Tree-Based Convolution and its Applications

Lili Mou, Ph.D. Candidate

Software Institute, Peking University

doublepower.mou@gmail.com

http://sei.pku.edu.cn/~moull12

Outline

- **Introduction: Sentence Modeling**
- Related Work: CNNs, RNNs, etc
- Tree-Based Convolution
- Conclusion and Discussion

Discriminative Sentence Modeling

- Sentence modeling
  - Capture the “meaning” of a sentence
- Discriminative sentence modeling
  - Classify a sentence according to a certain criterion

    E.g., sentiment analysis, polarity classification

Discriminative Sentence Modeling

- Sentence modeling
  - Capture the “meaning” of a sentence
- Discriminative sentence modeling
  - Classify a sentence according to certain criterion
- Related to various NLP tasks
  - Sentence matching: QA [1], conversation [2]
  - Discourse analysis [3]
  - Extractive summarization [4]
  - Parsing [5]
  - Machine translation [6]

An Example: Sentiment Analysis

A movie review

An idealistic love story that brings out the latent 15-year-old romantic in everyone.

The sentiment?

Positive Neutral Negative

Human Engineering

- Feature engineering
  - Bag-of-words, n-gram, sentiment lexicon [7]

However, sentence modeling is usually non-trivial [8]

```
white blood cells destroying an infection
an infection destroying white blood cells
```

- Kernel machines, e.g., SVM
  - Circumvent explicit feature representation
  - The kernel is crucial as it fully summarizes data information

Neural Networks

- Automatic feature learning
  - Word embeddings [9]
  - Paragraph vectors [10]

- Prevailing neural sentence models
  - Convolutional neural networks (CNNs) [11]
    - Capture invariant features
    - Efficient feature extraction and learning
    - Structure insensitive
  - Recursive neural networks (RNNs) [8, 12, 13]
    - Structure sensitive
    - Long propagation path

Our Intuition

- Can we combine?
  - Short propagation path like convolutional nets
  - Structure-sensitive like recursive nets

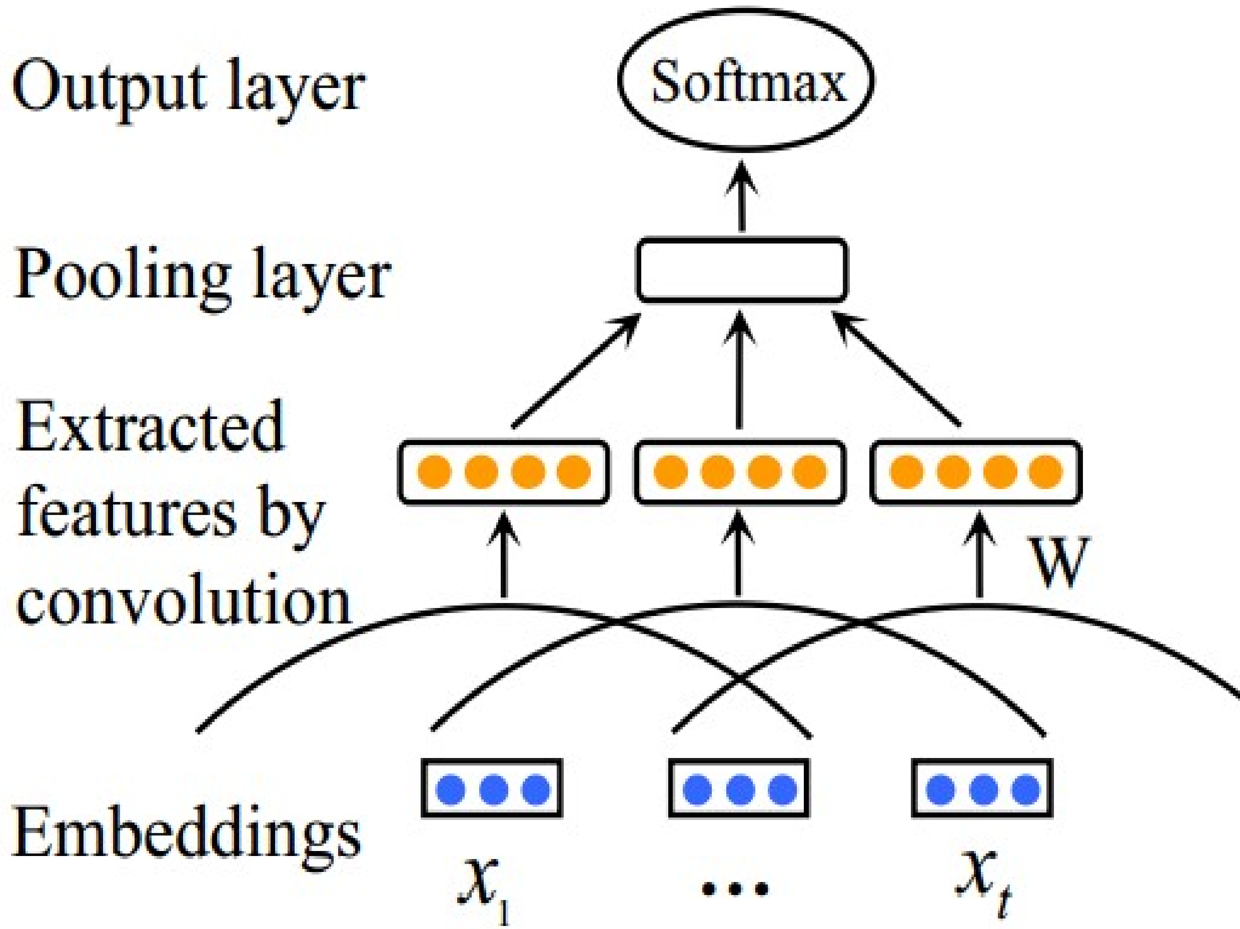
- Our solution: Tree-based convolution [14, 15, 16]
  - Design subtree convolutional kernel to extract invariant structural features

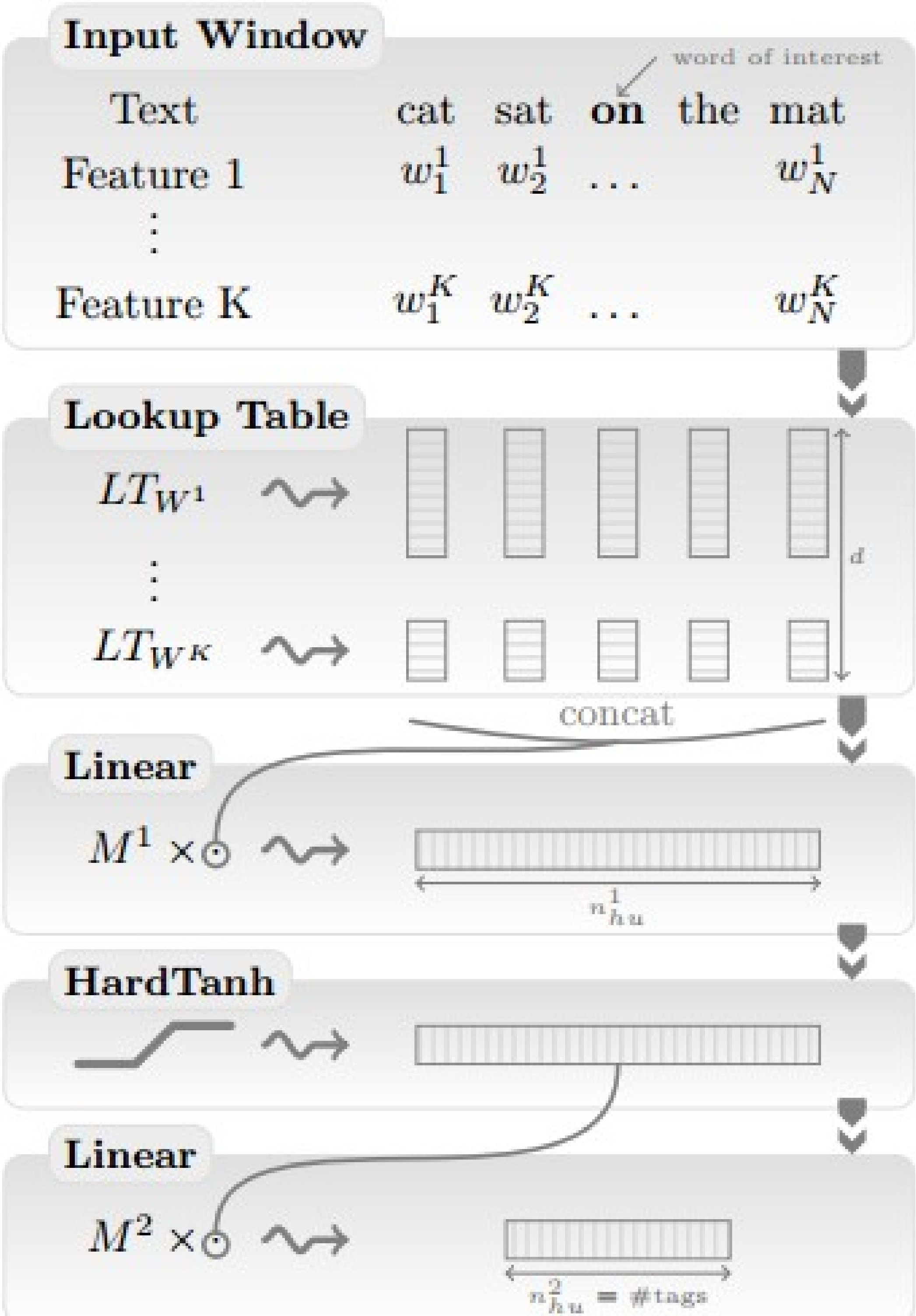
Outline

- Introduction: Sentence Modeling
- **Related Work: CNNs, RNNs, etc**
- Tree-Based Convolution
- Conclusion

Convolutional Neural Networks (CNNs)

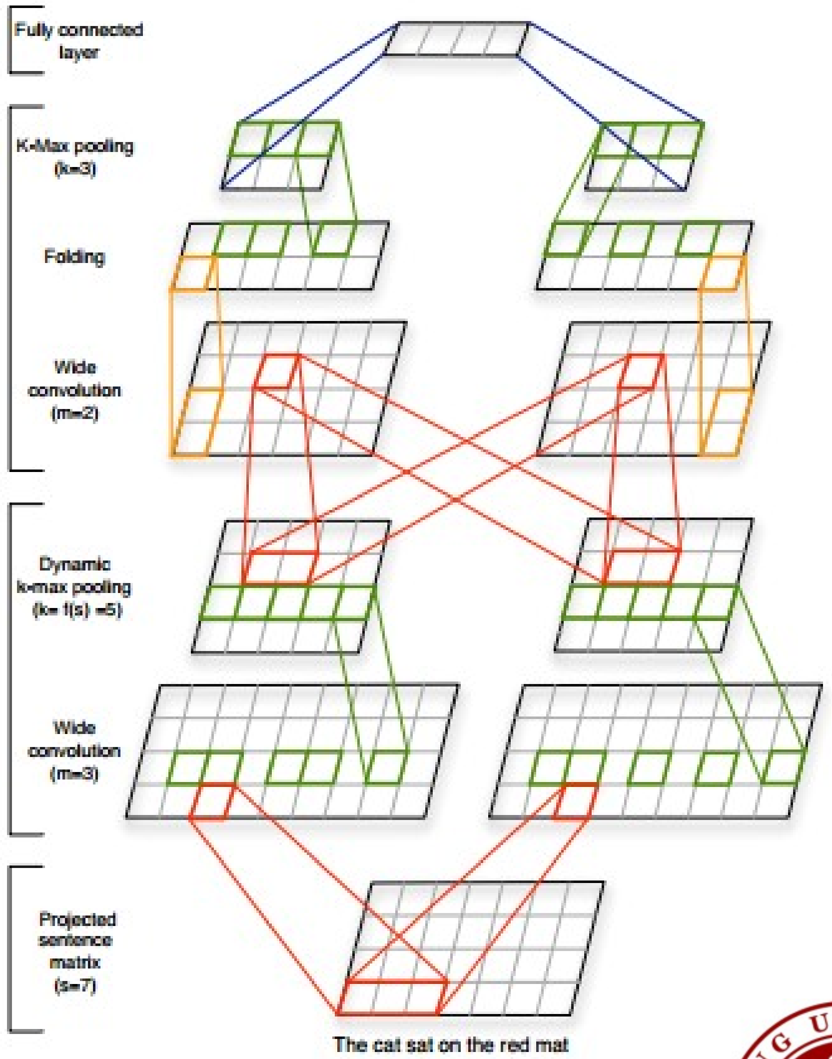
- **Convolution** in signal processing: Linear time-invariant system
  - Flip, inner-product, and slide
- **Convolution** in the neural network regime
  - Sliding window





[17] Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. JMLR, 2011.

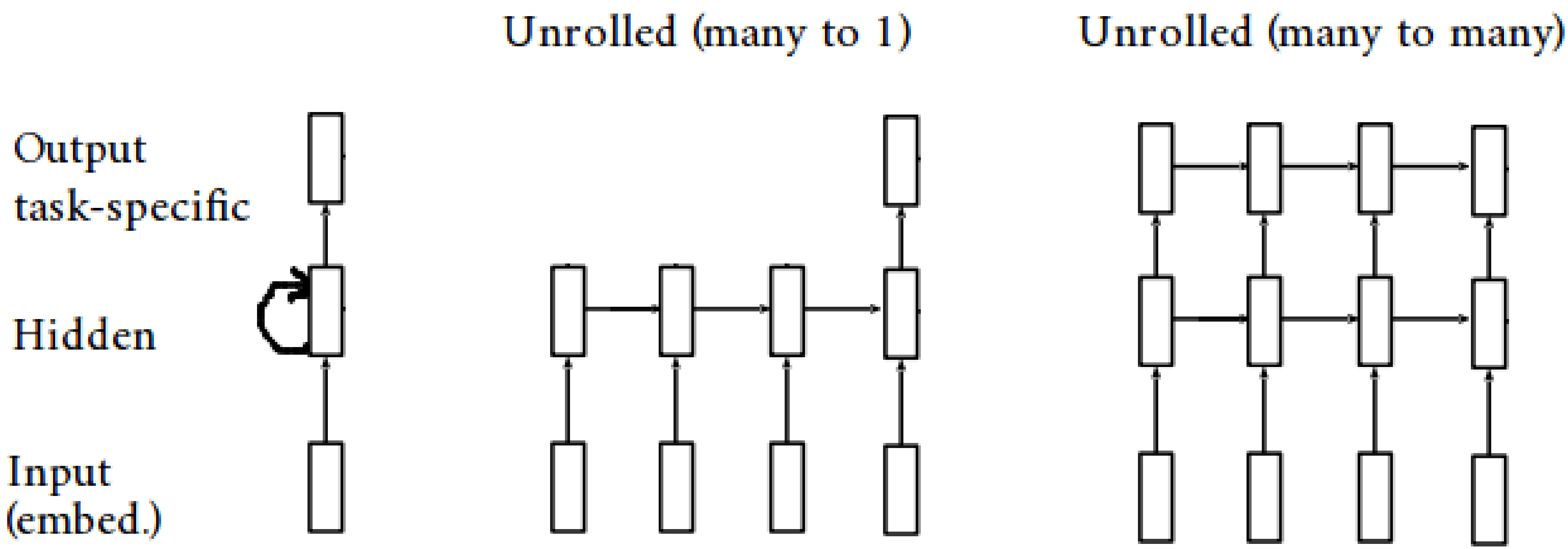
Convolutional Neural Networks (CNNs)



[11] Blunsom, Phil, Edward Grefenstette, and Nal Kalchbrenner. "A Convolutional Neural Network for Modelling Sentences." ACL, 2014.

Recurrent Nerual Networks (RNNs)

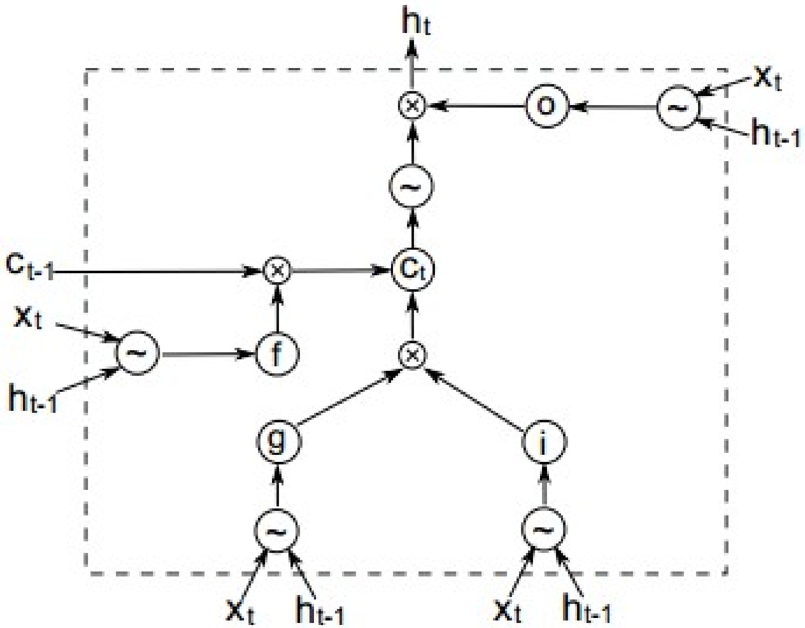
- A recurrent net keeps one or a few hidden layers as the state
- The state changes according to the discrete input



Problem

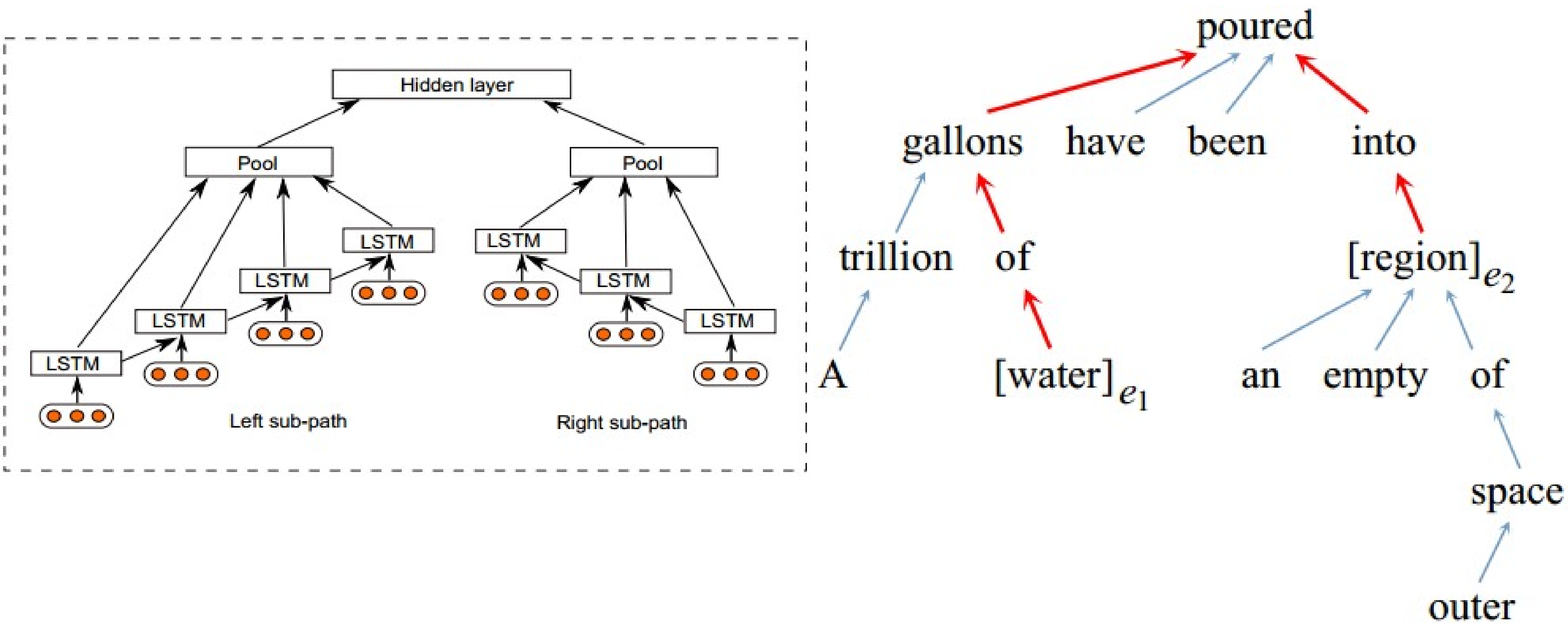
- Gradient vanishing or exploding
  - Backpropagation is a linear system
- Design long short term memory (LSTM) units or gate recurrent units (GRU) to alleviate the problem

  (**N.B.**: Not completely solved)



$$
\begin{aligned}
\boldsymbol{i}_t &= \sigma(W_i \cdot \boldsymbol{x}_t + U_i \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{f}_t &= \sigma(W_f \cdot \boldsymbol{x}_t + U_f \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{o}_t &= \sigma(W_o \cdot \boldsymbol{x}_t + U_o \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \\
\boldsymbol{g}_t &= \tanh(W_g \cdot \boldsymbol{x}_t + U_g \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_g) \\
\boldsymbol{c}_t &= \boldsymbol{i}_t \otimes \boldsymbol{g}_t + \boldsymbol{f}_t \otimes \boldsymbol{c}_{t-1} \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \otimes \tanh(\boldsymbol{c}_t)
\end{aligned}
$$

Example: Relation Classification



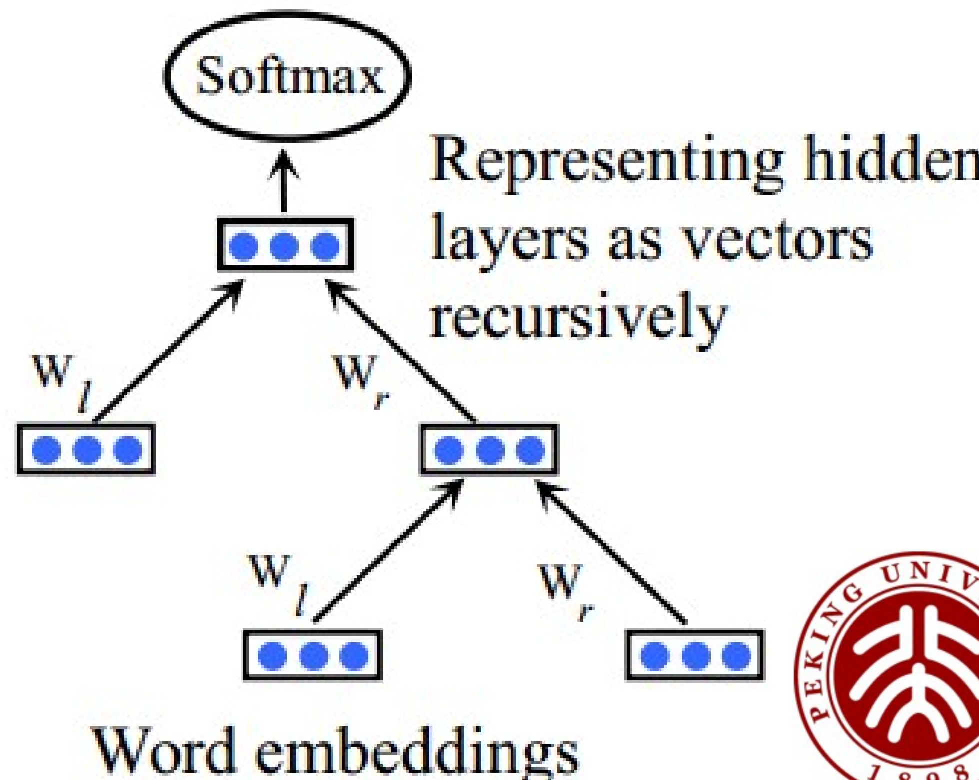
[18] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng and Zhi Jin. "Classifying relations via long short term memory networks along shortest dependency paths." In EMNLP, pages 1785--1794, 2015.

Recursive Neural Networks (RNNs again)

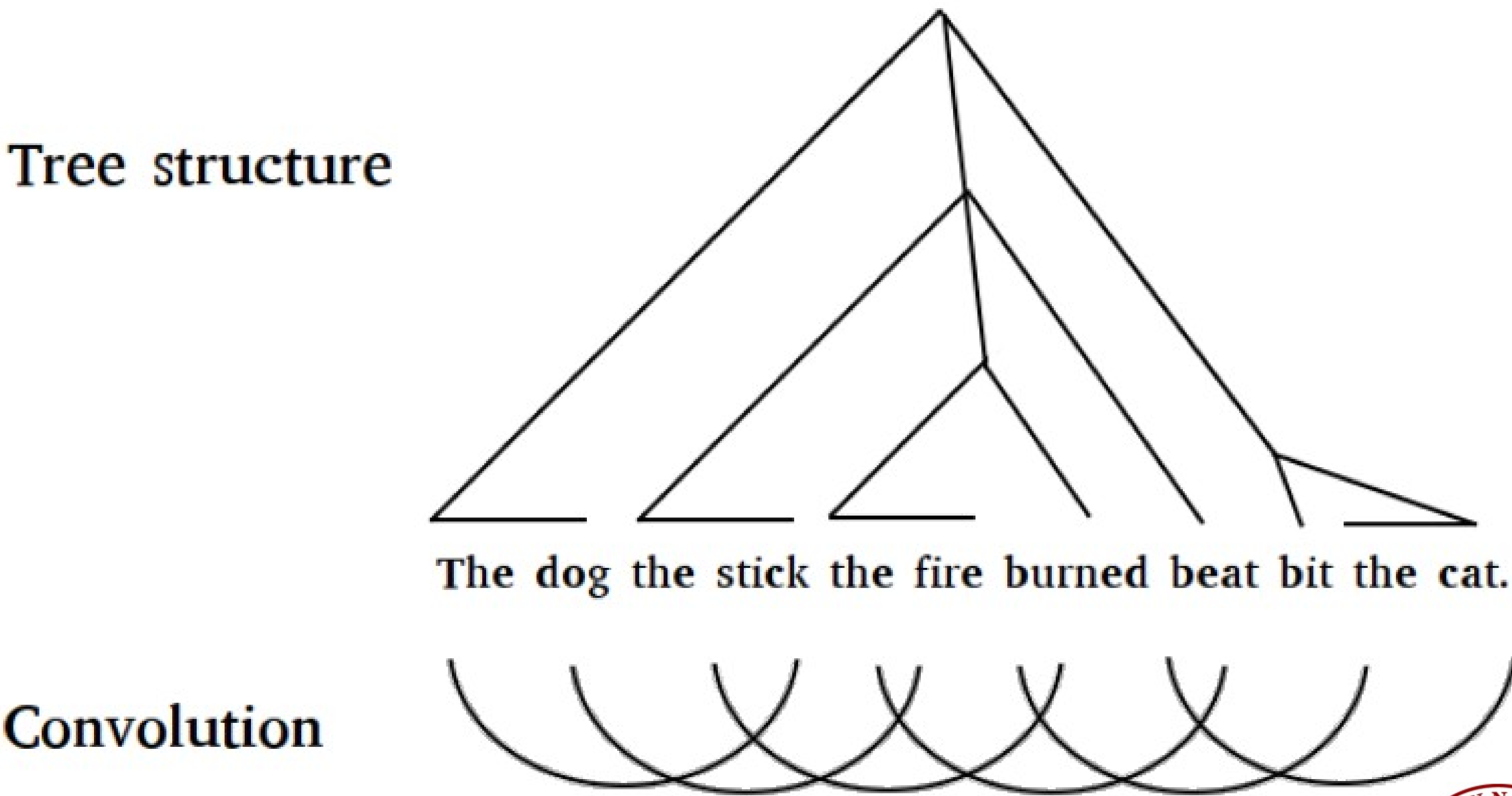
- Where does the tree come from?
  - Dynamically constructing a tree structure similar to constituency
  - Parsed by external parsers

Constituency tree

- Leaf nodes = words
- Interior nodes = abstract components of a sentence (e.g., noun phrase)
- Root nodes = the whole sentence

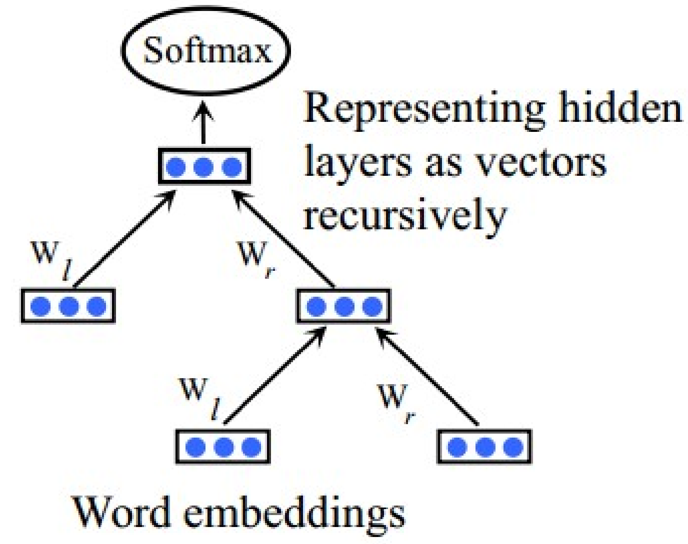


Why parse trees may be important?



[19] Pinker, Steven. The Language Instinct: The New Science of Language and Mind. Penguin UK, 1995.

Recursive Propagation

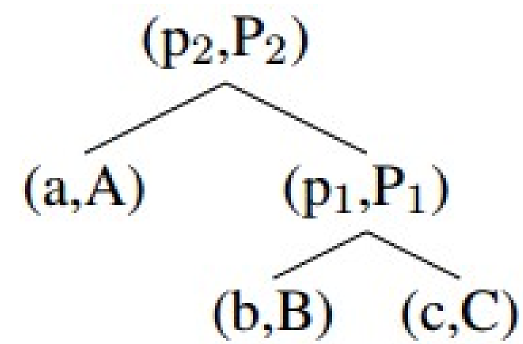


- Perception-like interaction [8]

  $p = f(W[c_1\ c_2]^T)$

- Matrix-vector interaction [12]

$$p_1 = f\left(W\begin{bmatrix} Cb \\ Bc \end{bmatrix}\right), P_1 = f\left(W_M\begin{bmatrix} B \\ C \end{bmatrix}\right)$$



- Tensor interaction [13]

$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]}\begin{bmatrix} b \\ c \end{bmatrix} + W\begin{bmatrix} b \\ c \end{bmatrix}\right)$$

[8] Socher R, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions. EMNLP, 2011

[12] Socher, R, et al. "Semantic compositionality through recursive matrix-vector spaces." EMNLP-CoNLL, 2012.

[13] Socher, R, et al. "Recursive deep models for semantic compositionality over a sentiment treebank." EMNLP, 2013.

Recursive Propagation

- Perception-like interaction [8]

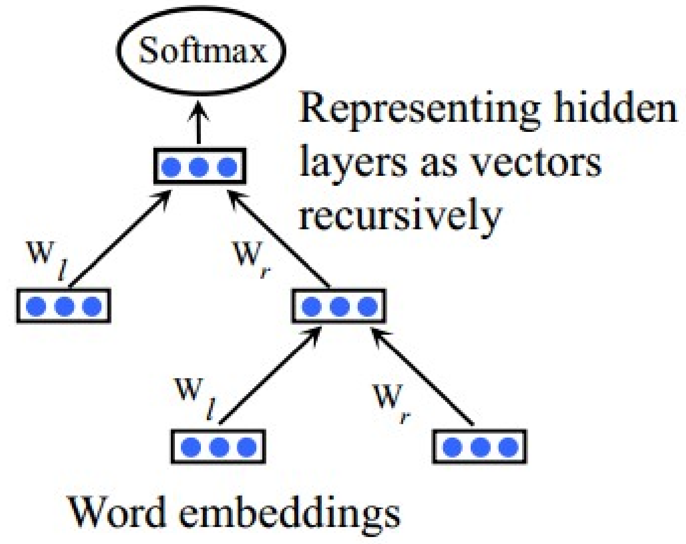
  $p = f(W[c_1\ c_2]^T)$

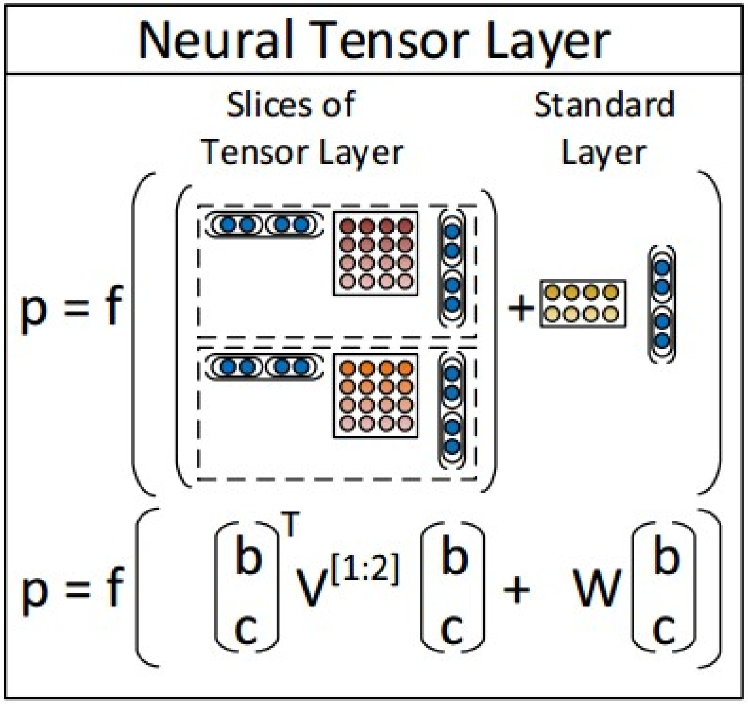
- Matrix-vector interaction [12]

$$p_1 = f\left(W\begin{bmatrix} Cb \\ Bc \end{bmatrix}\right), P_1 = f\left(W_M\begin{bmatrix} B \\ C \end{bmatrix}\right)$$

- Tensor interaction [13]

$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]}\begin{bmatrix} b \\ c \end{bmatrix} + W\begin{bmatrix} b \\ c \end{bmatrix}\right)$$





[8] Socher R, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions. EMNLP, 2011

[12] Socher, R, et al. "Semantic compositionality through recursive matrix-vector spaces." EMNLP-CoNLL, 2012.

[13] Socher, R, et al. "Recursive deep models for semantic compositionality over a sentiment treebank." EMNLP, 2013.

Even More Interaction

- LSTM interaction [18, 19, 20]

$$\tilde{h}_j = \sum_{k \in C(j)} h_k,$$

$$i_j = \sigma\left(W^{(i)} x_j + U^{(i)} \tilde{h}_j + b^{(i)}\right),$$

$$f_{jk} = \sigma\left(W^{(f)} x_j + U^{(f)} h_k + b^{(f)}\right),$$

$$o_j = \sigma\left(W^{(o)} x_j + U^{(o)} \tilde{h}_j + b^{(o)}\right),$$

$$u_j = \tanh\left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)}\right),$$

$$c_j = i_j \odot u_j + \sum_{k \in C(j)} f_{jk} \odot c_k,$$

$$h_j = o_j \odot \tanh(c_j),$$

[18] Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. "Improved semantic representations from tree-structured long short-term memory networks." ACL, 2015

[19] Zhu, Xiaodan, Parinaz Sobihani, and Hongyu Guo. "Long short-term memory over recursive structures." ICML, 2015.

[20] Le, Phong, and Willem Zuidema. "Compositional distributional semantics with long short term memory." arXiv:1503.02510 (2015).

Outline

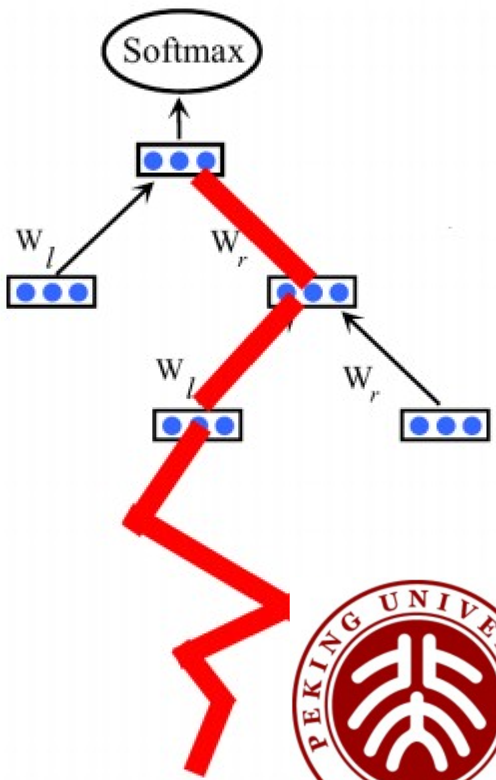
- Introduction: Sentence Modeling
- Related Work: CNNs, RNNs, etc
- **Tree-Based Convolution**
- Discussion
- Conclusion

Tree-Based Convolutional Neural Network (TBCNN)

- CNNs
  - 😋 Efficient feature learning and extraction

    The propagation path is irrelevant to the length of a sentence
  - 😿 Structure-insensitive
- Recursive networks
  - 😋 Structure-sensitive
  - 😿 Long propagation path

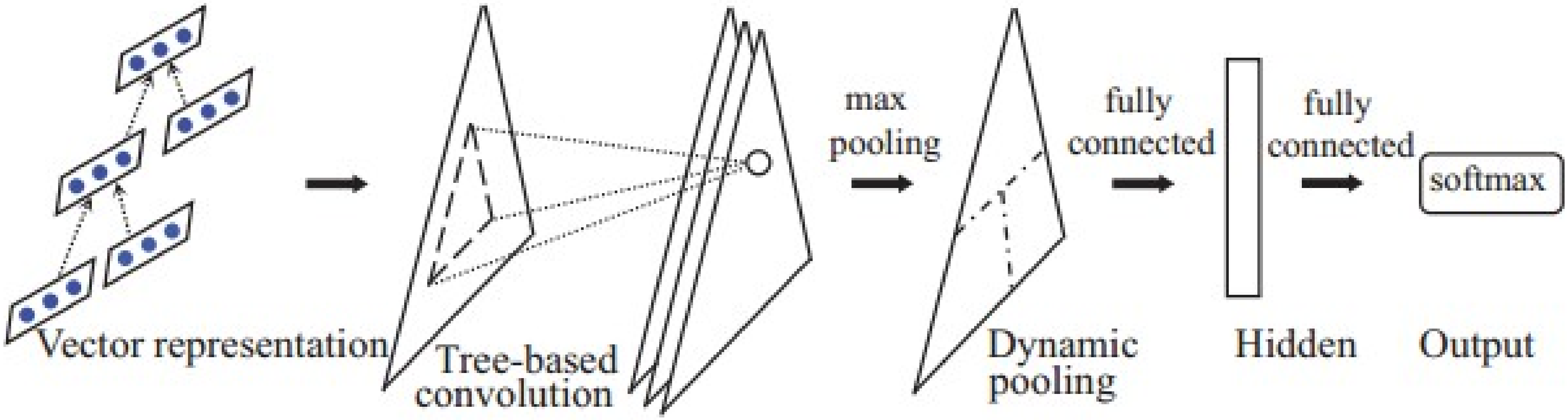
    The problem of “gradient vanishing or explosion”



Our intuition

- Can we combine?
  - Structure-sensitive as recursive neural networks
  - Short propagation path as convolutional neural networks
- Solution
  - The tree-based convolutional neural network (TBCNN)
    - Recall convolution = sliding window in the NN regime
    - Tree-based convolution = sliding window of a subtree

Tree-Based Convolution

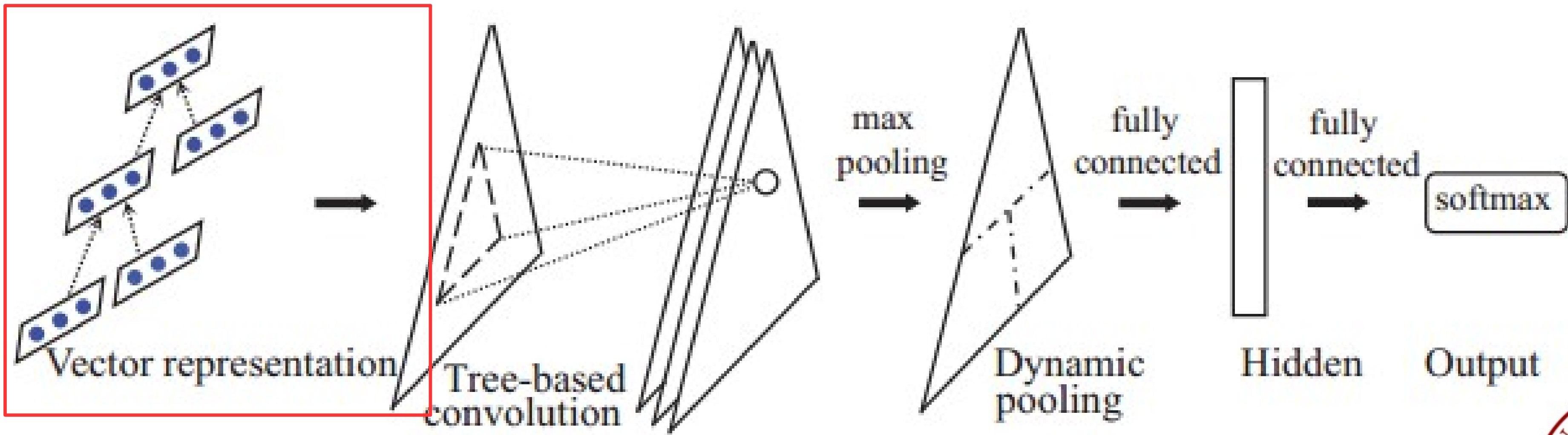


$$\boldsymbol{y} = f\left(\sum_{i=1}^{t} W_i \cdot \boldsymbol{x}_i + \boldsymbol{b}\right)$$

Technical Difficulties

- How to represent nodes as vectors?
- How to determine weights?
- How to aggregate information?

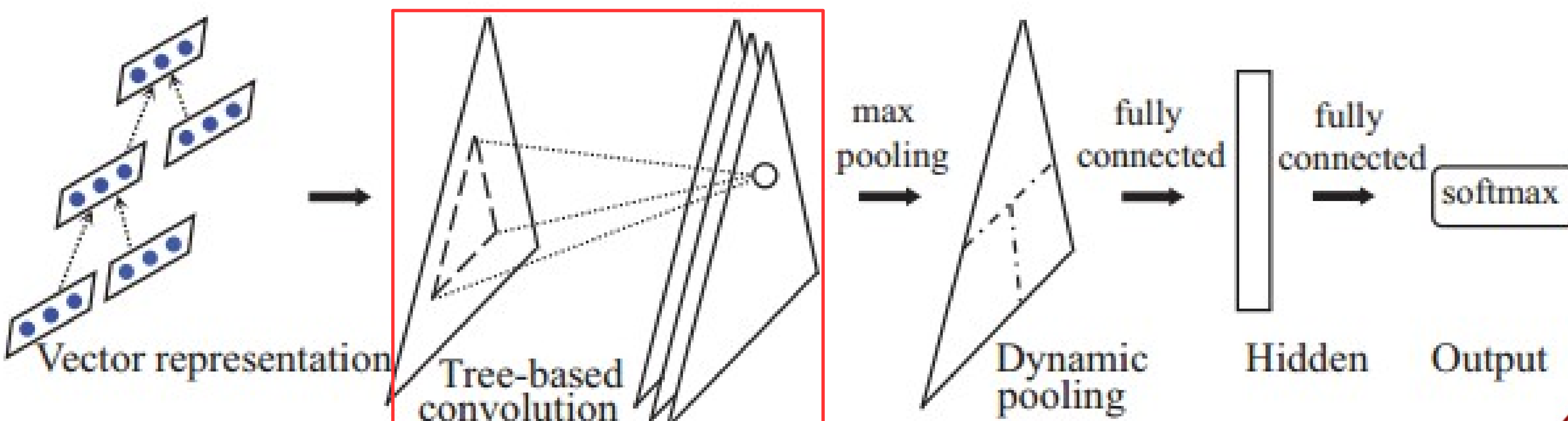
$$\boldsymbol{y} = f\left(\sum_{i=1}^{t} W_i \cdot \boxed{\boldsymbol{x}_i} + \boldsymbol{b}\right)$$



Technical Difficulties

- How to represent nodes as vectors?
- How to determine weights?
- How to aggregate information?

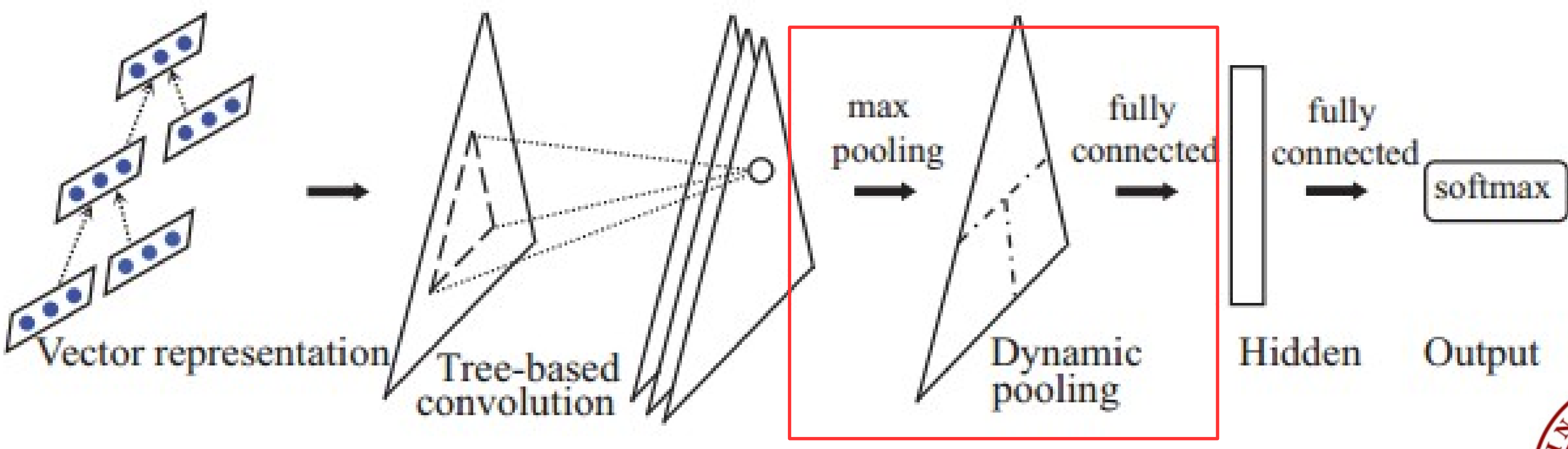
$$\boldsymbol{y} = f\left(\sum_{i=1}^{t} \boxed{W_i} \boldsymbol{x}_i + \boldsymbol{b}\right)$$



Technical Difficulties

- How to represent nodes as vectors?
- How to determine weights?
- How to aggregate information?

$$\boxed{\boldsymbol{y}} = f\left(\sum_{i=1}^{t} W_i \cdot \boldsymbol{x}_i + \boldsymbol{b}\right)$$



A Few Variants

- Constituency tree-based convolutional neural network (c-TBCNN) [15]
- Dependency tree-based convolutional neural network

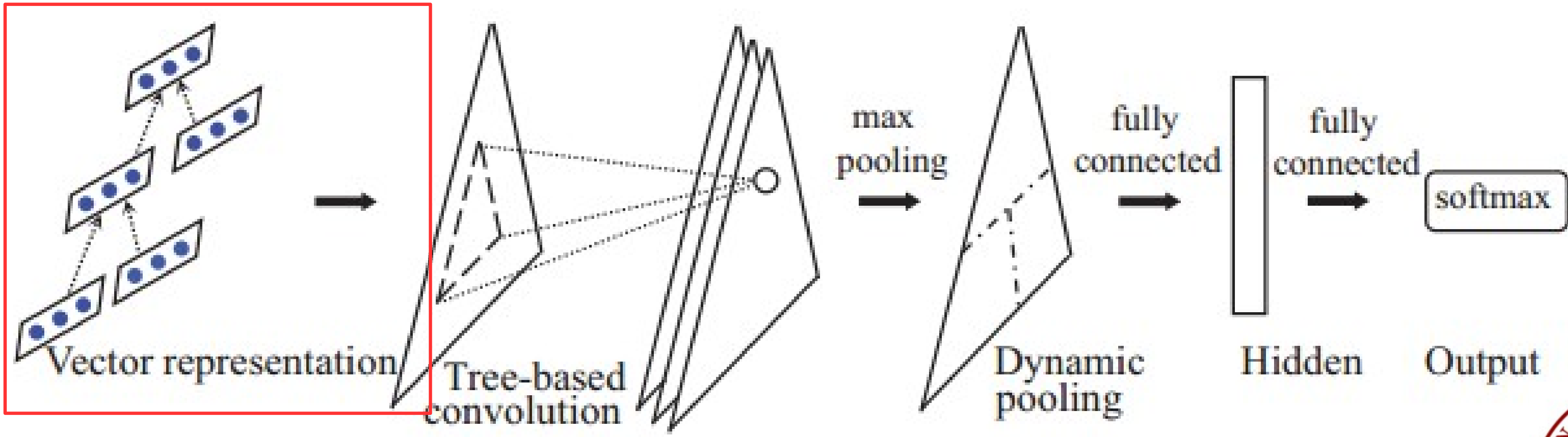
  (d-TBCNN) [15]
- Abstract syntrax tree-based convolutional neural network

  (asTBCNN) [14]

c-TBCNN: constituency tree

- How to represent nodes as vectors?
- Pretrain a recursive net

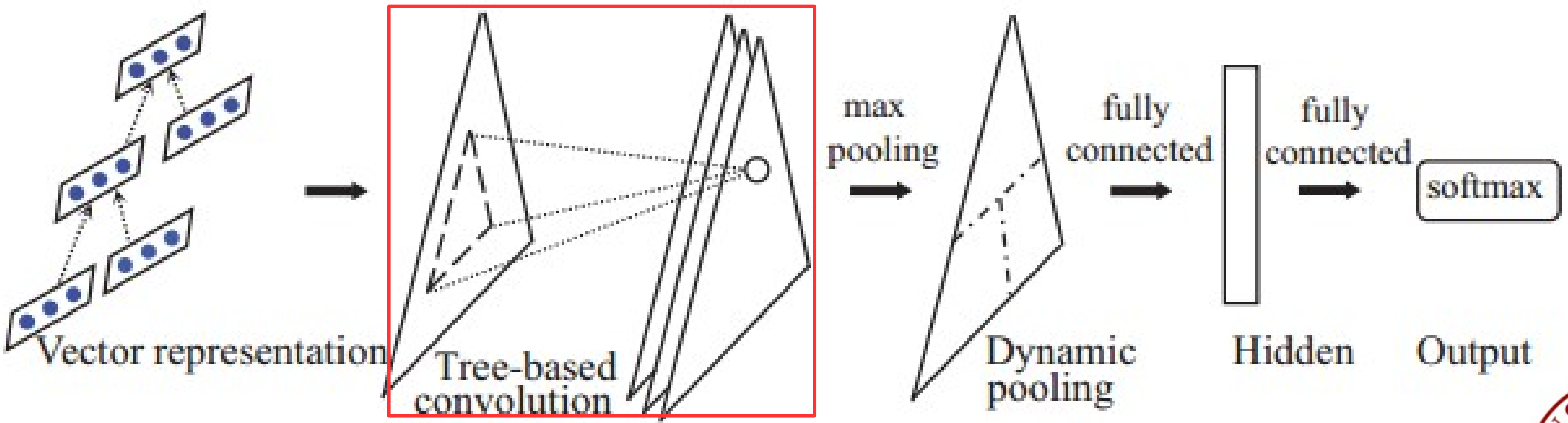
$$\boldsymbol{y} = f\left(\sum_{i=1}^{t} W_i \cdot \boxed{\boldsymbol{x}_i} + \boldsymbol{b}\right)$$



c-TBCNN: constituency tree

- How to determine weights?
- 3 matrices as parameters
  - Parent, left child, and right child

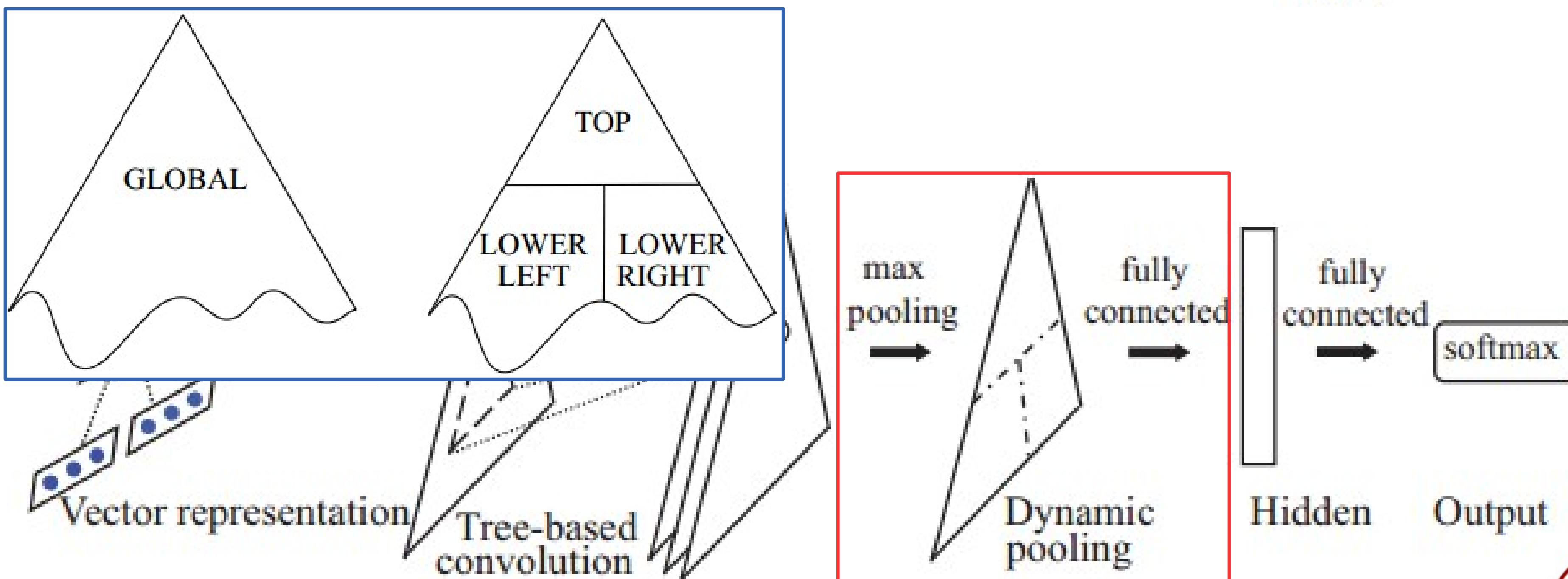
$$y = f\left(\sum_{i=1}^{t} \boxed{W_i} \cdot x_i + b\right)$$



c-TBCNN: constituency tree

- How to aggregate information?
- Dynamic pooling: 1-way v.s. 3-way

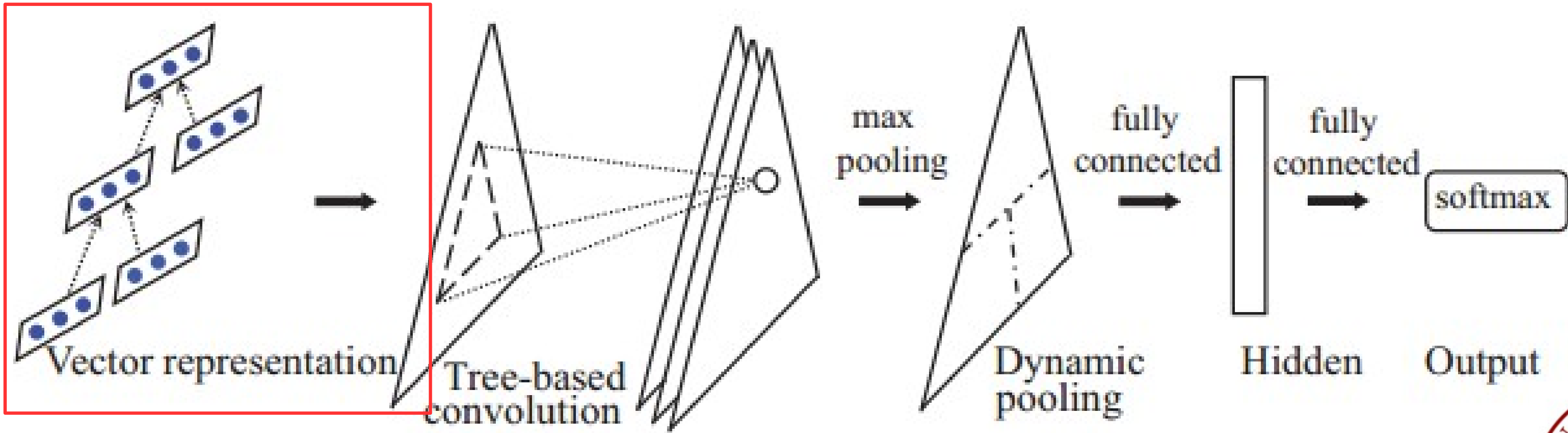
$$\boxed{\boldsymbol{y}} = f\left(\sum_{i=1}^{t} W_i \cdot \boldsymbol{x}_i + \boldsymbol{b}\right)$$



d-TBCNN: dependency tree

- How to represent nodes as vectors?
- Word embeddings

$$y = f\left(\sum_{i=1}^{t} W_i \cdot \boxed{x_i} + b\right)$$

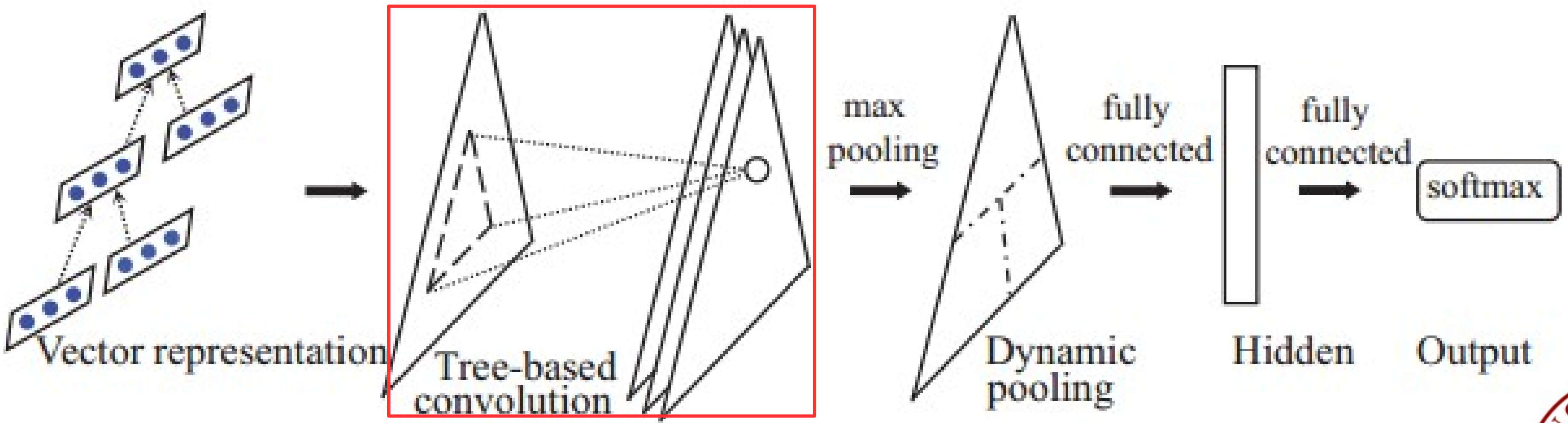


d-TBCNN: dependency tree

- How to determine weights?
- Assign weight by dependency type (e.g., nsubj)

$$\boldsymbol{y} = f\left(\sum_{i=1}^{t} \boxed{W_i} \cdot \boldsymbol{x}_i + \boldsymbol{b}\right)$$

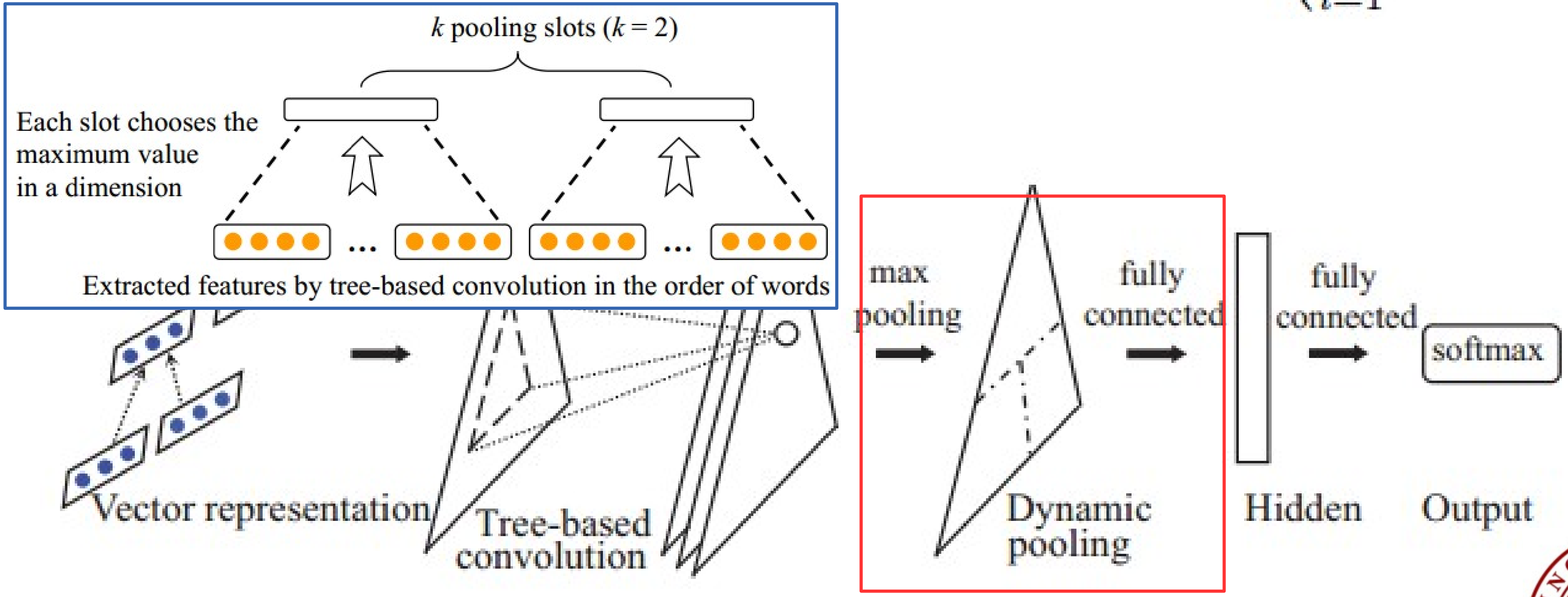
$$\boldsymbol{y} = f\left(W_p^{(d)} \cdot \boldsymbol{p} + \sum_{i=1}^{n} W_{r[c_i]}^{(d)} \cdot \boldsymbol{c}_i + \boldsymbol{b}^{(d)}\right)$$



d-TBCNN: dependency tree

- How to aggregate information?
- *k*-way pooling

$$\boxed{\boldsymbol{y}} = f\left(\sum_{i=1}^{t} W_i \cdot \boldsymbol{x}_i + \boldsymbol{b}\right)$$



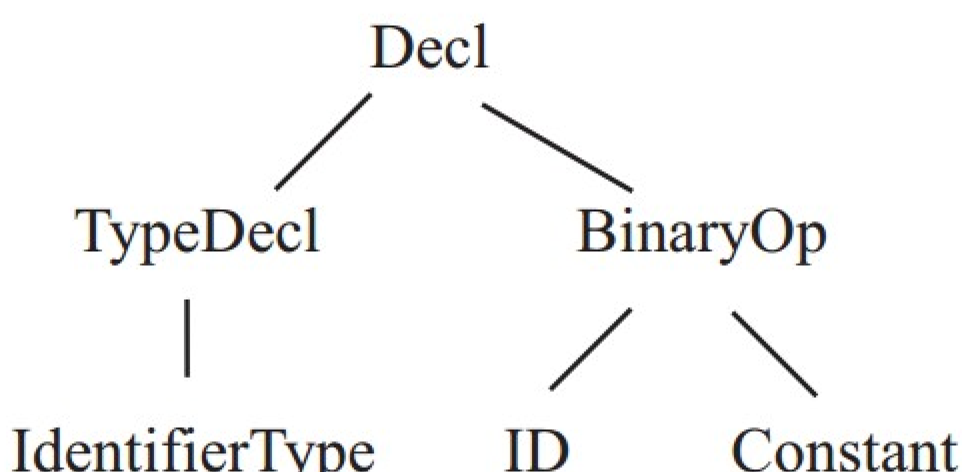
asTBCNN: abstract syntax tree

Programming language processing

vs

Natural language processing

- Abstract syntax tree of “int a = b + 3”

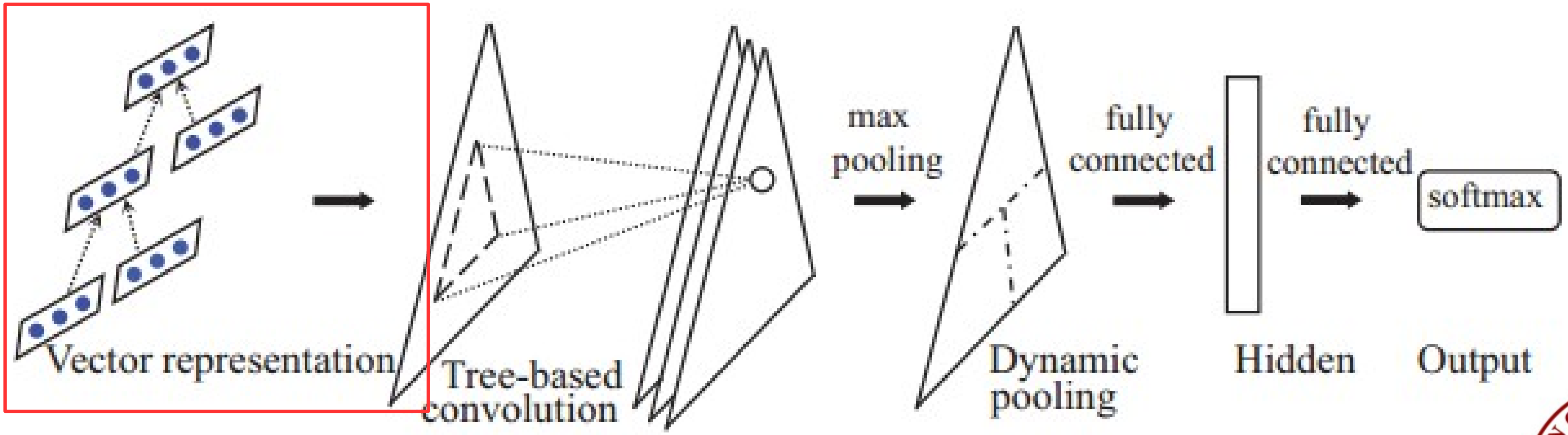


asTBCNN: abstract syntax tree

- How to represent nodes as vectors?
- Pretrain or random initialization

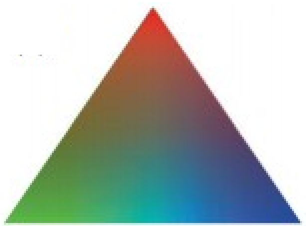
  (44 AST symbols)

$$y = f\left(\sum_{i=1}^{t} W_i \cdot \boxed{x_i} + b\right)$$

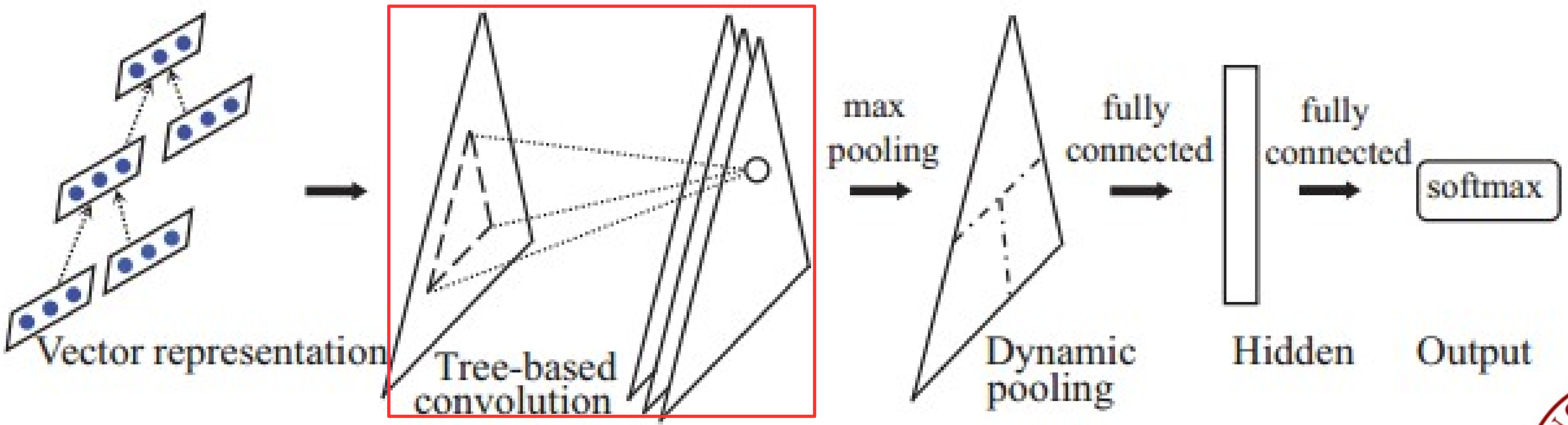


asTBCNN: abstract syntax tree

- How to determine weights?
- Continuous binary tree



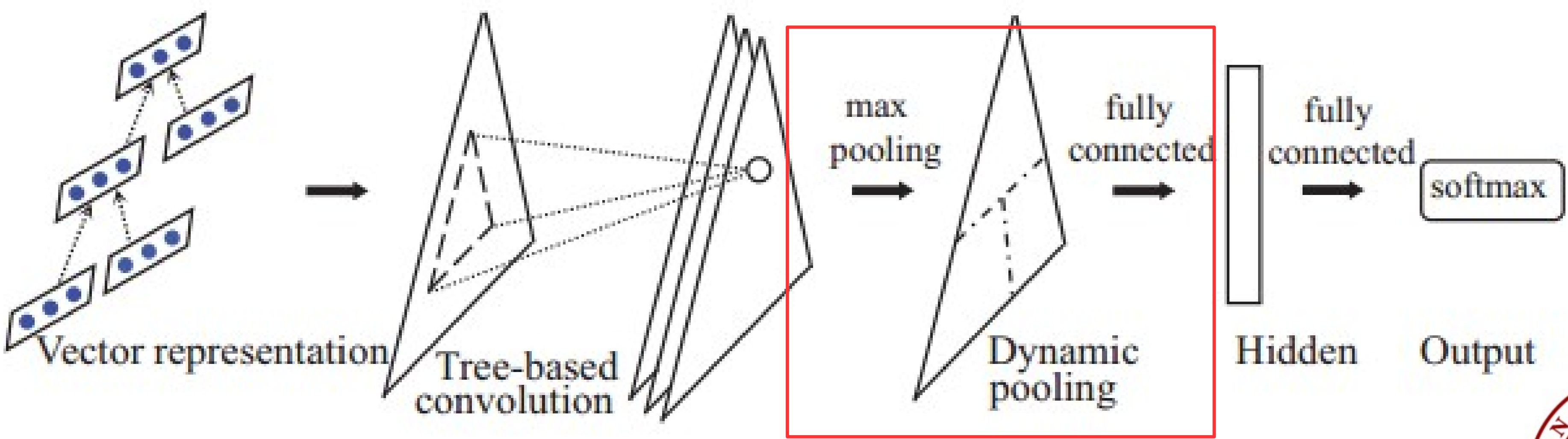
$$\boldsymbol{y} = f\left(\sum_{i=1}^{t} \boxed{W_i} \boldsymbol{x}_i + \boldsymbol{b}\right)$$



asTBCNN: abstract syntax tree

- How to aggregate information?
- Global pooling or 3-way pooling

$$\boxed{\boldsymbol{y}} = f\left(\sum_{i=1}^{t} W_i \cdot \boldsymbol{x}_i + \boldsymbol{b}\right)$$



Performance

- Sentiment analysis [15]

Group	Method	5-class accuracy	2-class accuracy	Reported in
Baseline	SVM	40.7	79.4	Socher et al. (2013)
	Naïve Bayes	41.0	81.8	Socher et al. (2013)
CNNs	1-layer convolution	37.4	77.1	Blunsom et al. (2014)
	Deep CNN	48.5	86.8	Blunsom et al. (2014)
	Non-static	48.0	87.2	Kim (2014)
	Multichannel	47.4	**88.1**	Kim (2014)
RNNs	Basic	43.2	82.4	Socher et al. (2013)
	Matrix-vector	44.4	82.9	Socher et al. (2013)
	Tensor	45.7	85.4	Socher et al. (2013)
	Tree LSTM (variant 1)	48.0	–	Zhu et al. (2015)
	Tree LSTM (variant 2)	51.0	88.0	Tai et al. (2015)
	Tree LSTM (variant 3)	49.9	88.0	Le and Zuidema (2015)
	Deep RNN	49.8	$86.6^\dagger$	Irsoy and Cardie (2014)
Recurrent	LSTM	45.8	86.7	Tai et al. (2015)
	bi-LSTM	49.1	86.8	Tai et al. (2015)
Vector	Word vector avg.	32.7	80.1	Socher et al. (2013)
	Paragraph vector	48.7	87.8	Le and Mikolov (2014)
TBCNNs	c-TBCNN	50.4	$86.8^\dagger$	Our implementation
	d-TBCNN	**51.4**	$87.9^\dagger$	Our implementation

Performance

- Question classification [15]

Method	Acc. (%)	Reported in
SVM 10k features + 60 rules	95.0	Silva et al. (2011)
CNN-non-static	93.6	Kim (2014)
CNN-mutlichannel	92.2	Kim (2014)
RNN	90.2	Zhao et al. (2015)
Deep-CNN	93.0	Blunsom et al. (2014)
Ada-CNN	92.4	Zhao et al. (2015)
c-TBCNN	94.8	Our implementation
d-TBCNN	**96.0**	Our implementation

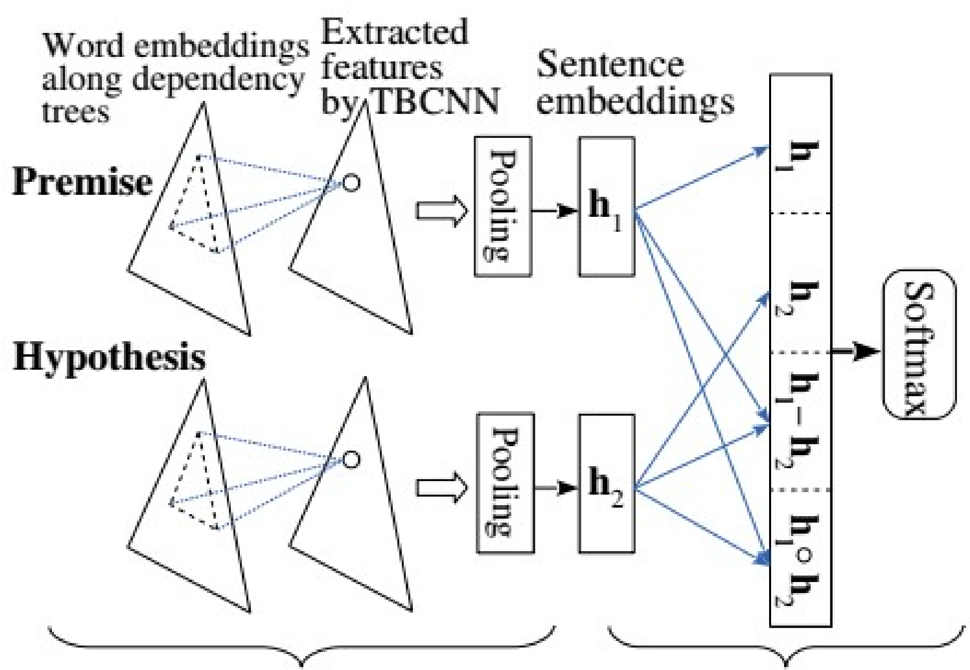
Performance

- Natural language inference [16]

  (entailment and contradiction recognition)

Model	Test acc. (%)	Matching complexity
Unlexicalized features[b]	50.4	$\mathcal{O}(1)$
Lexicalized features[b]	78.2	
Vector sum + MLP[b]	75.3	
Vanilla RNN + MLP[b]	72.2	
LSTM RNN + MLP[b]	77.6	
CNN + cat	77.0	
GRU w/ skip-thought pretraining[v]	81.4	
TBCNN-pair + cat	79.3	
TBCNN-pair + cat,∘,-	**82.1**	
Single-chain LSTM RNNs[r]	81.4	$\mathcal{O}(n)$
+ static attention[r]	**82.4**	
LSTM + word-by-word attention[r]	**83.5**	$\mathcal{O}(n^2)$

…



(a) Individual sentence modeling by TBCNN

(b) Combining two sentences by heuristics

Performance

- 104-label program classification [14]

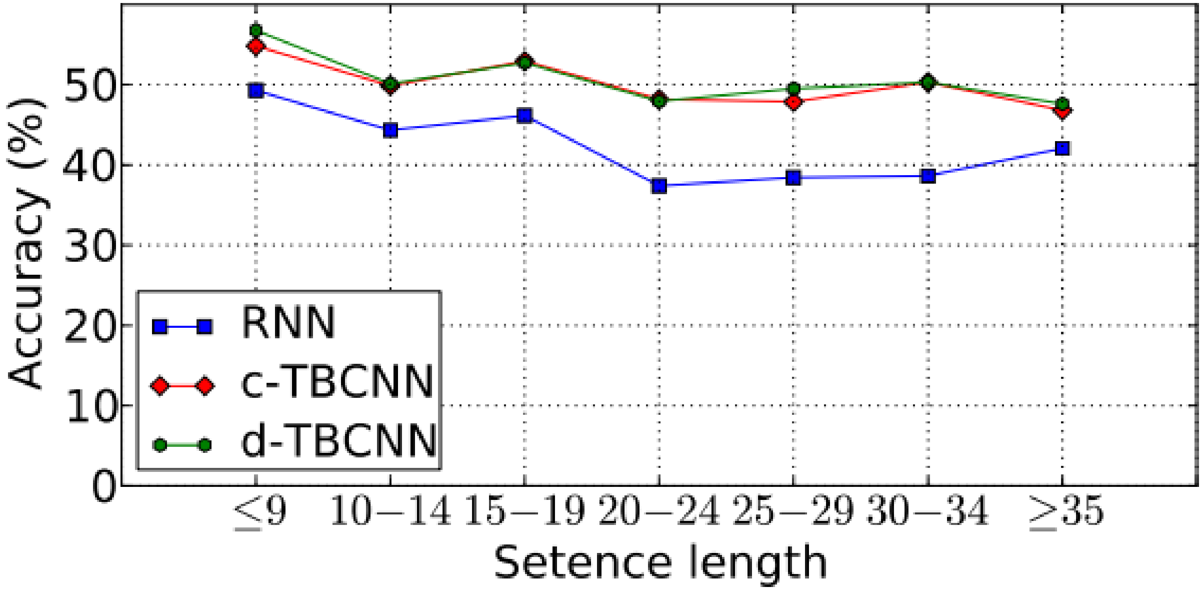
Group	Method	Test Accuracy (%)
Surface features	linear SVM+BoW	52.0
	RBF SVM+BoW	83.9
	linear SVM+BoT	72.5
	RBF SVM+BoT	88.2
NN-based approaches	DNN+BoW	76.0
	DNN+BoT	89.7
	Vector avg.	53.2
	RNN	84.8
Our method	TBCNN	**94.0**

Model Analysis

- TBCNN is not sensitive to pooling

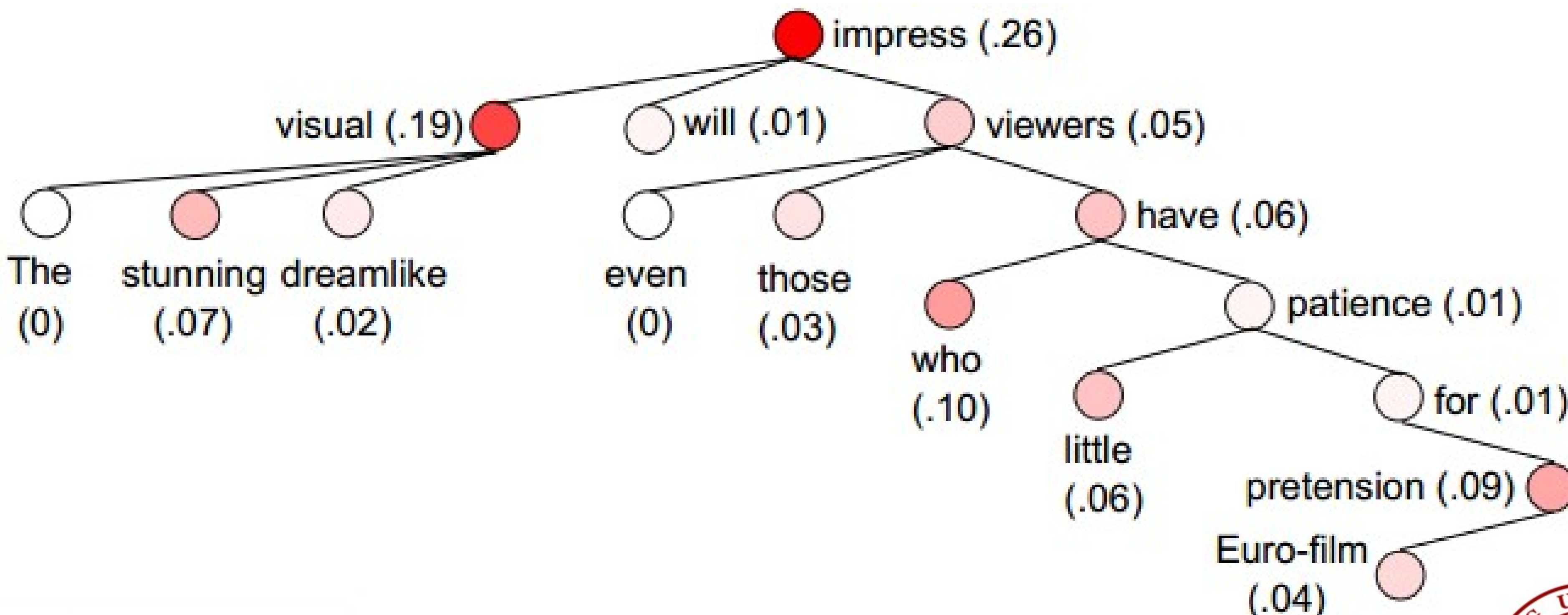
Model	Pooling method	5-class accuracy (%)
c-TBCNN	Global	48.48 ± 0.54
	3-slot	48.69 ± 0.40
d-TBCNN	Global	49.39 ± 0.24
	2-slot	49.94 ± 0.63

- TBCNN is less sensitive to length vis-a-vis recursive nets



Visualization

- TBCNN does mix information



Outline

- Introduction: Sentence Modeling
- Related Work: CNNs, RNNs, etc
- Tree-Based Convolution
- **Conclusion and Discussion**

Conclusion

- A new neural model: Tree-based convolution
- Applications
  - constituency trees, dependency tree, abstract syntax trees

		Way of information propagation	
		Iterative	Sliding
Structure	Flat	Recurrent	Convolution
	Tree	Recursive	Tree-base convolution

Discussion

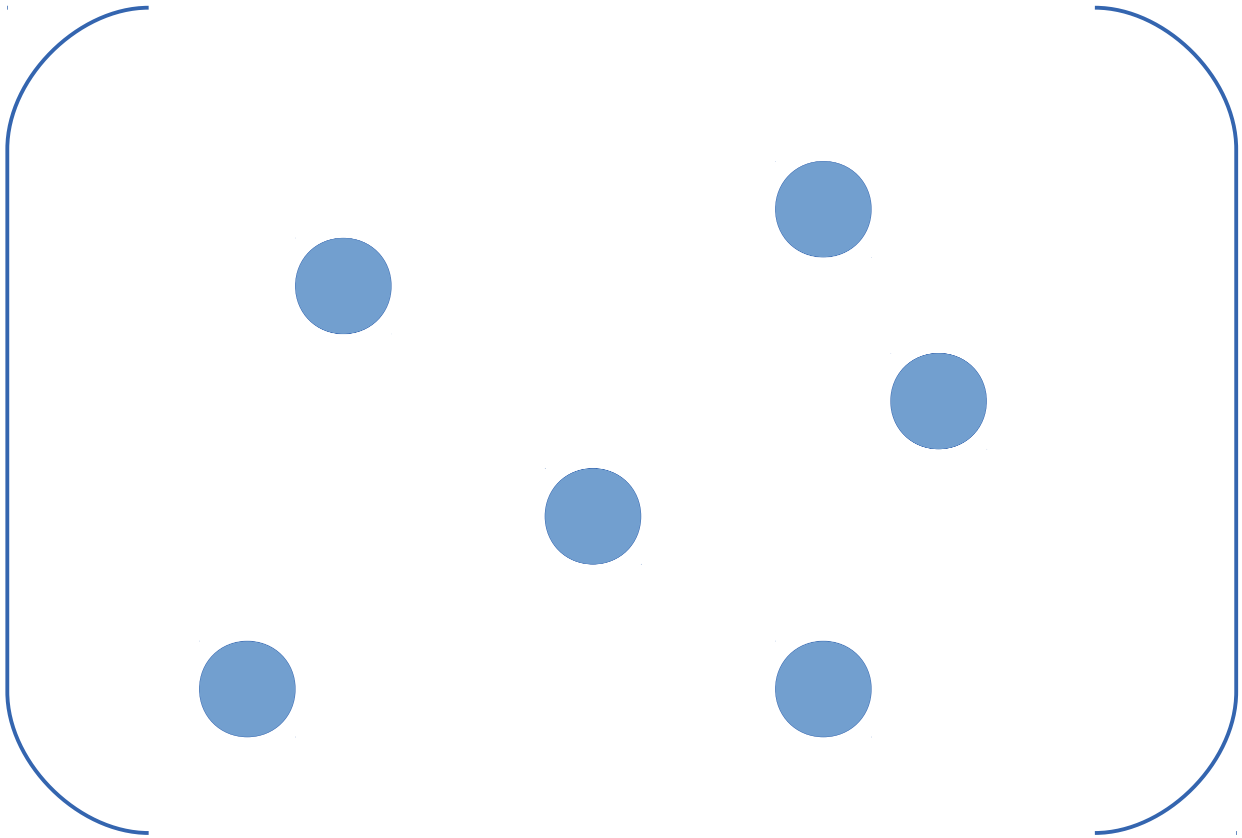
- Is it possible for a neural network to “automatically” focus on relevant (e.g., tree-dependent) information by some attention mechanisms?

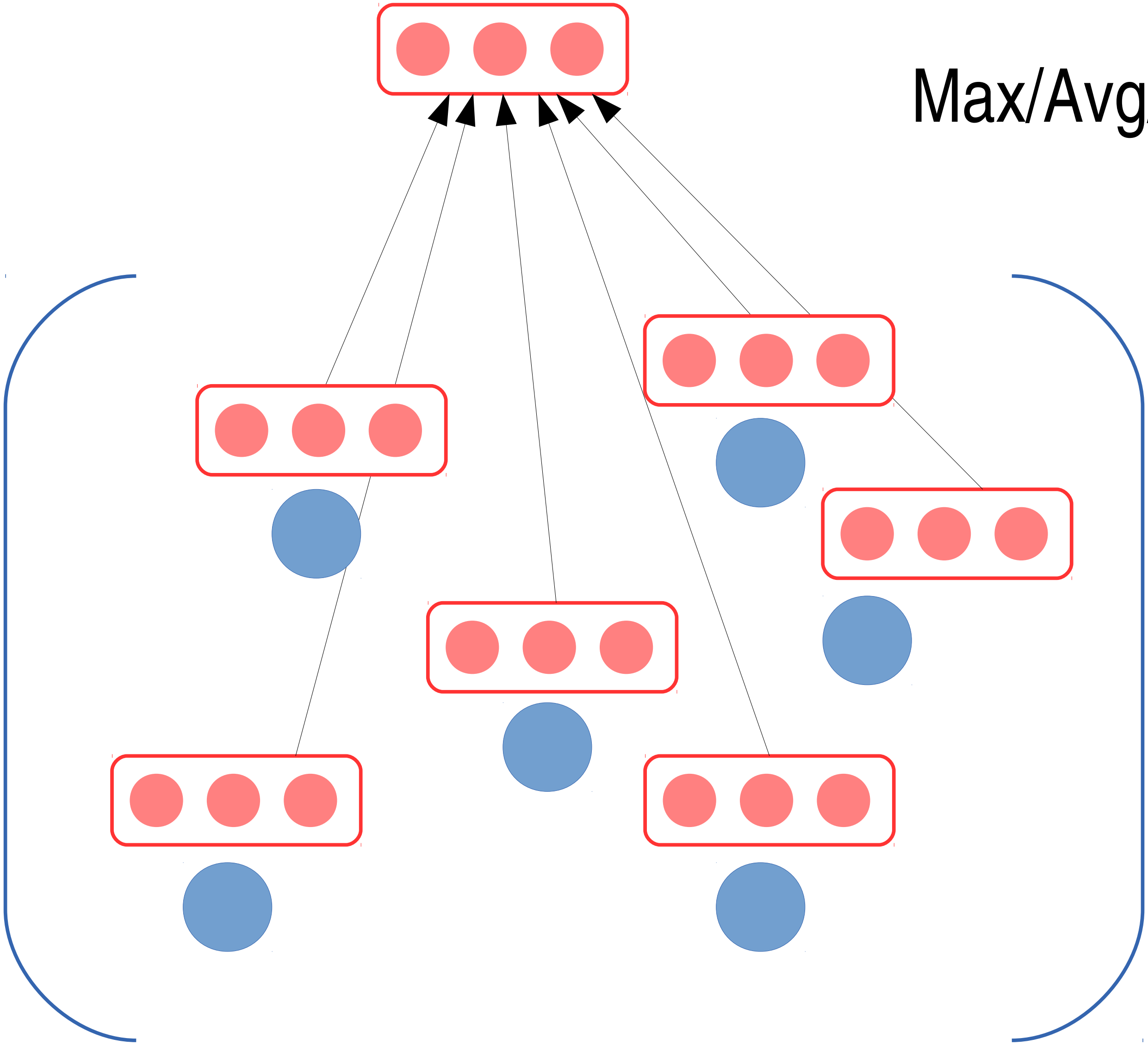
- Yes, but...

Discussion

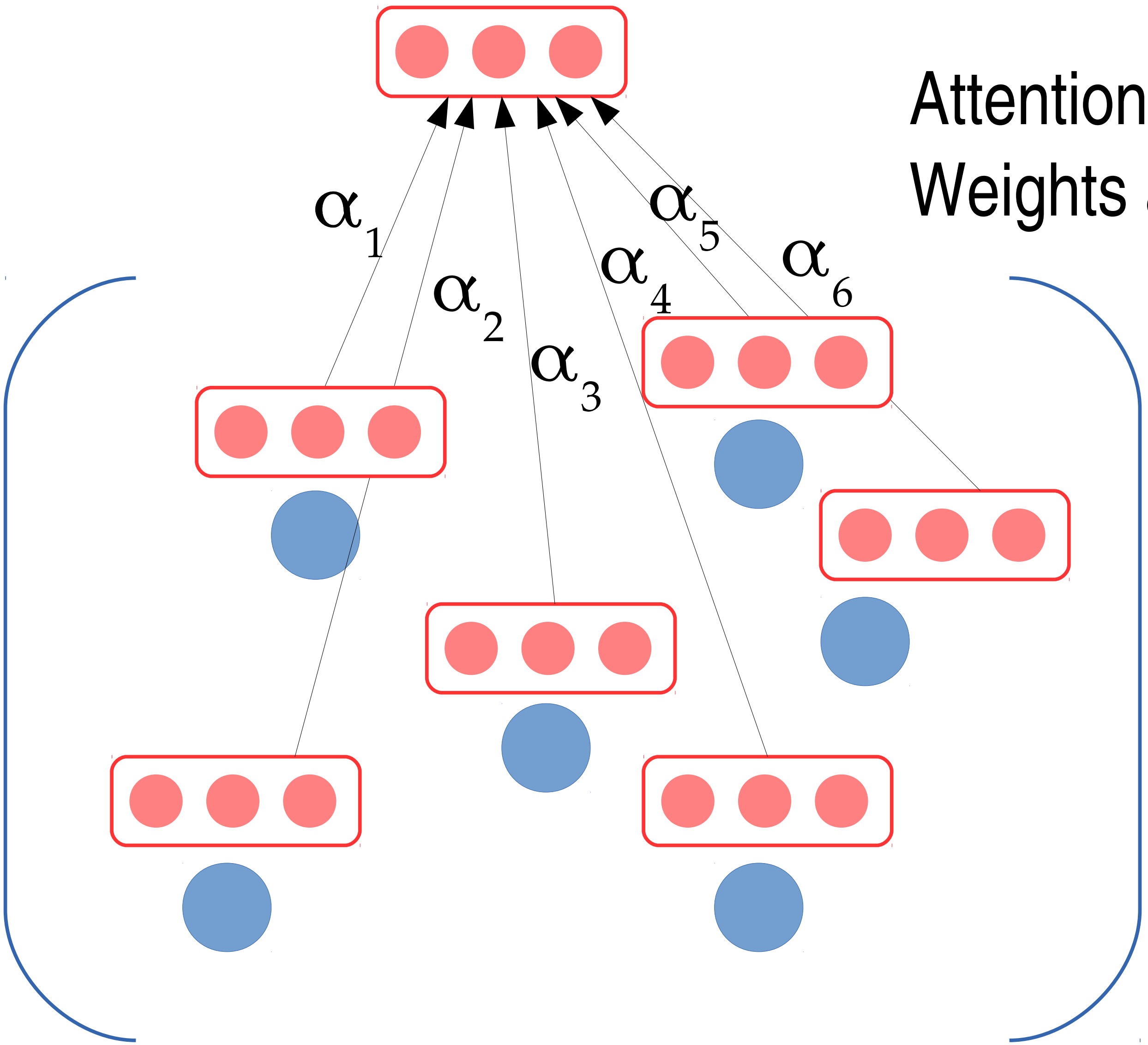
Challenge of end-to-end learning:

	avg	sum	max	attention	argmax
Differentiability	😋	😋	😋	😋	🙀
Supervision	😐	😐	😐	😐	🙀
Scalability	🙀	🙀	🙀	🙀	😋





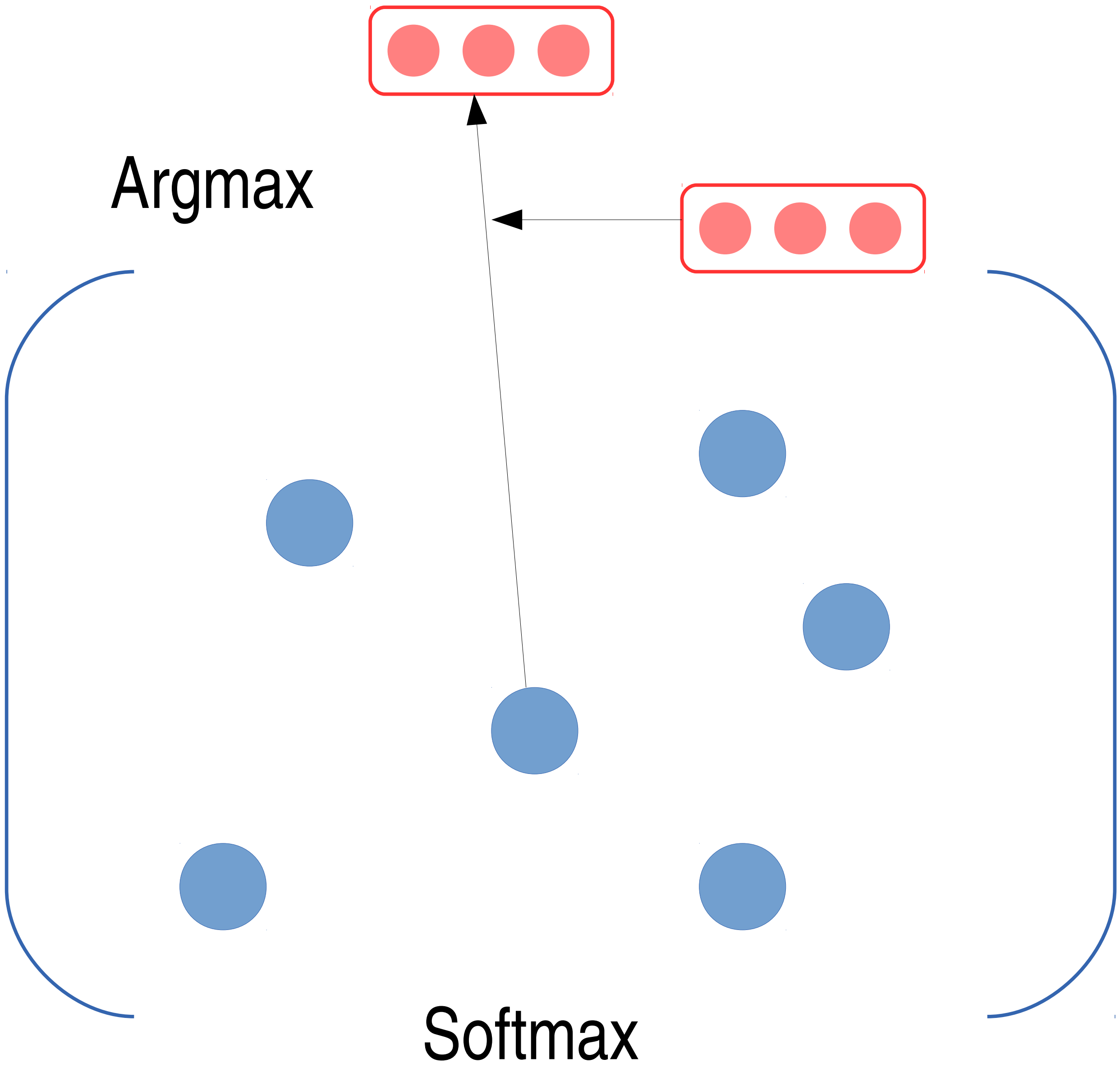
Max/Avg/Sum pooling



Attention=Weighted avg

Weights are self-adaptive.

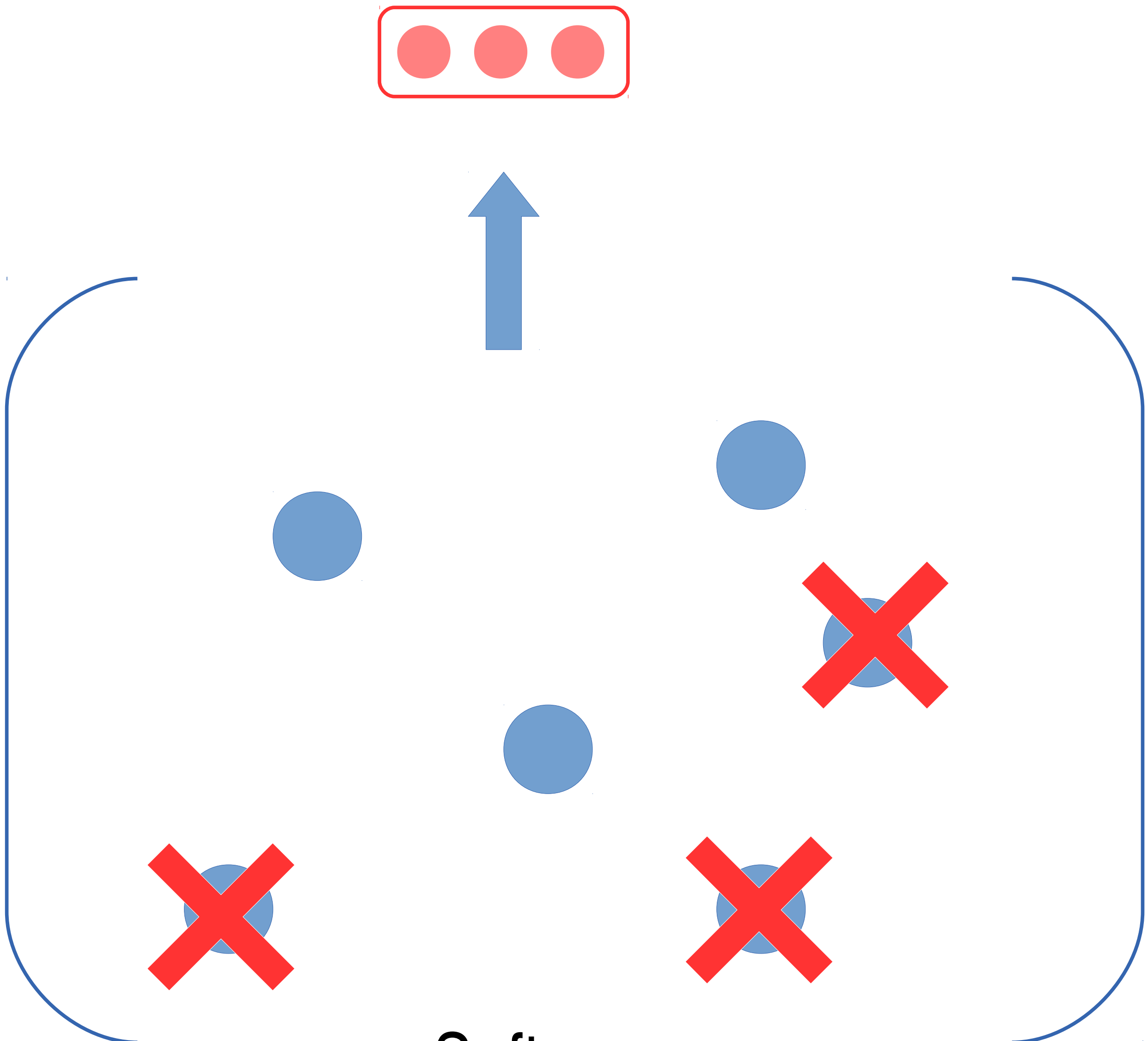
Recall sentence generation



Intuition

- Using external information to guide an NN instead of designing end-to-end machines
  - Better performance in short term
  - May or may not conform to the goal of AI, depending on how strict the external information is

	Hard mechanism
Differentiability	😋
Supervision	😋
Scalability	😋



Softmax

Recall sentence generation

Hard mechanism non end-to-end

Thanks for listening!

Q&A?

References

[1] Chen K, Wang J, Chen LC, Gao H, Xu W, Nevatia R. ABC-CNN: An Attention Based Convolutional Neural Network for Visual Question Answering. arXiv:1511.05960, 2015.

[2] Rui Yan, Yiping Song, Hua Wu. Learning to Respond with Deep Neural Networks for Retrieval based Human-Computer Conversation System. In SIGIR, 2016.

[3] Liu Y, Li S, Zhang X, Sui Z. Implicit discourse relation classification via multi-task neural networks. In AAAI, 2016.

[4] Cao Z, Wei F, Dong L, Li S, Zhou M. Ranking with Recursive Neural Networks and Its Application to Multi-Document Summarization. In AAAI, 2015.

[5] Pei W, Ge T, Chang B. An effective neural network model for graph-based dependency parsing. In ACL, 2015.

[6] Meng F, Lu Z, Wang M, Li H, Jiang W, Liu Q. Encoding source language with convolutional neural network for machine translation. In ACL, 2015.

[7] Qiu G, Liu B, Bu J, Chen C. Expanding Domain Sentiment Lexicon through Double Propagation. In IJCAI, 2009.

[8] Socher R., et al. Semi-supervised recursive autoencoders for predicting sentiment distributions. EMNLP, 2011.

[9] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[10] Le QV, Mikolov T. Distributed representations of sentences and documents. In ICML, 2014.

[11] Blunsom, Phil, Edward Grefenstette, and Nal Kalchbrenner. "A Convolutional Neural Network for Modelling Sentences." ACL, 2014.

[12] Socher, R, et al. "Semantic compositionality through recursive matrix-vector spaces." EMNLP-CoNLL, 2012.

[13] Socher, R, et al. "Recursive deep models for semantic compositionality over a sentiment treebank." EMNLP, 2013.

[14] Lili Mou, Ge Li, Lu Zhang, Tao Wang, Zhi Jin. "Convolutional neural networks over tree structures for programming language processing." In AAAI, pages 1287--1293, 2016.

[15] Lili Mou, Hao Peng, Ge Li, Yan Xu, Lu Zhang, Zhi Jin. "Discriminative neural sentence modeling by tree-based convolution." In EMNLP, pages 2315--2325, 2015.

[16] Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, Zhi Jin. "Natural language inference by tree-based convolution and heuristic matching." ACL(2), 2016.

[17] Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. JMLR, 2011.

[18] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng and Zhi Jin. "Classifying relations via long short term memory networks along shortest dependency paths." In EMNLP, 2015.

[19] Pinker, Steven. The Language Instinct: The New Science of Language and Mind. Penguin UK, 1995.

[20] Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. "Improved semantic representations from tree-structured long short-term memory networks." ACL, 2015

[21] Zhu, Xiaodan, Parinaz Sobihani, and Hongyu Guo. "Long short-term memory over recursive structures." ICML, 2015.

[22] Le, Phong, and Willem Zuidema. "Compositional distributional semantics with long short term memory." arXiv:1503.02510 (2015).