \prod_{j=1}^{nc(n_1)} (1 + \Delta(ch^j_{n_1}, ch^j_{n_2}))$.

Similarity based on WordNet

Inverted Path Length:

$$sim_{IPL}(c_1, c_2) = \frac{1}{(1 + d(c_1, c_2))^{\alpha}}$$

Wu & Palmer:

$$sim_{WUP}(c_1, c_2) = \frac{2\, dep(lso(c_1, c_2))}{d(c_1, lso(c_1, c_2)) + d(c_2, lso(c_1, c_2)) + 2\, dep(lso(c_1, c_2))}$$

Resnik:

$$sim_{RES}(c_1, c_2) = -\log\ P(lso(c_1, c_2))$$

Lin:

$$sim_{LIN}(c_1, c_2) = \frac{2\ \log\ P(lso(c_1, c_2))}{\log\ P(c_1) + \log P\,(c_2)}$$

Question Classification with S/STK

	Accuracy				
λ parameter	**0.4**	**0.05**	**0.01**	**0.005**	**0.001**
linear (bow)	0.905				
string matching	0.890	0.910	**0.914**	**0.914**	0.912
full	0.904	**0.924**	0.918	0.922	0.920
full-ic	0.908	**0.922**	0.916	0.918	0.918
path-1	0.906	**0.918**	0.912	**0.918**	0.916
path-2	0.896	0.914	0.914	**0.916**	**0.916**
lin	0.908	**0.924**	0.918	0.922	0.922
wup	0.908	**0.926**	0.918	0.922	0.922

Practical Example

SVM-light-TK Software

- Encodes ST, SST and combination kernels in SVM-light [Joachims, 1999]
- Available at http://dit.unitn.it/~moschitt/
- Tree forests, vector sets
- New extensions: the PT kernel will be released asap

Data Format

- **"What does Html stand for?"**
- 1 |**BT**| (SBARQ (WHNP (WP What))(SQ (AUX does)(NP (NNP S.O.S.))(VP (VB stand)(PP (IN for))))(. ?))

|**BT**| (***BOW*** (What *)(does *)(S.O.S. *)(stand *)(for *)(? *))

|**BT**| (***BOP*** (WP *)(AUX *)(NNP *)(VB *)(IN *)(. *))

|**BT**| (***PAS*** (ARG0 (R-A1 (What *)))(ARG1 (A1 (S.O.S. NNP)))(ARG2 (rel stand)))

|**ET**| 1:1 21:2.742439465642236E-4 23:1 30:1 36:1 39:1 41:1 46:1 49:1 66:1 152:1 274:1 333:1

|BV| 2:1 21:1.4421347148614654E-4 23:1 31:1 36:1 39:1 41:1 46:1 49:1 52:1 66:1 152:1 246:1 333:1 392:1 **|EV|**

Basic Commands

- Training and classification
  - **./svm_learn -t 5 -C T train.dat model**
  - **./svm_classify test.dat model**
- Learning with a vector sequence
  - **./svm_learn -t 5 -C V train.dat model**
- Learning with the sum of vector and kernel sequences
  - **./svm_learn -t 5 -C + train.dat model**

Question and Answer Classification Canonical Mapping + Kernel Combinations

TASK: Automatic Classification
[Moschitti, CIKM 2008]

- The classifier detects if a pair (question and answer) is correct or not
- A representation for the pair is needed
- The classifier can be used to re-rank the output of a basic QA system

Dataset 2: TREC data

- 138 TREC 2001 test questions labeled as “description”
- 2,256 sentences, extracted from the best ranked paragraphs (using a basic QA system based on Lucene search engine on TREC dataset)
- 216 of which labeled as correct by one annotator

Dataset 2: TREC data

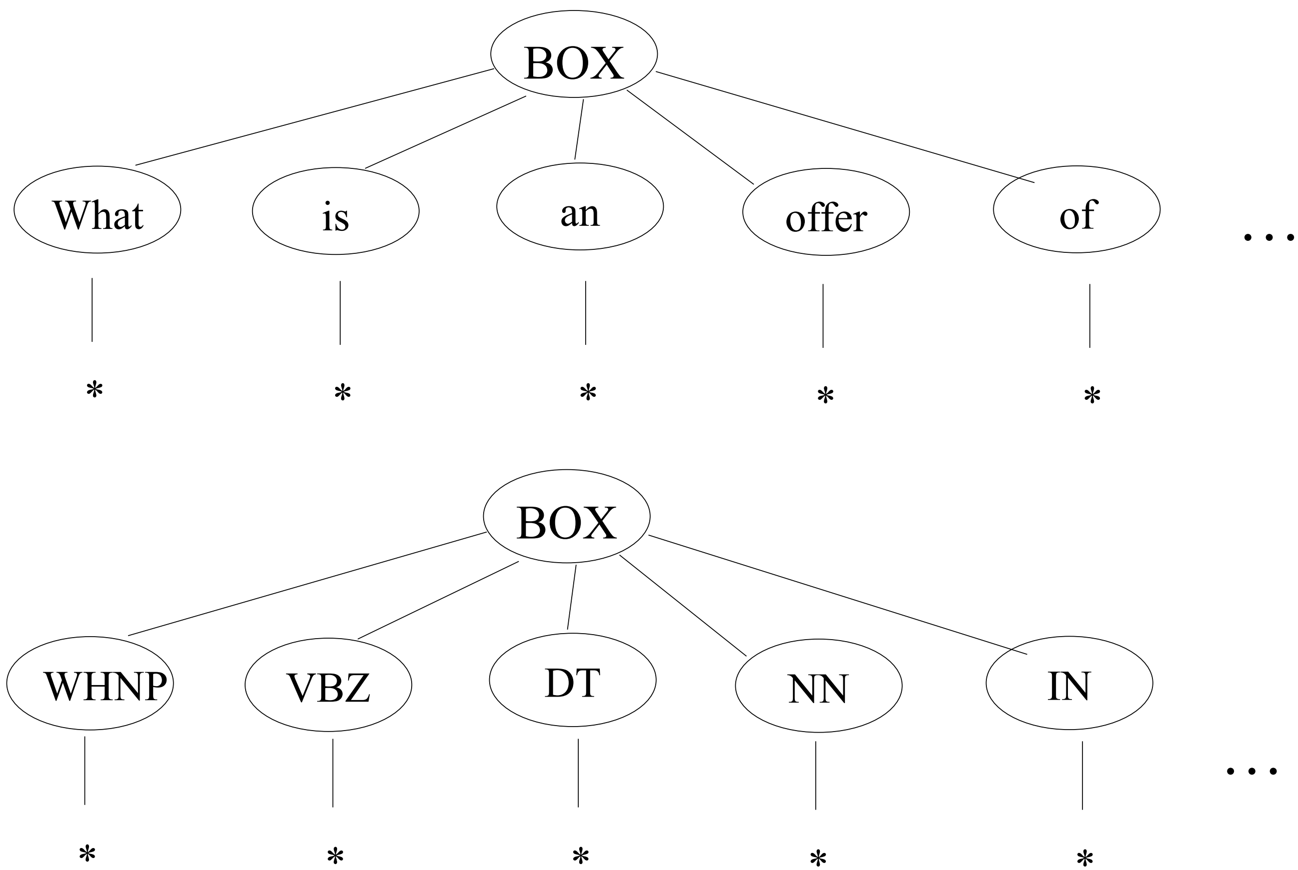
- 138 TREC 2001 test questions labeled as "description"

A question is linked to many answers: all its derived pairs cannot be shared by training and test sets

- 216 of which labeled as correct by one annotator

Bags of words (BOW) and POS-tags (POS)

- To save time, apply STK to these trees:

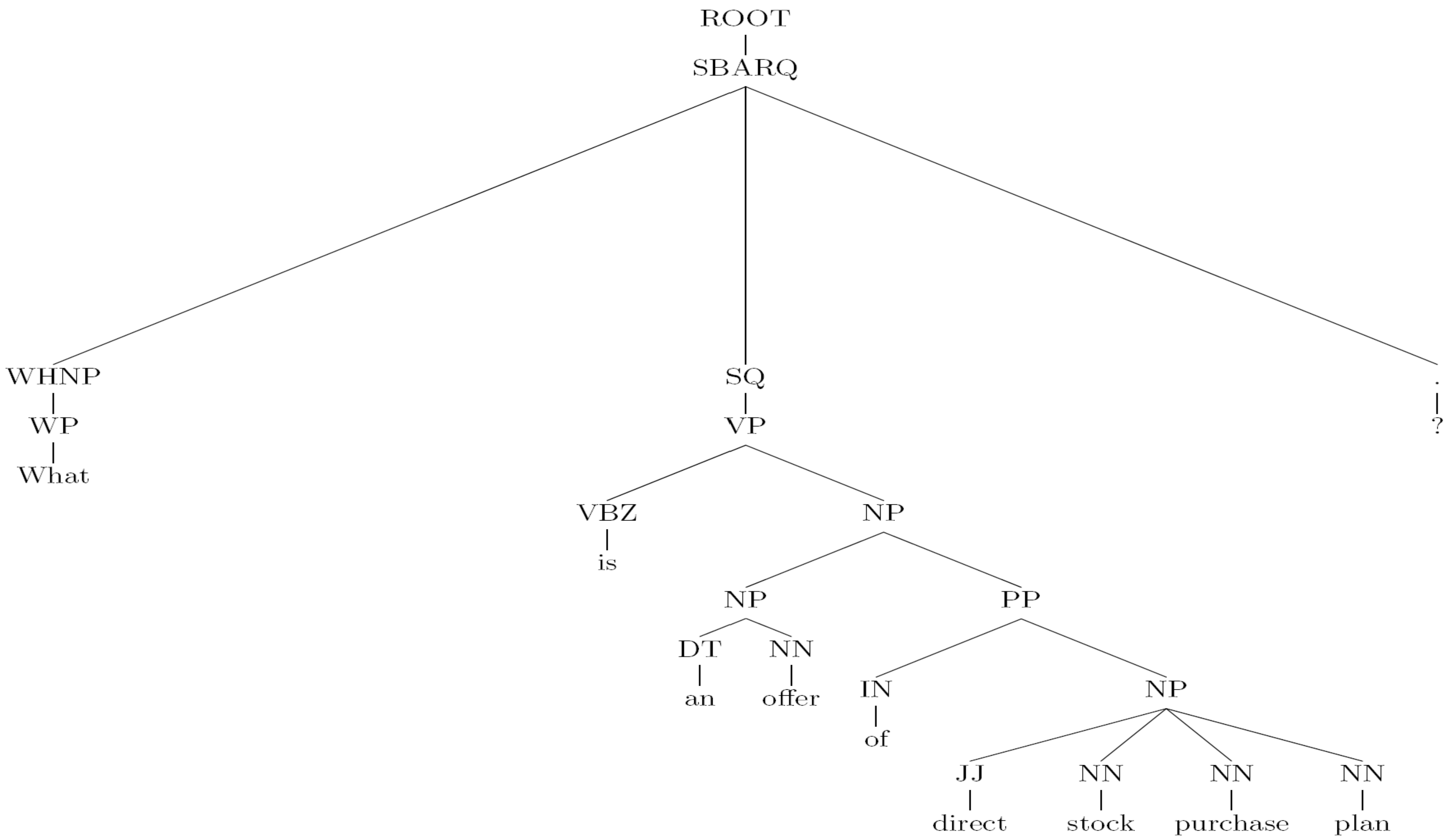


Word and POS Sequences

- What is an offer of…? (word sequence, **WSK**)
  - → `What_is_offer`
  - → `What_is`

- WHNP VBZ DT NN IN…(POS sequence, **POSSK**)
  - → `WHNP_VBZ_NN`
  - → `WHNP_NN_IN`

Syntactic Parse Trees (PT)

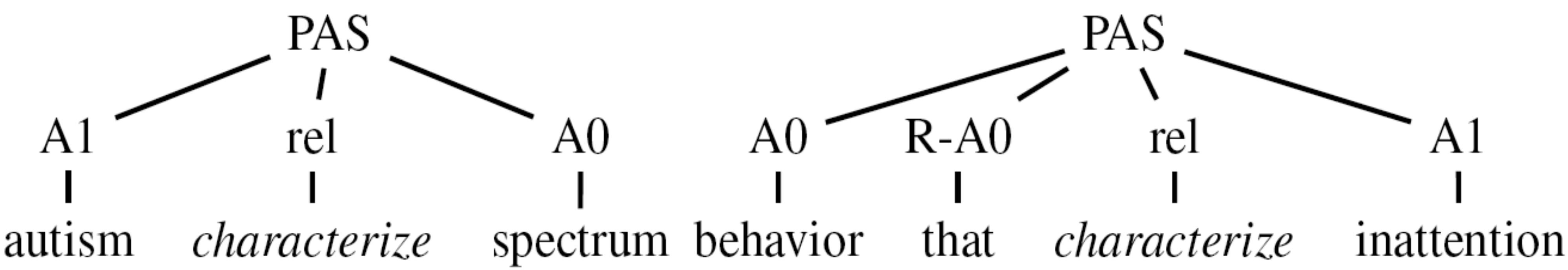


Predicate Argument Structure for Partial Tree Kernel (PAS$_{PTK}$)

s_1: *Autism is characterized by a broad spectrum of behavior that includes extreme inattention to surroundings and hypersensitivity to sound and other stimuli.*,

$[_{A1}$ *Autism*$]$ *is* $[_{rel}$ *characterized*$]$ $[_{A0}$ *by a broad spectrum of behavior*$]$ $[_{R-A0}$ *that*$]$ $[_{rel}$*includes*$]$ $[_{A1}$ *extreme inattention to surroundings and hypersensitivity to sound and other stimuli*$]$.

Predicate Argument Structure for Partial Tree Kernel (PAS$_{PTK}$)

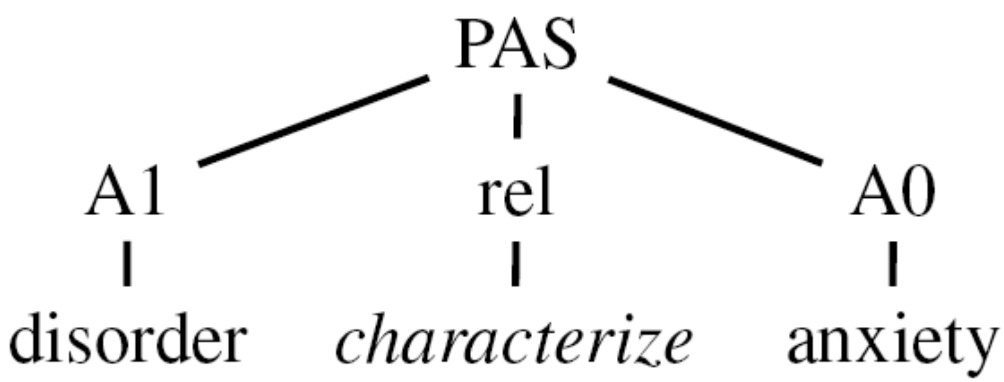


(a)

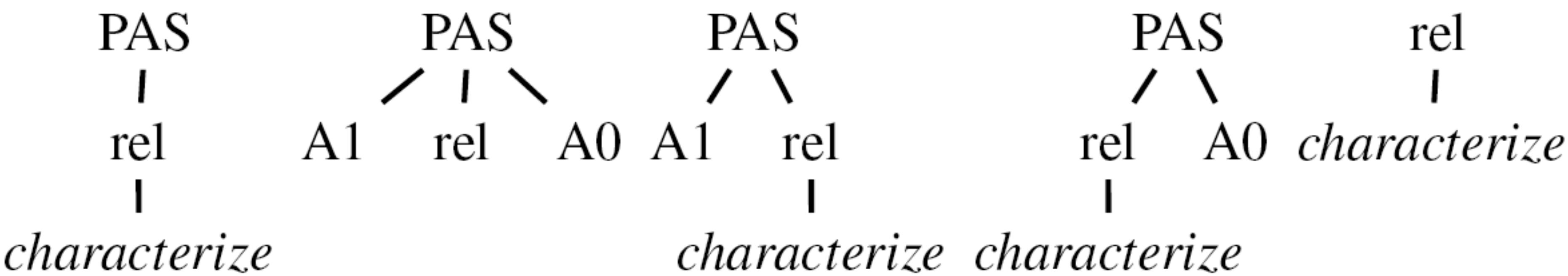
Predicate Argument Structure for Partial Tree Kernel (PAS$_{PTK}$)

s_2: *Panic disorder is characterized by unrealistic or excessive anxiety.*

[$_{A1}$ Panic disorder] is [$_{rel}$ characterized] [$_{A0}$ by unrealistic or excessive anxiety].

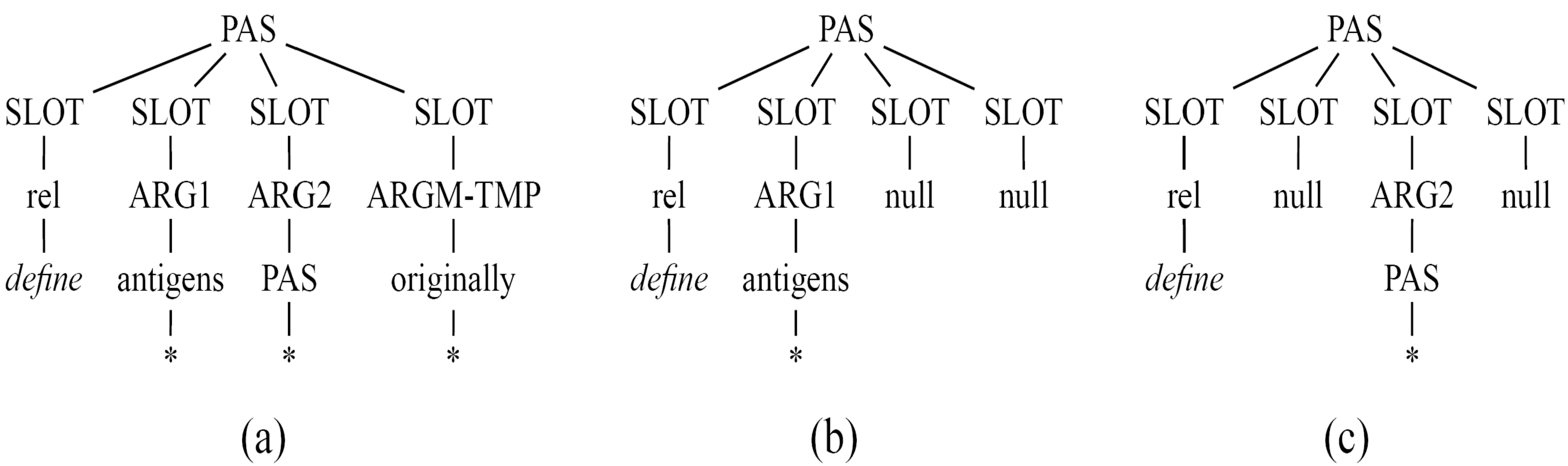


Common Substructures



Shallow Semantic Trees for SST kernel
[Moschitti et al, ACL 2007]

- To generate the substructures with STK we need to add slot nodes (from (a) obtain (b) and (C))

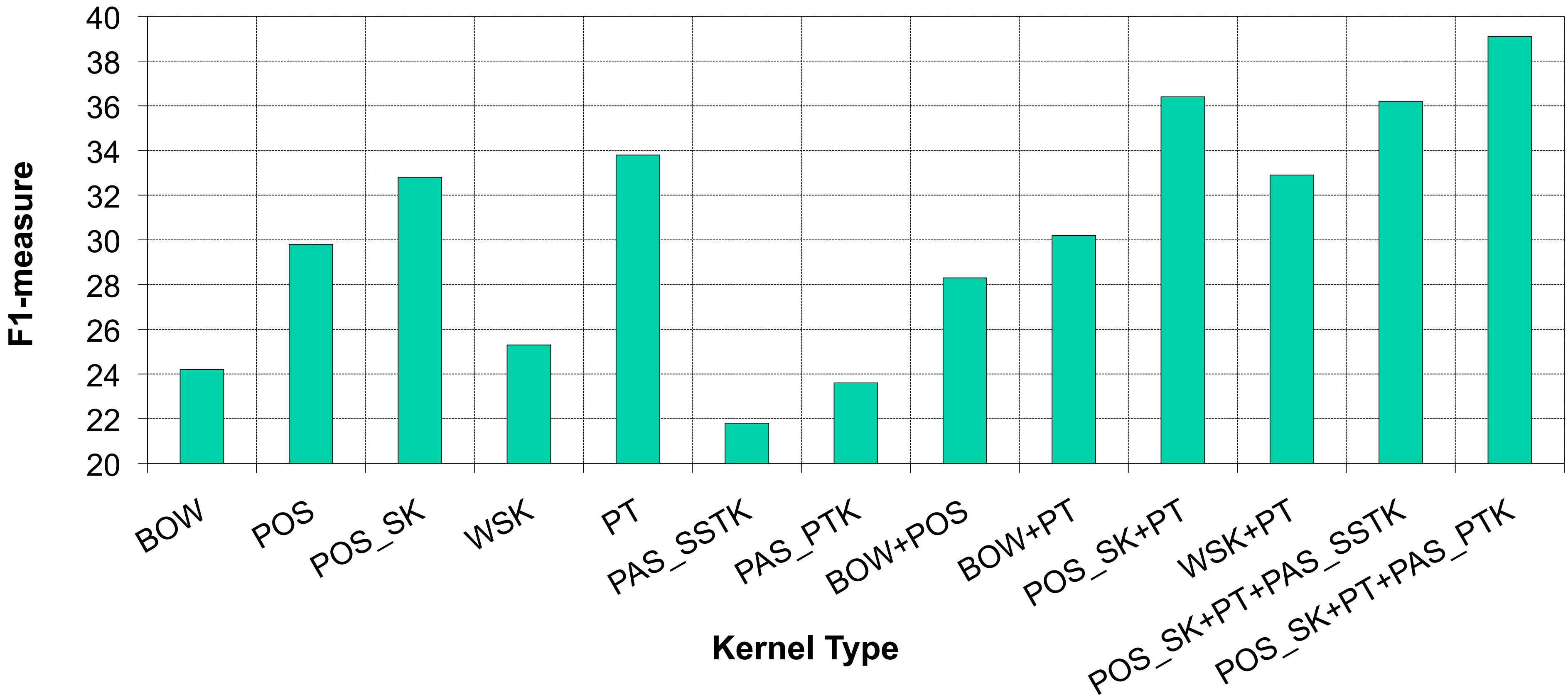


Kernels and Combinations

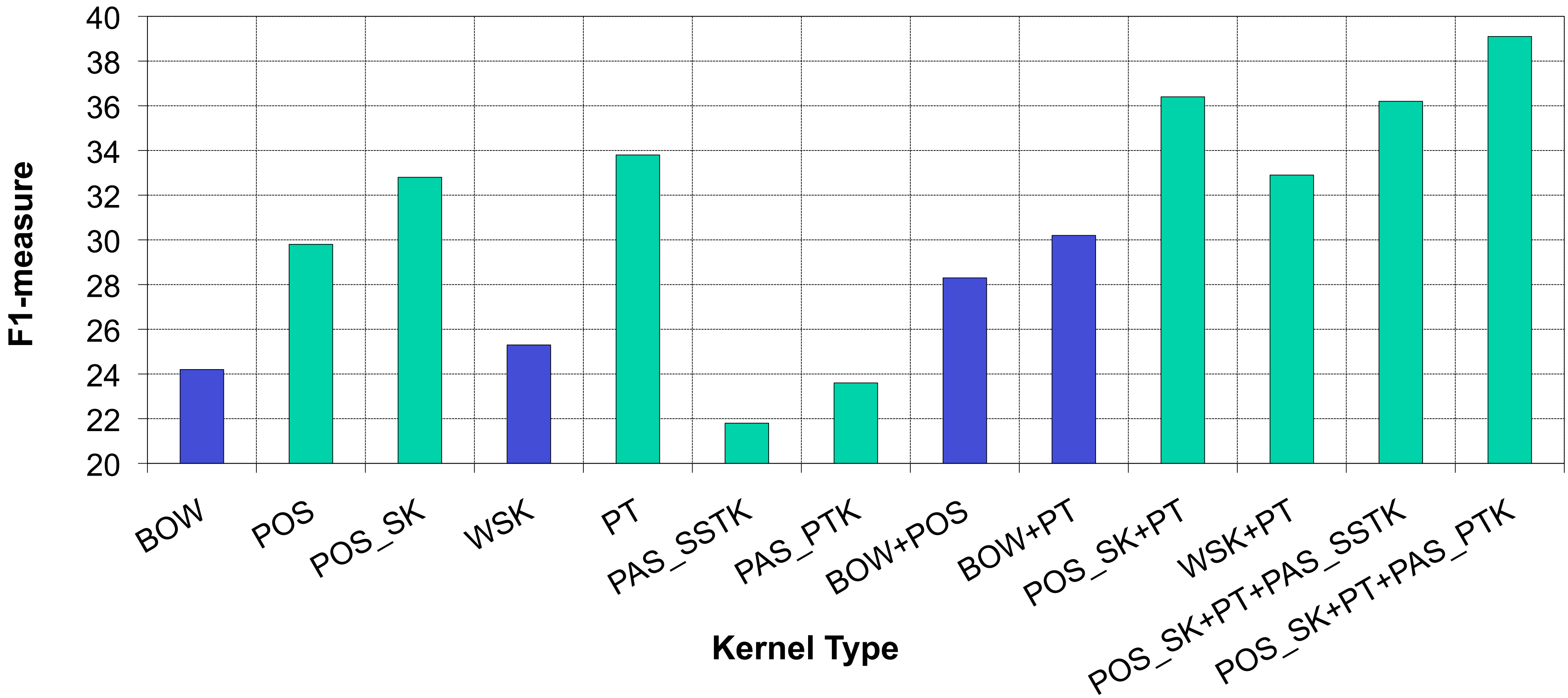
- Exploiting the property: $k(x,z) = k_1(x,z)+k_2(x,z)$
- BOW, POS, WSK, POSSK, PT, PAS_{PTK}

$\Rightarrow$ BOW+POS, BOW+PT, PT+POS, …

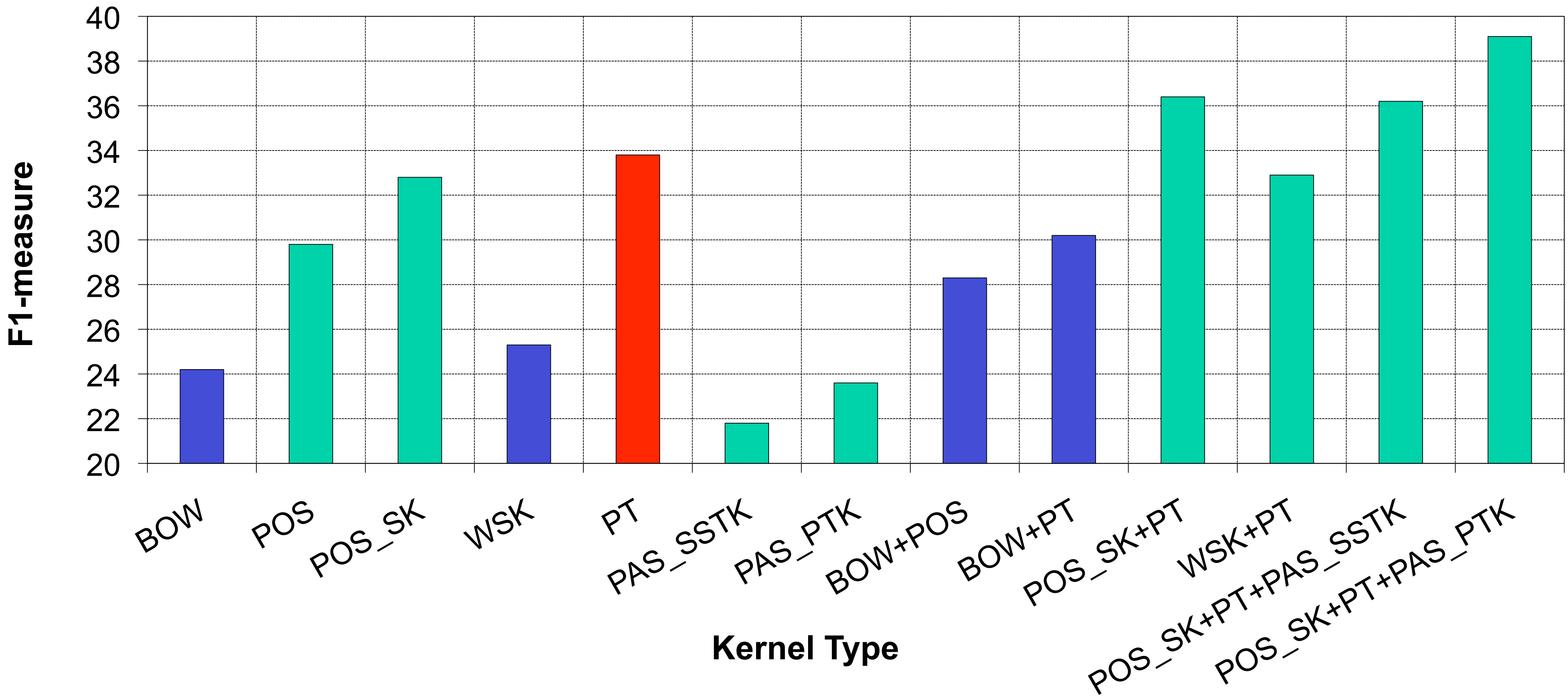
Results on TREC Data (5 folds cross validation)



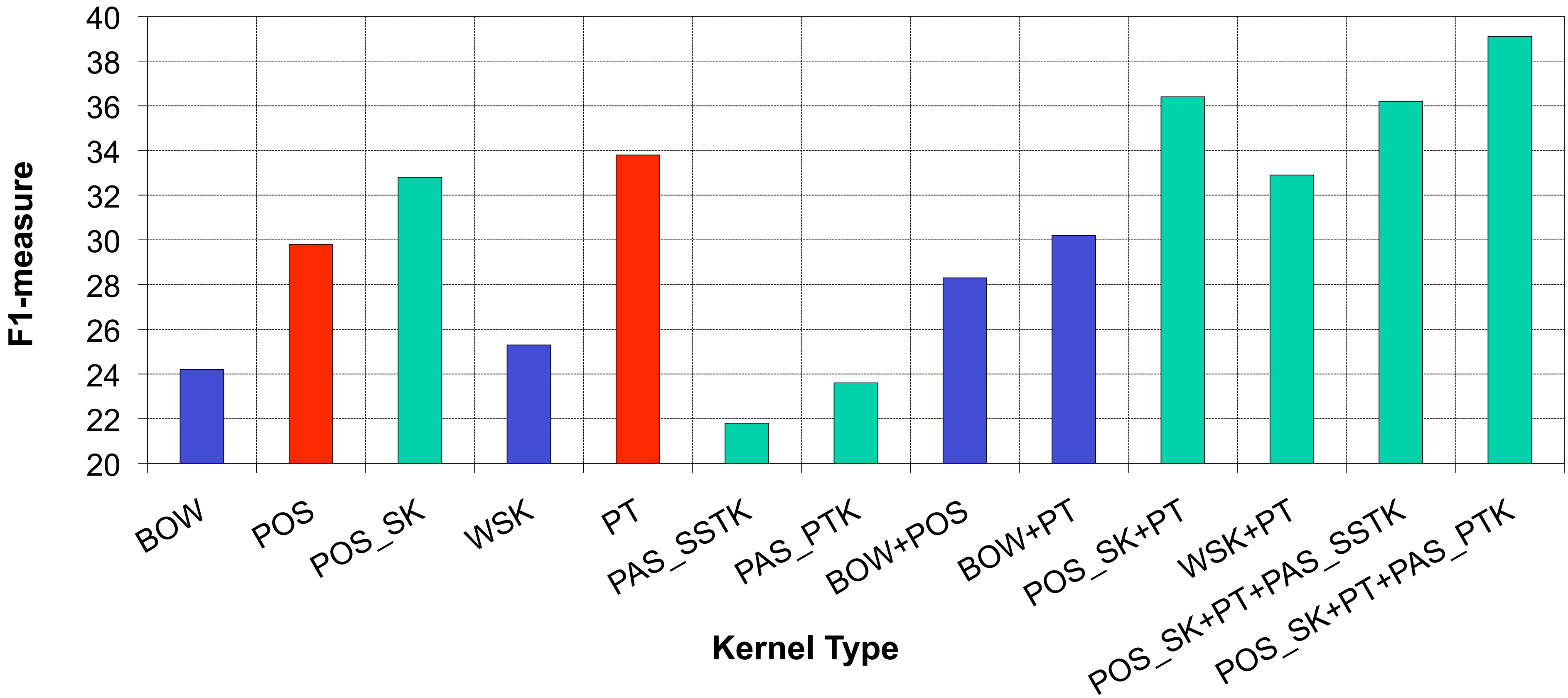
Results on TREC Data
(5 folds cross validation)



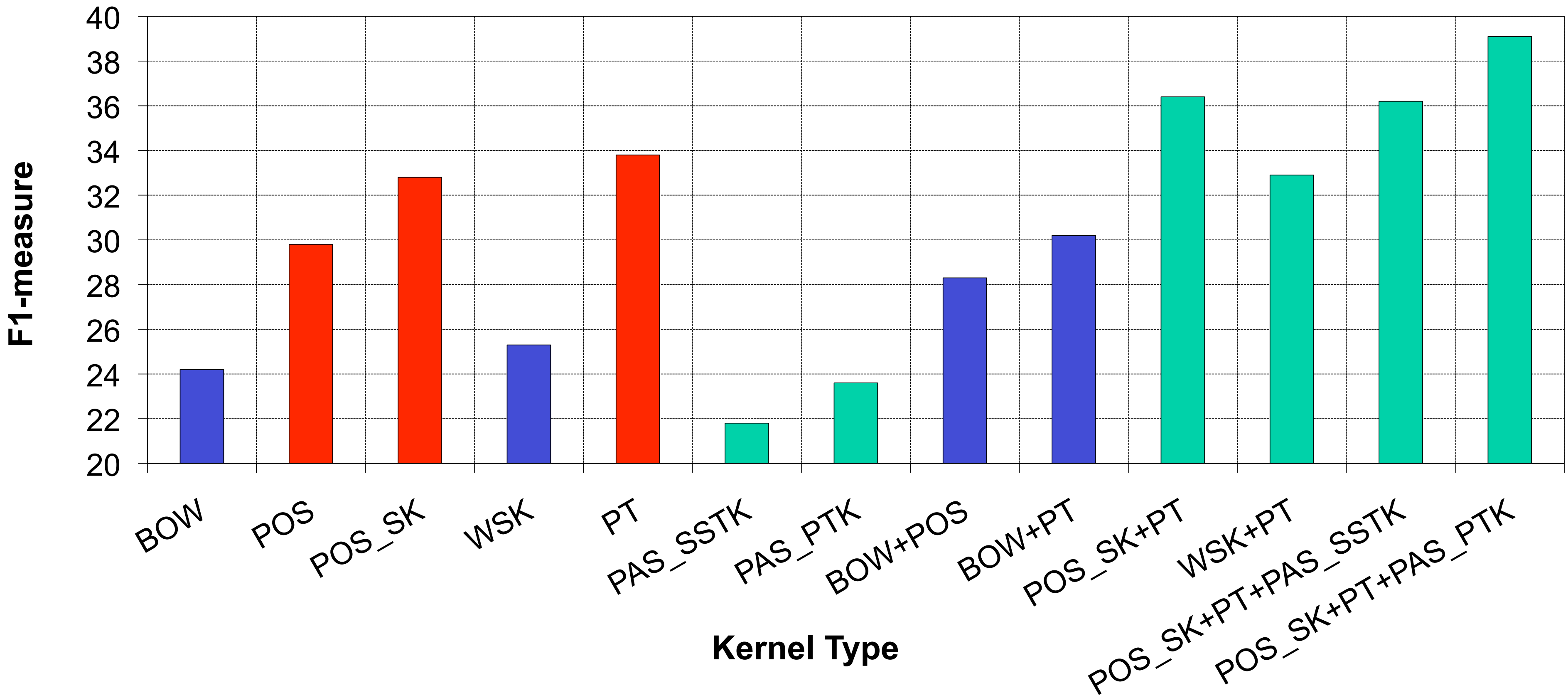
Results on TREC Data (5 folds cross validation)



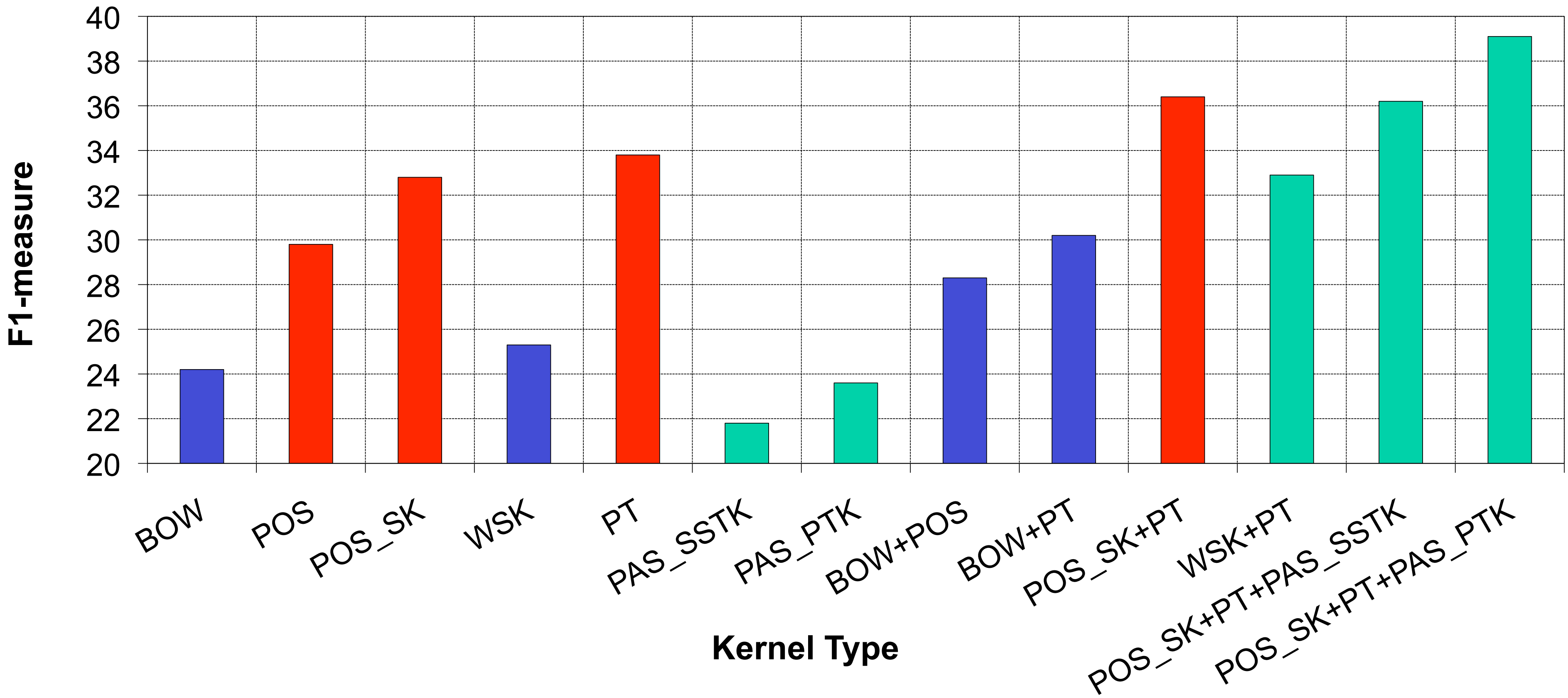
Results on TREC Data
(5 folds cross validation)



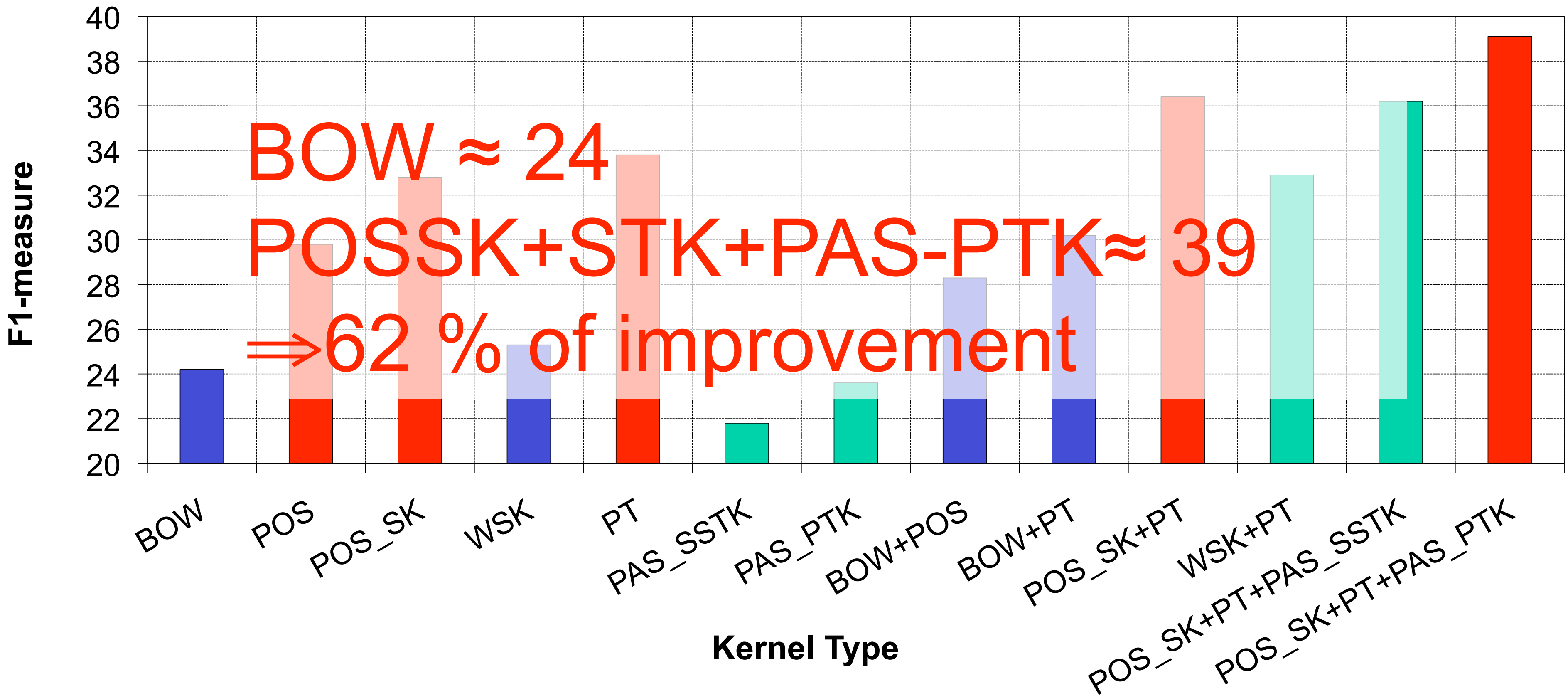
Results on TREC Data
(5 folds cross validation)



Results on TREC Data (5 folds cross validation)



Results on TREC Data
(5 folds cross validation)



BOW ≈ 24

POSSK+STK+PAS-PTK≈ 39

⇒62 % of improvement

Custom Kernel

- `double custom_kernel(KERNEL_PARM *kernel_parm, DOC *a, DOC *b) {`
- **`int i=0;`** `double k1;`
- `k1 =` **`tree_kernel`**`(kernel_parm, a, b, i, i);`
- `return k1;`
- `}`

Conclusions

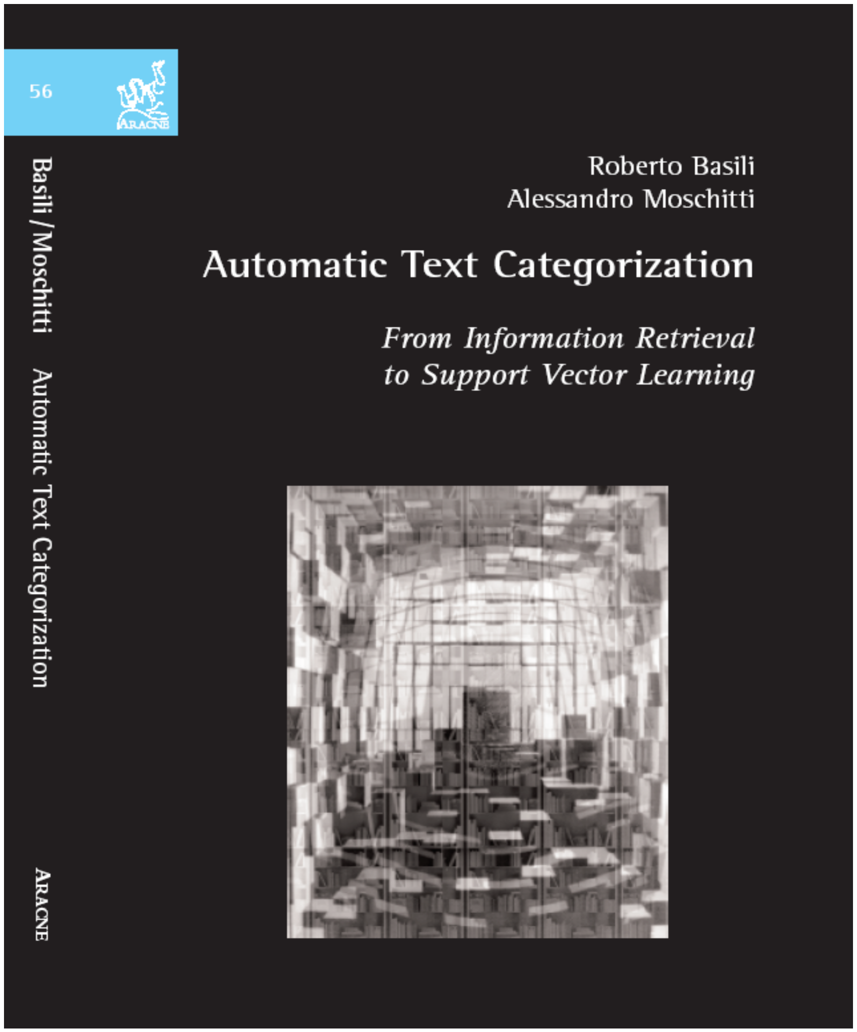
- Kernel methods and SVMs are useful tools to design language applications
- Kernel design still require some level of expertise
- Engineering approaches to tree kernels
  - Basic Combinations
  - Canonical Mappings, e.g.
    - Node Marking
  - Merging of kernels in more complex kernels
- State-of-the-art in SRL and QC
- An efficient tool to use them

Thank you

References

- Alessandro Moschitti, Silvia Quarteroni, Roberto Basili and Suresh Manandhar, *Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classification*, Proceedings of the 45th Conference of the Association for Computational Linguistics (ACL), Prague, June 2007.
- Alessandro Moschitti and Fabio Massimo Zanzotto, *Fast and Effective Kernels for Relational Learning from Texts*, Proceedings of The 24th Annual International Conference on Machine Learning (ICML 2007), Corvallis, OR, USA.
- Daniele Pighin, Alessandro Moschitti and Roberto Basili, *RTV: Tree Kernels for Thematic Role Classification*, Proceedings of the 4th International Workshop on Semantic Evaluation (SemEval-4), English Semantic Labeling, Prague, June 2007.
- Stephan Bloehdorn and Alessandro Moschitti, *Combined Syntactic and Semanitc Kernels for Text Classification*, to appear in the 29th European Conference on Information Retrieval (ECIR), April 2007, Rome, Italy.
- Fabio Aiolli, Giovanni Da San Martino, Alessandro Sperduti, and Alessandro Moschitti, *Efficient Kernel-based Learning for Trees*, to appear in the IEEE Symposium on Computational Intelligence and Data Mining (CIDM), Honolulu, Hawaii, 2007

An introductory book on SVMs, Kernel methods and Text Categorization



References

- Roberto Basili and Alessandro Moschitti, *Automatic Text Categorization: from Information Retrieval to Support Vector Learning*, Aracne editrice, Rome, Italy.
- Alessandro Moschitti, *Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees*. In Proceedings of the 17th European Conference on Machine Learning, Berlin, Germany, 2006.
- Alessandro Moschitti, Daniele Pighin, and Roberto Basili, *Tree Kernel Engineering for Proposition Re-ranking*, In Proceedings of Mining and Learning with Graphs (MLG 2006), Workshop held with ECML/PKDD 2006, Berlin, Germany, 2006.
- Elisa Cilia, Alessandro Moschitti, Sergio Ammendola, and Roberto Basili, *Structured Kernels for Automatic Detection of Protein Active Sites*. In Proceedings of Mining and Learning with Graphs (MLG 2006), Workshop held with ECML/PKDD 2006, Berlin, Germany, 2006.

References

- Fabio Massimo Zanzotto and Alessandro Moschitti, *Automatic learning of textual entailments with cross-pair similarities*. In Proceedings of COLING-ACL, Sydney, Australia, 2006.
- Alessandro Moschitti, *Making tree kernels practical for natural language learning*. In Proceedings of the Eleventh International Conference on European Association for Computational Linguistics, Trento, Italy, 2006.
- Alessandro Moschitti, Daniele Pighin and Roberto Basili. *Semantic Role Labeling via Tree Kernel joint inference*. In Proceedings of the 10th Conference on Computational Natural Language Learning, New York, USA, 2006.
- Alessandro Moschitti, Bonaventura Coppola, Daniele Pighin and Roberto Basili, *Semantic Tree Kernels to classify Predicate Argument Structures*. In Proceedings of the the 17th European Conference on Artificial Intelligence, Riva del Garda, Italy, 2006.

References

- Alessandro Moschitti and Roberto Basili, *A Tree Kernel approach to Question and Answer Classification in Question Answering Systems*. In Proceedings of the Conference on Language Resources and Evaluation, Genova, Italy, 2006.
- Ana-Maria Giuglea and Alessandro Moschitti, *Semantic Role Labeling via FrameNet, VerbNet and PropBank*. In Proceedings of the Joint 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL), Sydney, Australia, 2006.
- Roberto Basili, Marco Cammisa and Alessandro Moschitti, *Effective use of wordnet semantics via kernel-based learning*. In Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL 2005), Ann Arbor(MI), USA, 2005

References

- Alessandro Moschitti, Ana-Maria Giuglea, Bonaventura Coppola and Roberto Basili. Hierarchical Semantic Role Labeling. In Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL 2005 shared task), Ann Arbor(MI), USA, 2005.
- Roberto Basili, Marco Cammisa and Alessandro Moschitti, A Semantic Kernel to classify texts with very few training examples. In Proceedings of the Workshop on Learning in Web Search, at the 22nd International Conference on Machine Learning (ICML 2005), Bonn, Germany, 2005.
- Alessandro Moschitti, Bonaventura Coppola, Daniele Pighin and Roberto Basili. Engineering of Syntactic Features for Shallow Semantic Parsing. In Proceedings of the ACL05 Workshop on Feature Engineering for Machine Learning in Natural Language Processing, Ann Arbor (MI), USA, 2005.

References

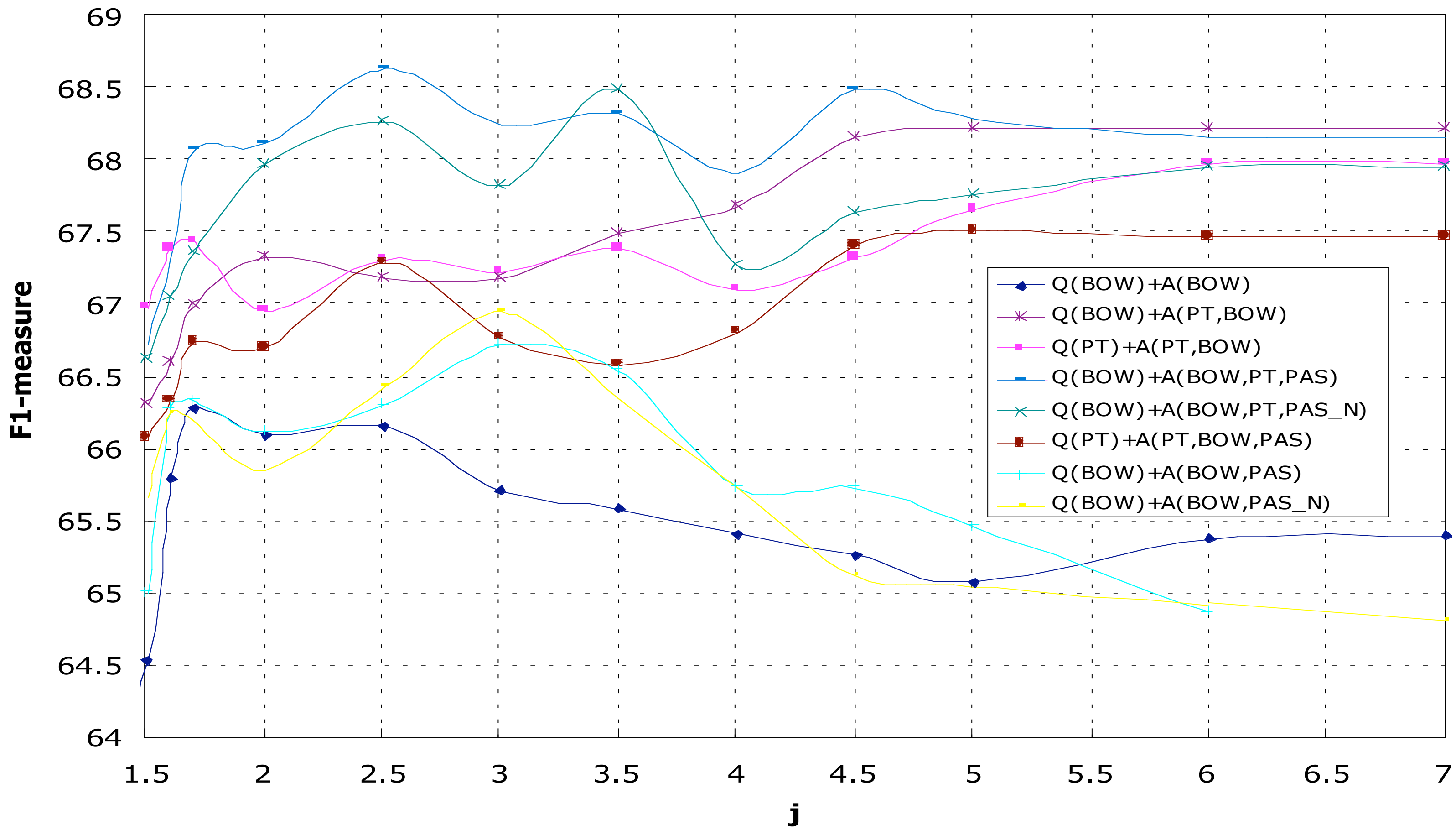
- Alessandro Moschitti, *A study on Convolution Kernel for Shallow Semantic Parsing*. In proceedings of ACL-2004, Spain, 2004.
- Alessandro Moschitti and Cosmin Adrian Bejan, *A Semantic Kernel for Predicate Argument Classification.* In proceedings of the CoNLL-2004, Boston, MA, USA, 2004.
- M. Collins and N. Duffy, *New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron*. In ACL02, 2002.
- S.V.N. Vishwanathan and A.J. Smola. *Fast kernels on strings and trees*. In Proceedings of Neural Information Processing Systems, 2002.

References

- *AN INTRODUCTION TO SUPPORT VECTOR MACHINES (and other kernel-based learning methods)* N. Cristianini and J. Shawe-Taylor Cambridge University Press
- Xavier Carreras and Llu´ıs M`arquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *proceedings of CoNLL'05*.
- Sameer Pradhan, Kadri Hacioglu, Valeri Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support vector learning for semantic argument classification. *to appear in Machine Learning Journal*.

function Evaluate_Pair_Set(Tree T_1, T_2) returns NODE_PAIR_SET;
LIST L_1,L_2;
NODE_PAIR_SET N_p;
begin
    $L_1 = T_1$.ordered_list;
    $L_2 = T_2$.ordered_list; */*the lists were sorted at loading time*/*
    $n_1 =$ extract(L_1); */*get the head element and*/*
    $n_2 =$ extract(L_2); */*remove it from the list*/*
    while (n_1 **and** n_2 are not NULL)
        if (production_of(n_1) $>$ production_of(n_2))
            then $n_2 =$ extract(L_2);
            else if (production_of(n_1) $<$ production_of(n_2))
                then $n_1 =$ extract(L_1);
                else
                    while (production_of(n_1) $==$ production_of(n_2))
                        while (production_of(n_1) $==$ production_of(n_2))
                            add($\langle n_1, n_2 \rangle$, N_p);
                            n_2=get_next_elem(L_2); */*get the head element*
                            and move the pointer to the next element/*
                        end
                        $n_1 =$ extract(L_1);
                        reset(L_2); */*set the pointer at the first element*/*
                    end
    end
    return N_p ;
end

The Impact of SSTK in Answer Classification



Mercer's conditions (1)

Def. B.11 *Eigen Values*
Given a matrix $\boldsymbol{A} \in \mathbb{R}^m \times \mathbb{R}^n$, *an egeinvalue* λ *and an egeinvector* $\vec{x} \in \mathbb{R}^n - \{\vec{0}\}$ *are such that*

$$\boldsymbol{A}\vec{x} = \lambda\vec{x}$$

Def. B.12 *Symmetric Matrix*
A square matrix $\boldsymbol{A} \in \mathbb{R}^n \times \mathbb{R}^n$ *is symmetric iff* $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$ *for* $i \neq j$ $i = 1, .., m$ *and* $j = 1, .., n$, *i.e. iff* $\boldsymbol{A} = \boldsymbol{A}'$.

Def. B.13 *Positive (Semi-) definite Matrix*
A square matrix $\boldsymbol{A} \in \mathbb{R}^n \times \mathbb{R}^n$ *is said to be positive (semi-) definite if its eigenvalues are all positive (non-negative).*

Mercer's conditions (2)

Proposition 2.27 *(Mercer's conditions)*
Let X be a finite input space with $K(\vec{x}, \vec{z})$ a symmetric function on X. Then $K(\vec{x}, \vec{z})$ is a kernel function if and only if the matrix

$$k(\vec{x}, \vec{z}) = \phi(\vec{x}) \cdot \phi(\vec{z})$$

is positive semi-definite (has non-negative eigenvalues).

- If the Gram matrix: $G = k(\vec{x}_i, \vec{x}_j)$
  is positive semi-definite there is a mapping ϕ that produces the target kernel function

The lexical semantic kernel is not always a kernel

- It may not be a kernel so we can use ***M´·M***, where ***M*** is the initial similarity matrix

Proposition B.14 *Let* $\boldsymbol{A}$ *be a symmetric matrix. Then* A *is positive (semi-) definite iff for any vector* $\vec{x} \neq 0$

$$\vec{x}'\boldsymbol{A}\vec{x} > \lambda\vec{x} \quad (\geq 0).$$

From the previous proposition it follows that: If we find a decomposition $\boldsymbol{A}$ in $\boldsymbol{M}'\boldsymbol{M}$, then $\boldsymbol{A}$ is semi-definite positive matrix as

$$\vec{x}'\boldsymbol{A}\vec{x} = \vec{x}'\boldsymbol{M}'\boldsymbol{M}\vec{x} = (\boldsymbol{M}\vec{x})'(\boldsymbol{M}\vec{x}) = \boldsymbol{M}\vec{x} \cdot \boldsymbol{M}\vec{x} = ||\boldsymbol{M}\vec{x}||^2 \geq 0.$$