Linguistic Structure

Biryukov Valentin

- Statistics methods
- Convolutional neural network
- TreeLSTM

Statistics methods

TF-IDF statistic

TF == Term frequency:

IDF == Inverse document frequency

Variants of term frequency (TF) weight

weighting scheme	TF weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$1 + \log(f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Variants of inverse document frequency (IDF) weight

weighting scheme	IDF weight ($n_t = \lvert\{d \in D : t \in d\}\rvert$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log \left(1 + \frac{N}{n_t}\right)$
inverse document frequency max	$\log \left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

https://en.wikipedia.org/wiki/Tf–idf

Okapi BM25

Given a query Q, containing keywords q, the BM25 score of a document D is:

$$\text{score}(D, Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)},$$

usually, k=2.0, b=0.75

Convolutional neural network

CNN 2014

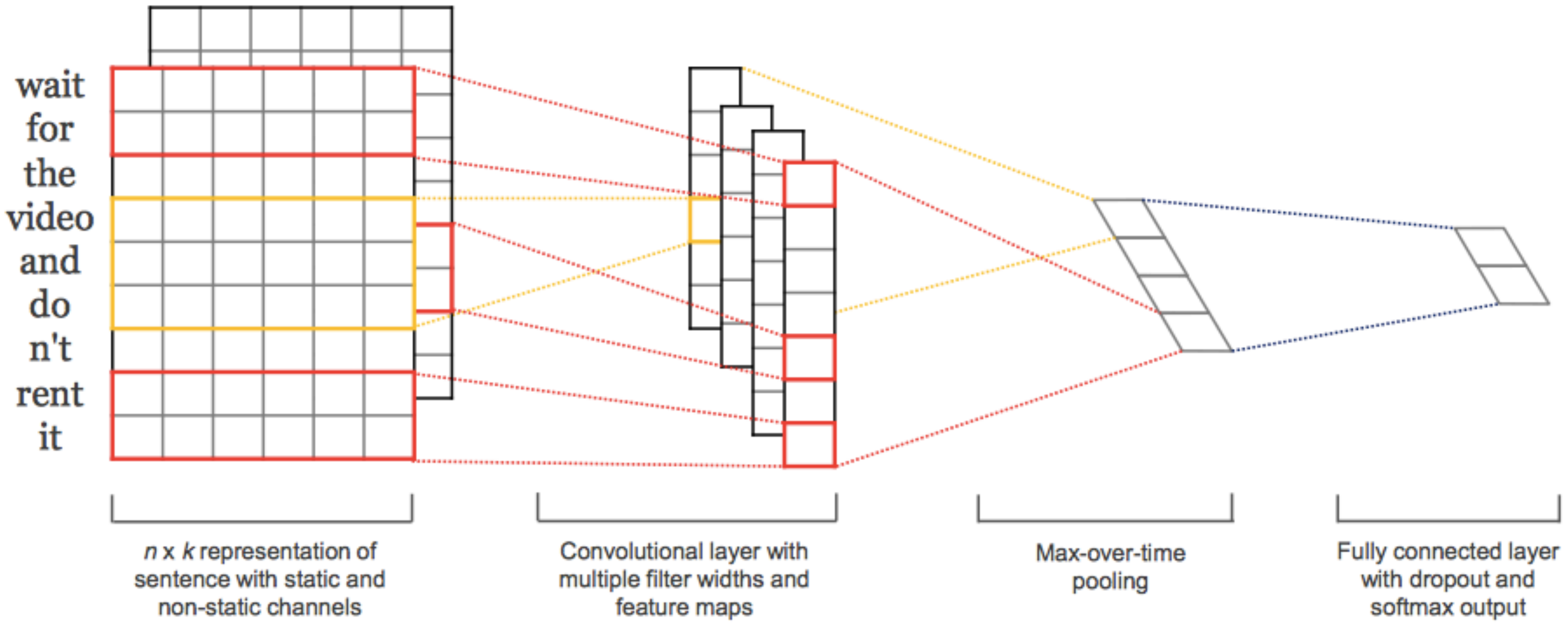
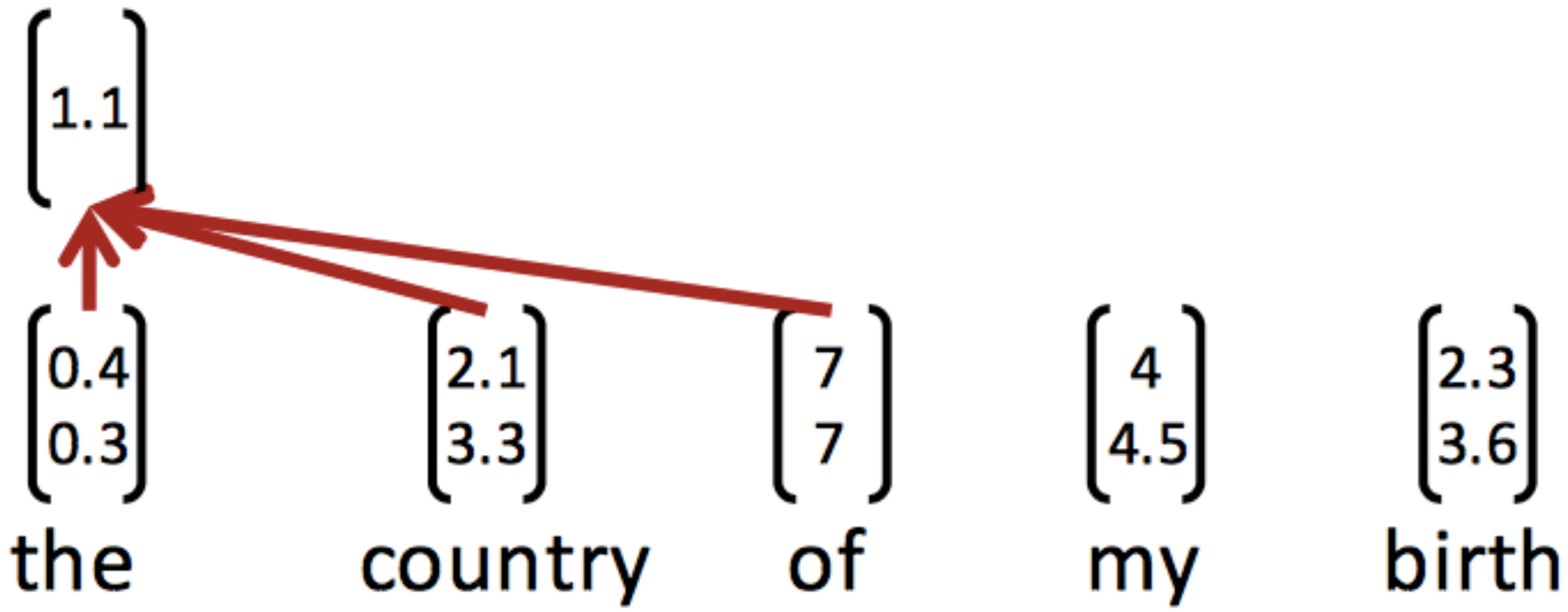


Figure 1: Model architecture with two channels for an example sentence.

http://www.aclweb.org/anthology/D14-1181

CNN: Layers

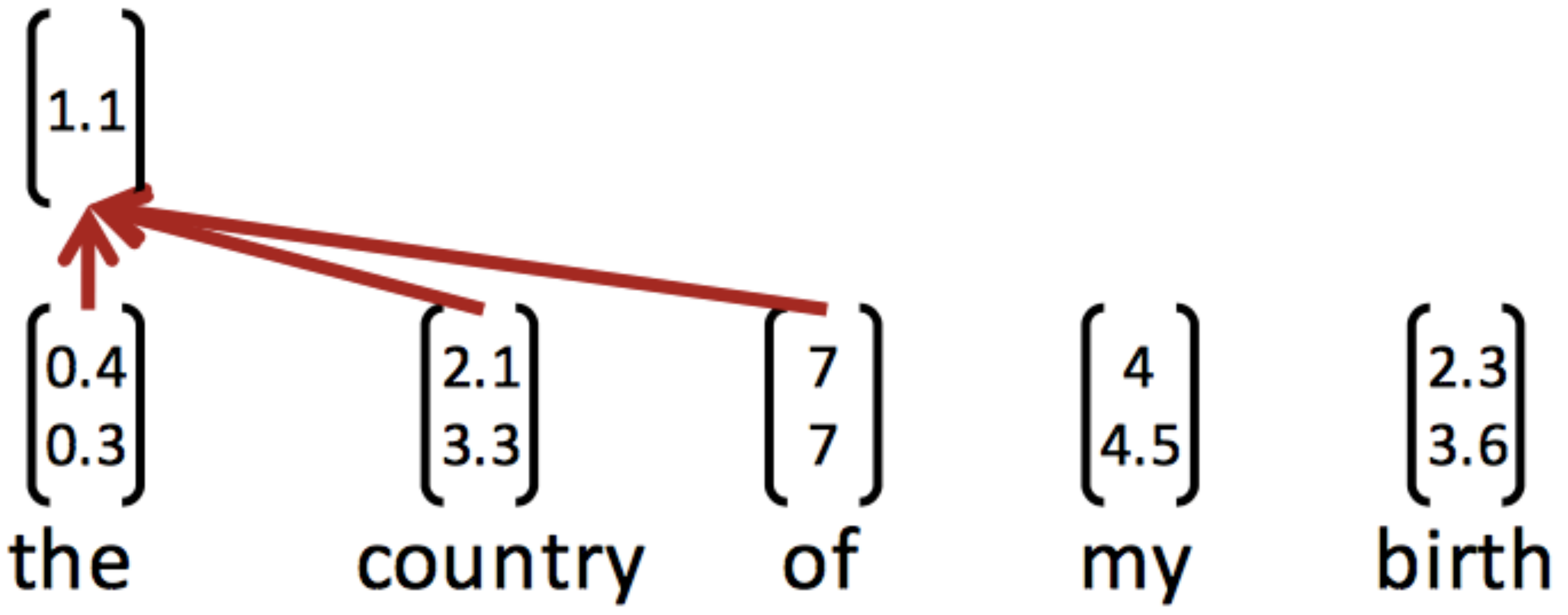
- A simple variant using one convolutional layer and **pooling**
- Based on Collobert and Weston (2011) and Kim (2014) "Convolutional Neural Networks for Sentence Classification"
- Word vectors: $\mathbf{x}_i \in \mathbb{R}^k$
- Sentence: $\mathbf{x}_{1:n} = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \ldots \oplus \mathbf{x}_n$ (vectors concatenated)
- Concatenation of words in range: $\mathbf{x}_{i:i+j}$
- Convolutional filter: $\mathbf{w} \in \mathbb{R}^{hk}$ (goes over window of h words)
- Could be 2 (as before) higher, e.g. 3:



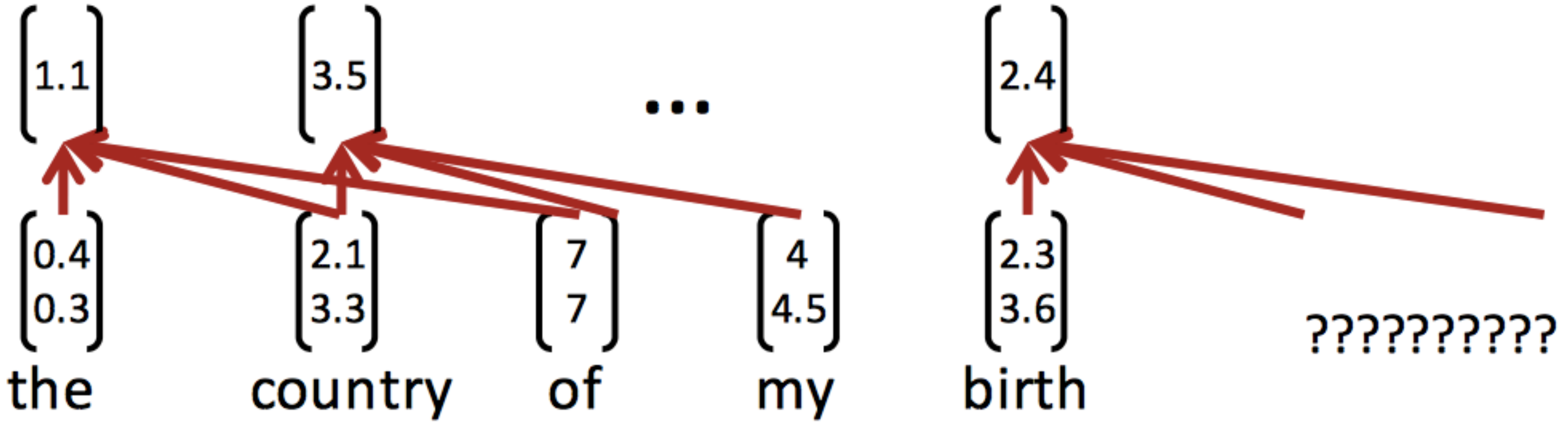
CNN: Layers

- Convolutional filter: $\mathbf{w} \in \mathbb{R}^{hk}$ (goes over window of h words)
- Note, filter is vector!
- Window size h could be 2 (as before) or higher, e.g. 3:
- To compute feature for CNN layer:

$$c_i = f(\mathbf{w}^T \mathbf{x}_{i:i+h-1} + b)$$



- Filter w is applied to all possible windows (concatenated vectors)
- Sentence: $\mathbf{x}_{1:n} = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \ldots \oplus \mathbf{x}_n$
- All possible windows of length h: $\{\mathbf{x}_{1:h}, \mathbf{x}_{2:h+1}, \ldots, \mathbf{x}_{n-h+1:n}\}$
- Result is a feature map: $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$



CNN 2014 Results

Table 2: Results of our CNN models against other methods
RAE: Recursive Autoencoders with pre-trained word vectors from Wikipedia (Socher et al., 2011).
MV-RNN: Matrix-Vector Recursive Neural Network with parse trees (Socher et al., 2012).
RNTN: Recursive Neural Tensor Network with tensor-basec feature function and parse trees (Socher et al., 2013).
DCNN: Dynamic Convolutional Neural Network with k-max pooling (Kalchbrenner et al., 2014). Paragraph-Vec: Logisti regres- sion on top of paragraph vectors (Le and Mikolov, 2014).
CCAE: Combinatorial Category Autoencoders with combinatorial category grammar operators (Hermann and Blunsom, 2013). Sent-Parser: Sentiment analysis-specific parser (Dong et al., 2014).
NBSVM, MNB: Naive Bayes SVM and Multinomial Naive Bayes with uni-bigrams from Wang and Manning (2012).
G-Dropout, F-Dropout: Gaussian Dropout and Fast Dropou from Wang and Manning (2013).
Tree-CRF: Dependency tree with Conditional Random Fields (Nakagawa et al., 2010).
CRF-PR: Conditional Random Fields with Posterior Regularization (Yang and Cardie, 2014).
SVM_S : SVM with uni-bi-trigrams, wh word, head word, POS, parser, hypernyms, and 60 hand-coded rules as features from Silva et al. (2011)

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	**89.6**
CNN-non-static	**81.5**	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	–	–	–	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	–	–	–	–
RNTN (Socher et al., 2013)	–	45.7	85.4	–	–	–	–
DCNN (Kalchbrenner et al., 2014)	–	48.5	86.8	–	93.0	–	–
Paragraph-Vec (Le and Mikolov, 2014)	–	**48.7**	87.8	–	–	–	–
CCAE (Hermann and Blunsom, 2013)	77.8	–	–	–	–	–	87.2
Sent-Parser (Dong et al., 2014)	79.5	–	–	–	–	–	86.3
NBSVM (Wang and Manning, 2012)	79.4	–	–	93.2	–	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	–	–	**93.6**	–	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	–	–	93.4	–	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	–	–	**93.6**	–	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	–	–	–	–	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	–	–	–	–	–	82.7	–
SVM_S (Silva et al., 2011)	–	–	–	–	**95.0**	–	–

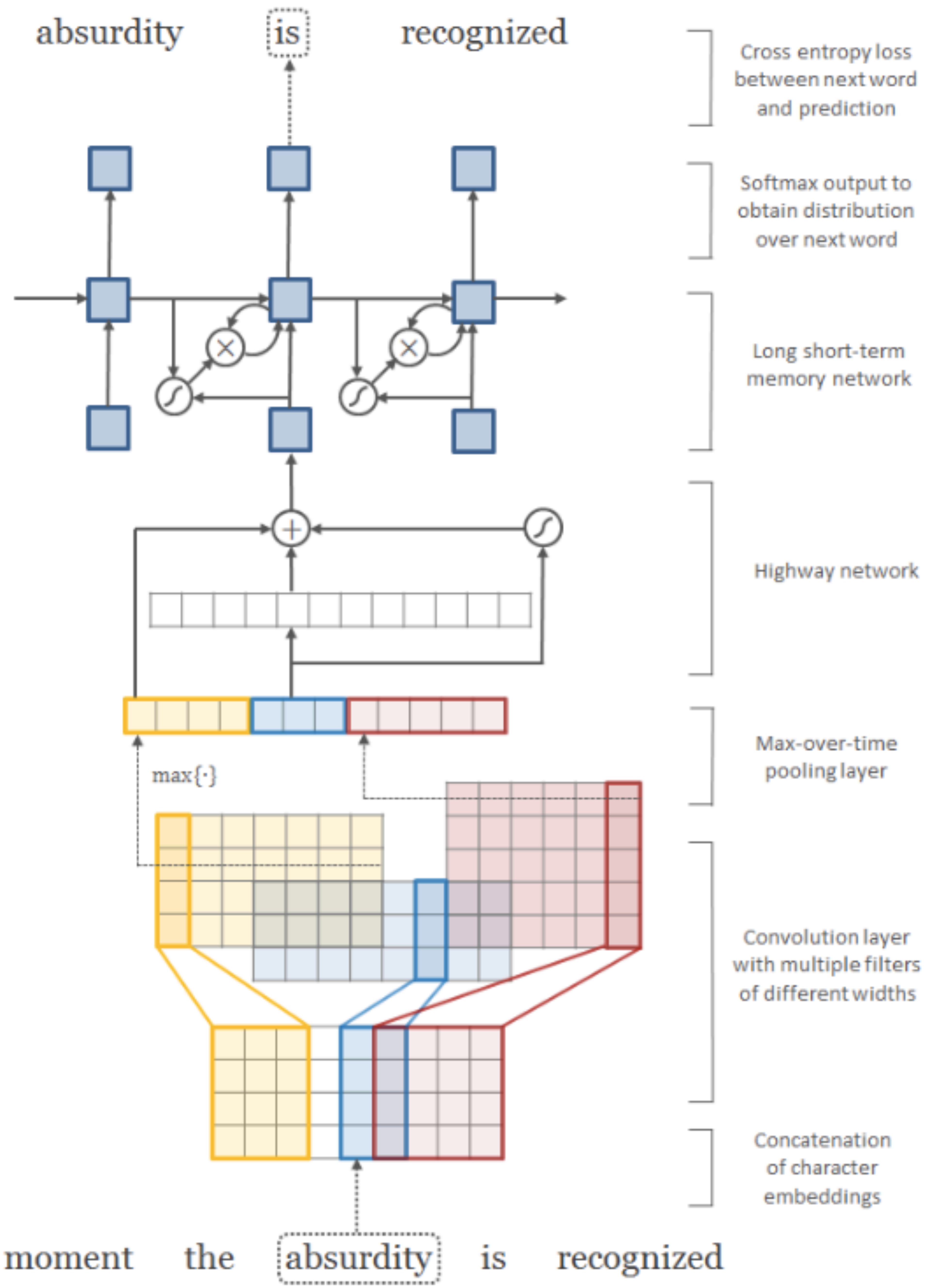
CNN 2014 Results

	Most Similar Words for	
	Static Channel	Non-static Channel
bad	*good* *terrible* *horrible* *lousy*	*terrible* *horrible* *lousy* *stupid*
good	*great* *bad* *terrific* *decent*	*nice* *decent* *solid* *terrific*
n't	*os* *ca* *ireland* *wo*	*not* *never* *nothing* *neither*
!	*2,500* *entire* *jez* *changer*	*2,500* *lush* *beautiful* *terrific*
,	*decasia* *abysmally* *demise* *valiant*	*but* *dragon* *a* *and*

Next idea: lets add LSTM! (2015)

	In Vocabulary					Out-of-Vocabulary		
	while	*his*	*you*	*richard*	*trading*	*computer-aided*	*misinformed*	*looooook*
LSTM-Word	*although* *letting* *though* *minute*	*your* *her* *my* *their*	*conservatives* *we* *guys* *i*	*jonathan* *robert* *neil* *nancy*	*advertised* *advertising* *turnover* *turnover*	– – – –	– – – –	– – – –
LSTM-Char (before highway)	*chile* *whole* *meanwhile* *white*	*this* *hhs* *is* *has*	*your* *young* *four* *youth*	*hard* *rich* *richer* *richter*	*heading* *training* *reading* *leading*	*computer-guided* *computerized* *disk-drive* *computer*	*informed* *performed* *transformed* *inform*	*look* *cook* *looks* *shook*
LSTM-Char (after highway)	*meanwhile* *whole* *though* *nevertheless*	*hhs* *this* *their* *your*	*we* *your* *doug* *i*	*eduard* *gerard* *edward* *carl*	*trade* *training* *traded* *trader*	*computer-guided* *computer-driven* *computerized* *computer*	*informed* *performed* *outperformed* *transformed*	*look* *looks* *looked* *looking*

Table 6: Nearest neighbor words (based on cosine similarity) of word representations from the large word-level and character-level (before and after highway layers) models trained on the PTB. Last three words are OOV words, and therefore they do not have representations in the word-level model.



https://arxiv.org/pdf/1508.06615.pdf

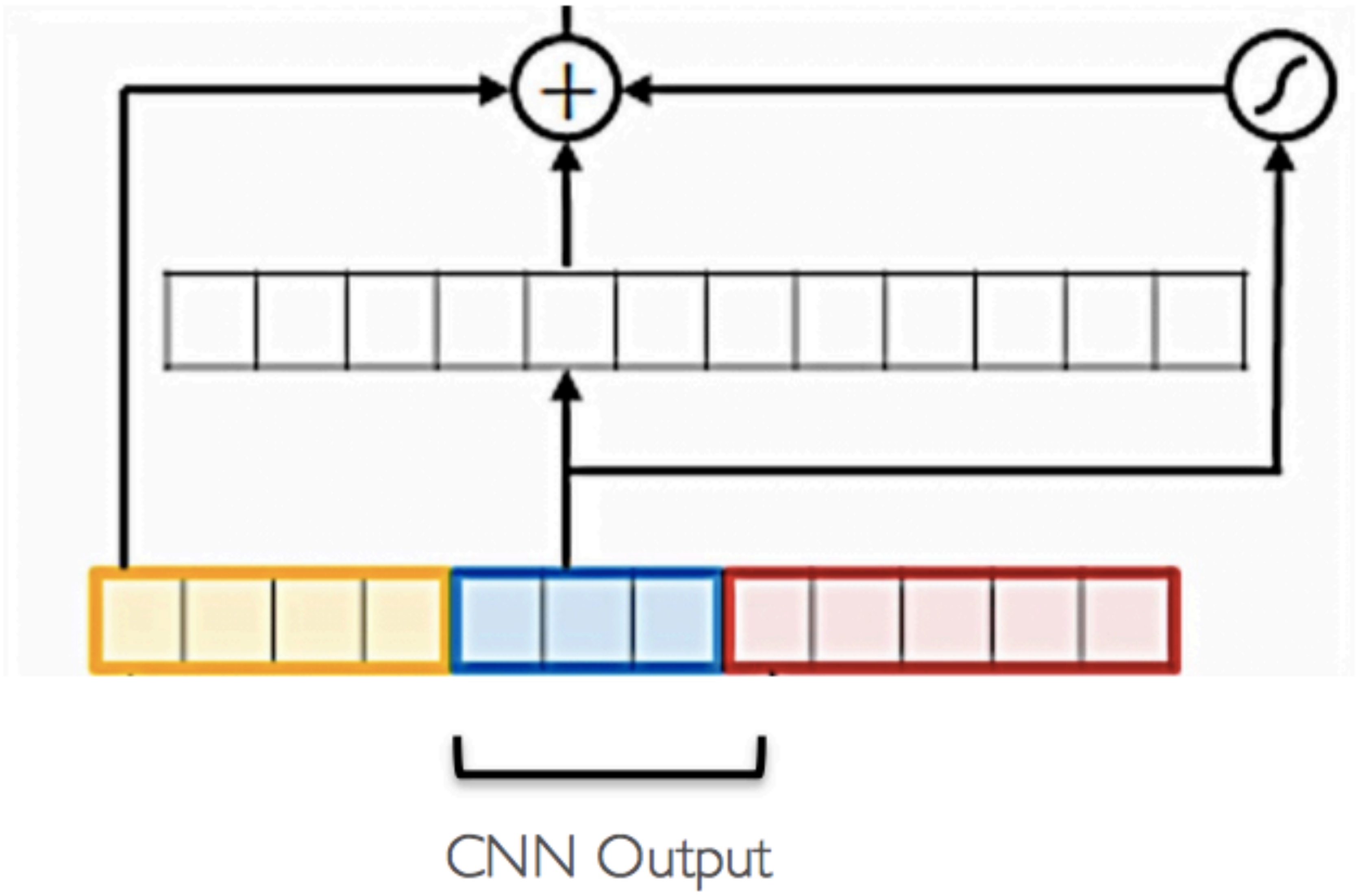
Highway Network (Srivastava et al. 2015)

- Model *n*-gram interactions.
- Apply transformation while carrying over
- Functions akin to an LSTM memory cell.

$$\mathbf{t} = \sigma(\mathbf{W}_T \mathbf{y} + \mathbf{b}_T)$$

$$\mathbf{z} = \mathbf{t} \odot g(\mathbf{W}_H \mathbf{y} + \mathbf{b}_H) + (\mathbf{1} - \mathbf{t}) \odot \mathbf{y}$$

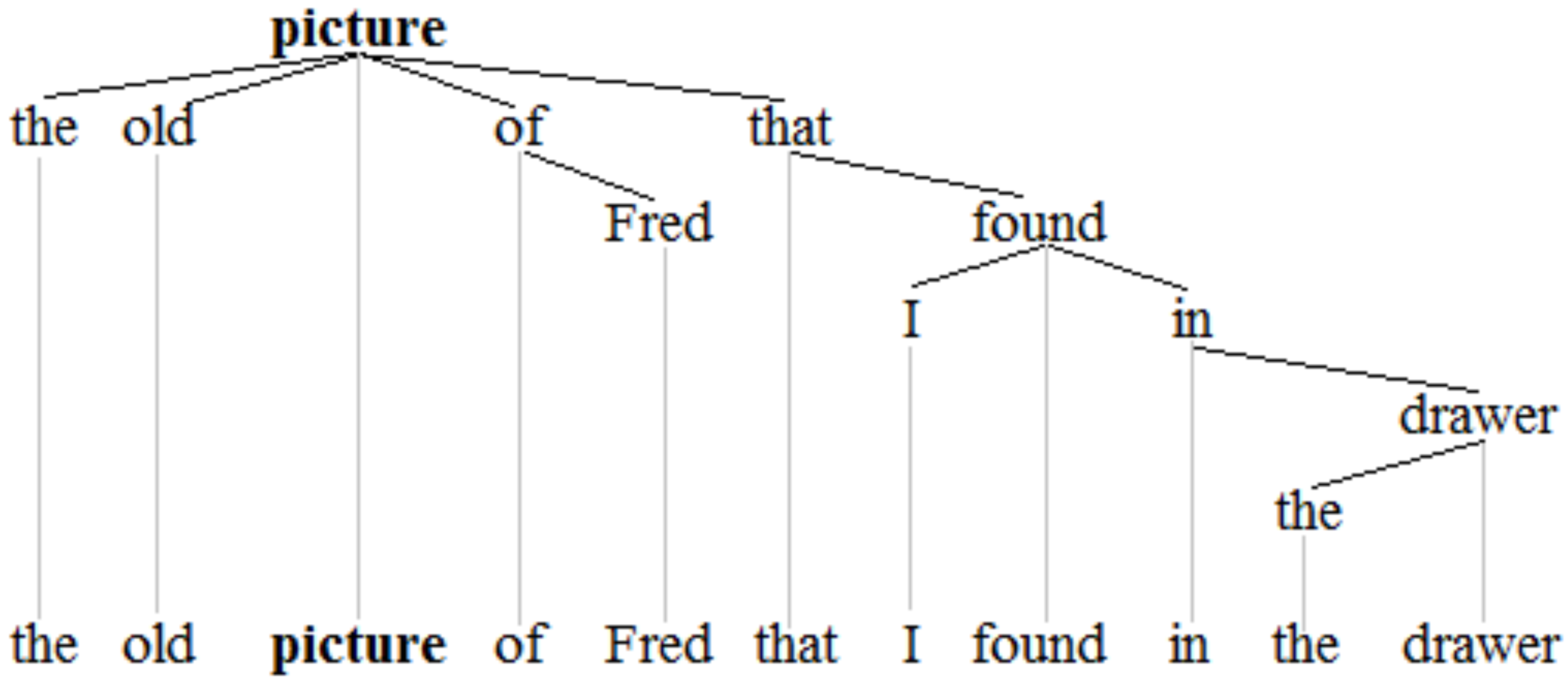
Transform Gate   Input   Carry Gate



LSTM TREE

Noun phrase

CP
C
TP
NP
T'
A
N
T
VP
маленький
мальчик
V
NP
кушал
N
PP
суп
P
NP
за
N
столом



picture
the
old
of
that
Fred
found
I
in
drawer
the
the old picture of Fred that I found in the drawer

S stands for sentence, the top-level structure.

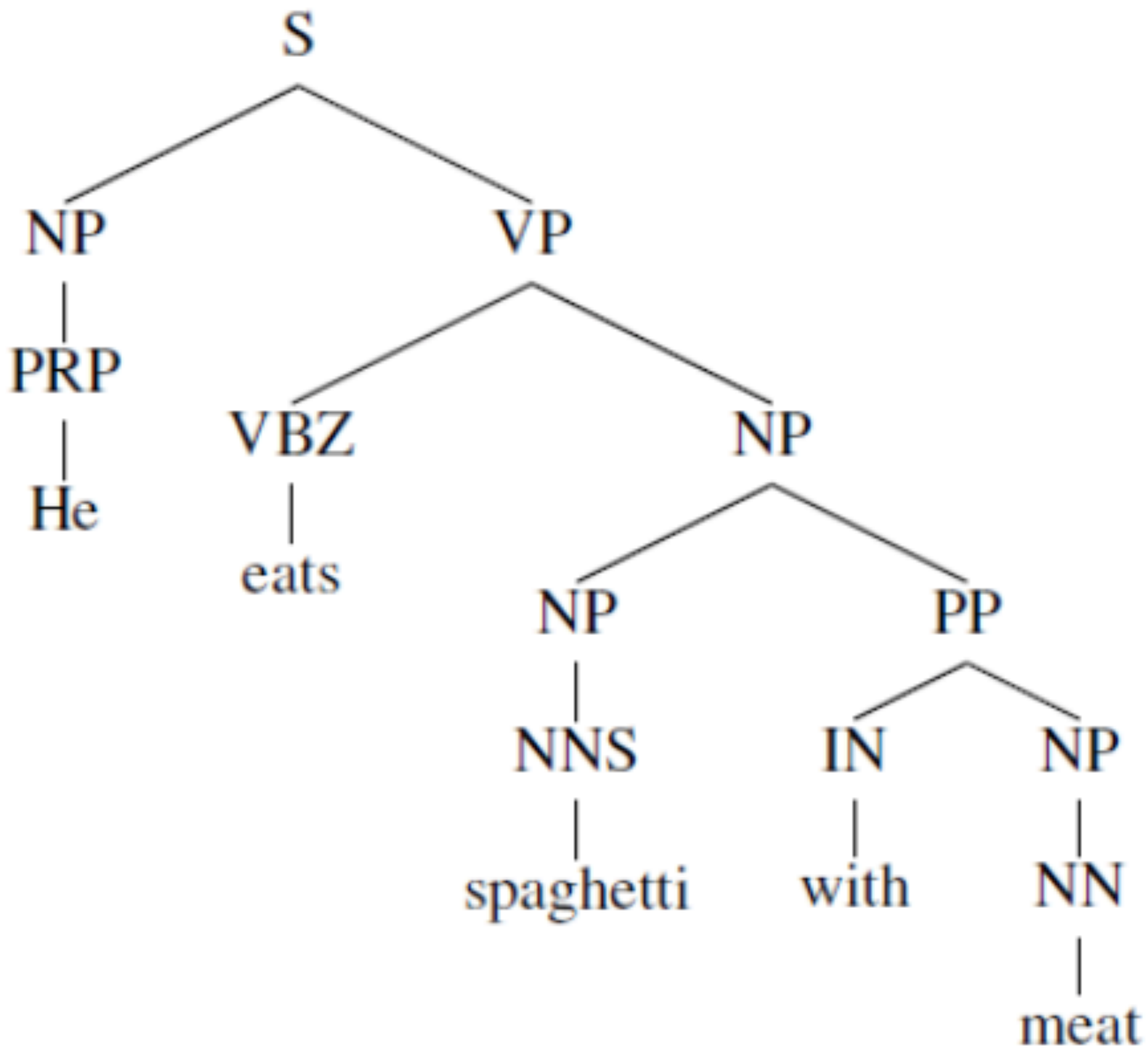
NP stands for noun phrase including the subject of the sentence and the object of the sentence.

VP stands for verb phrase, which serves as the predicate.

V stands for verb.

D stands for determiner, such as the definite article "the"

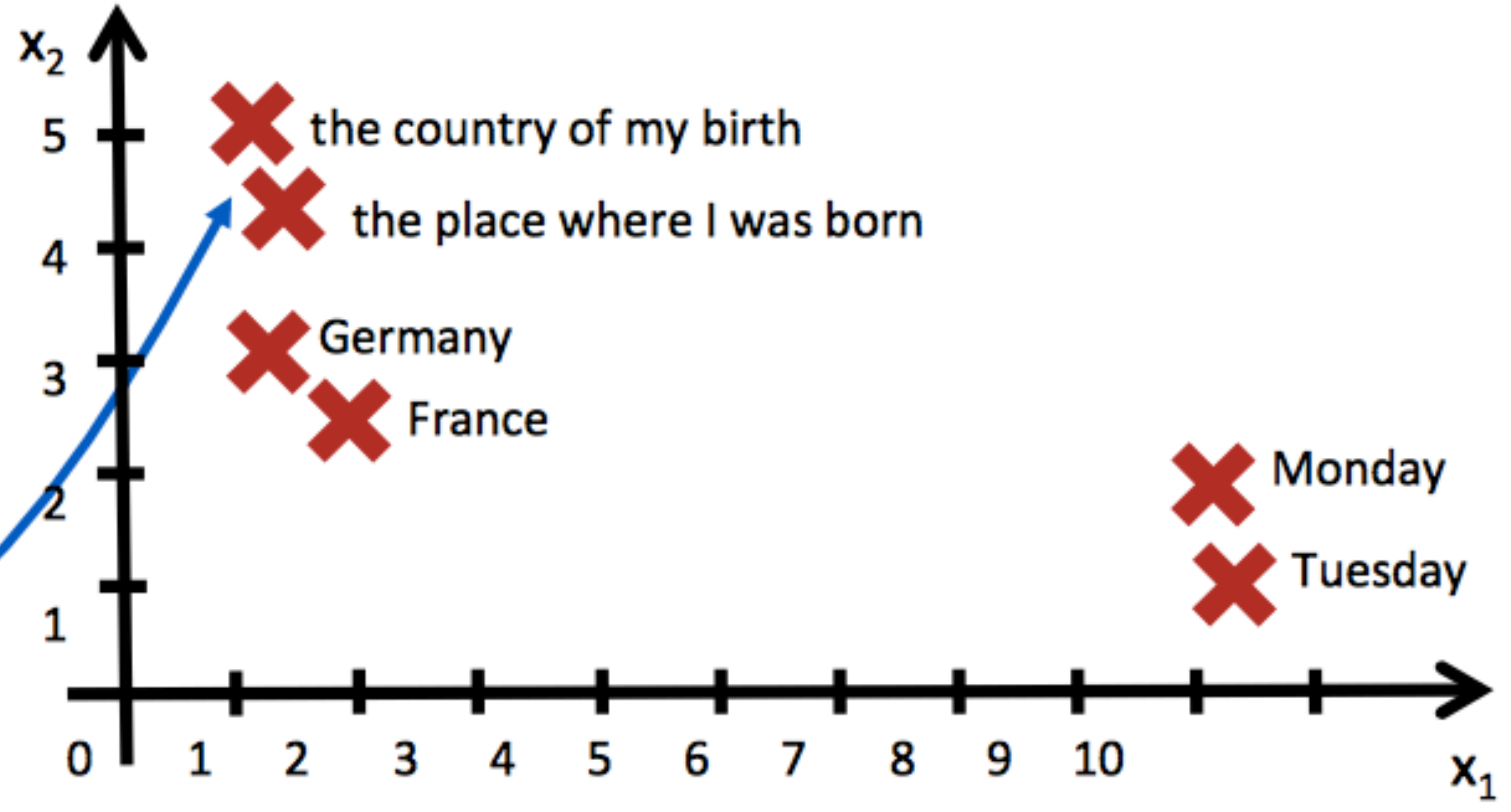
N stands for noun

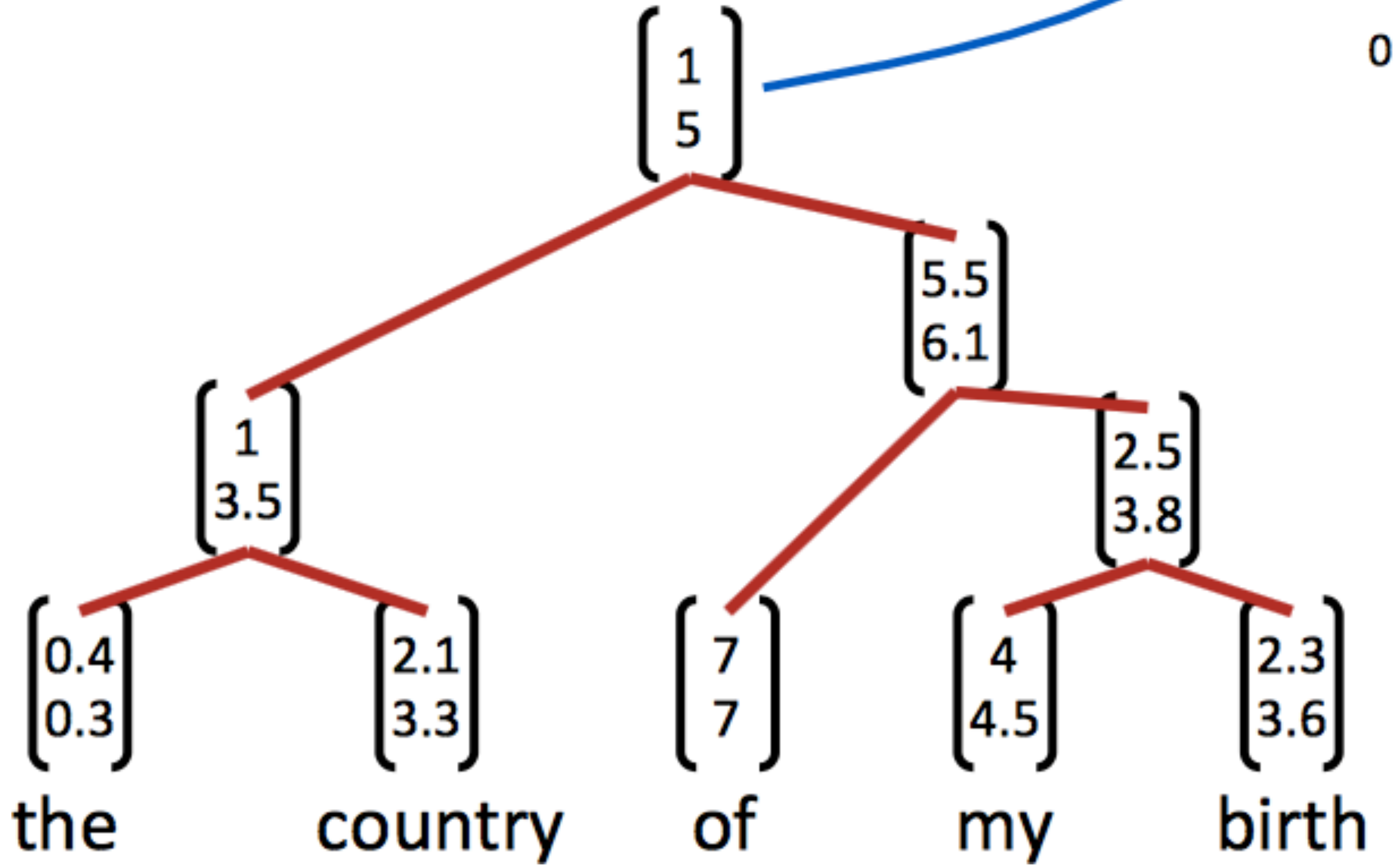


https://nlp.stanford.edu/pubs/SocherBauerManningNg_ACL2013.pdf

Use principle of compositionality

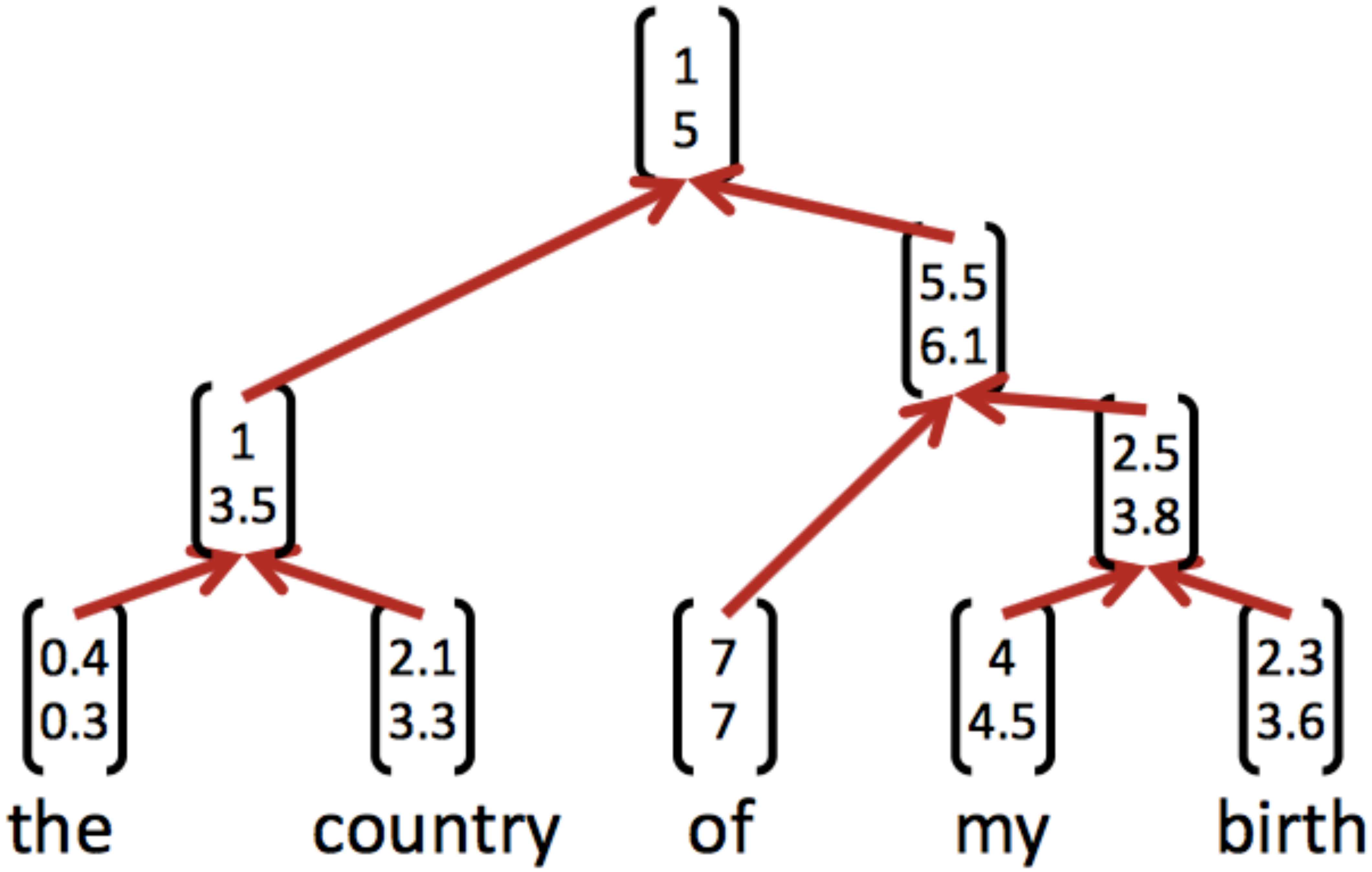
The meaning (vector) of a sentence is determined by
(1) the meanings of its words and
(2) the rules that combine them.

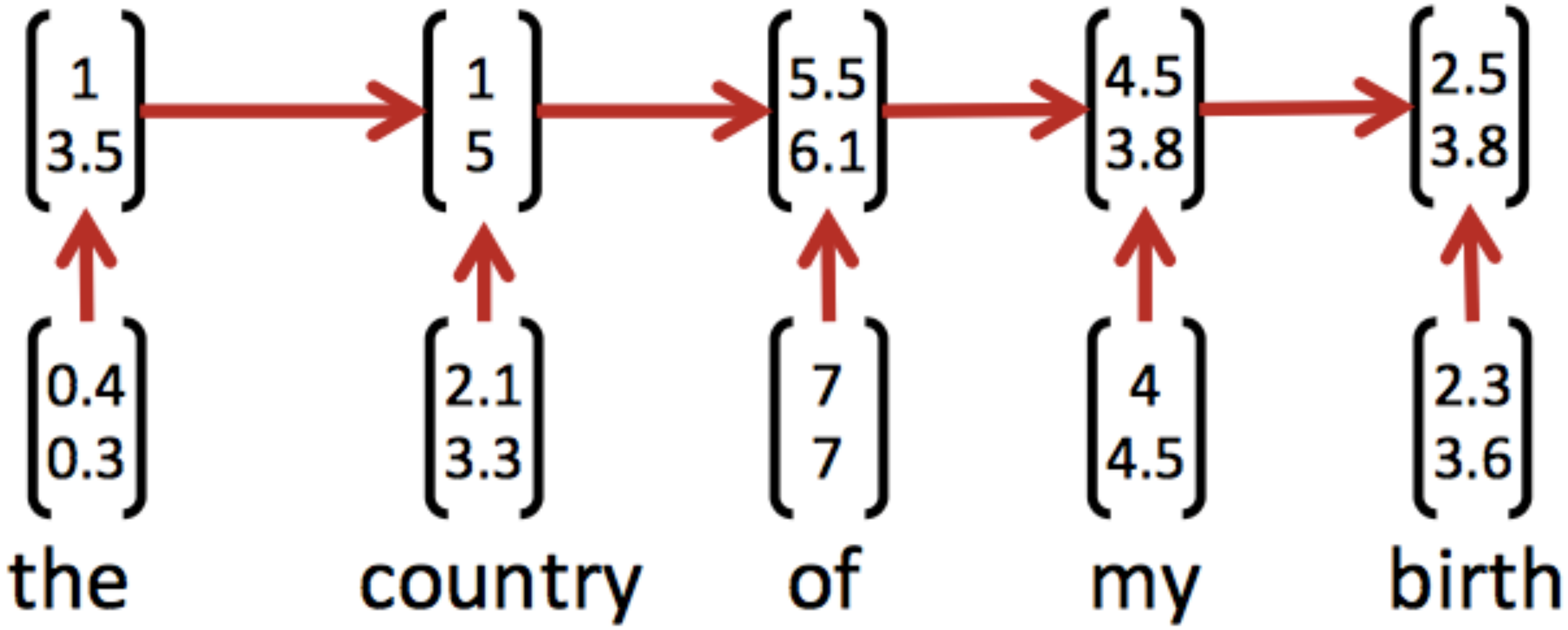




Models in this section can jointly learn parse trees and compositional vector representations

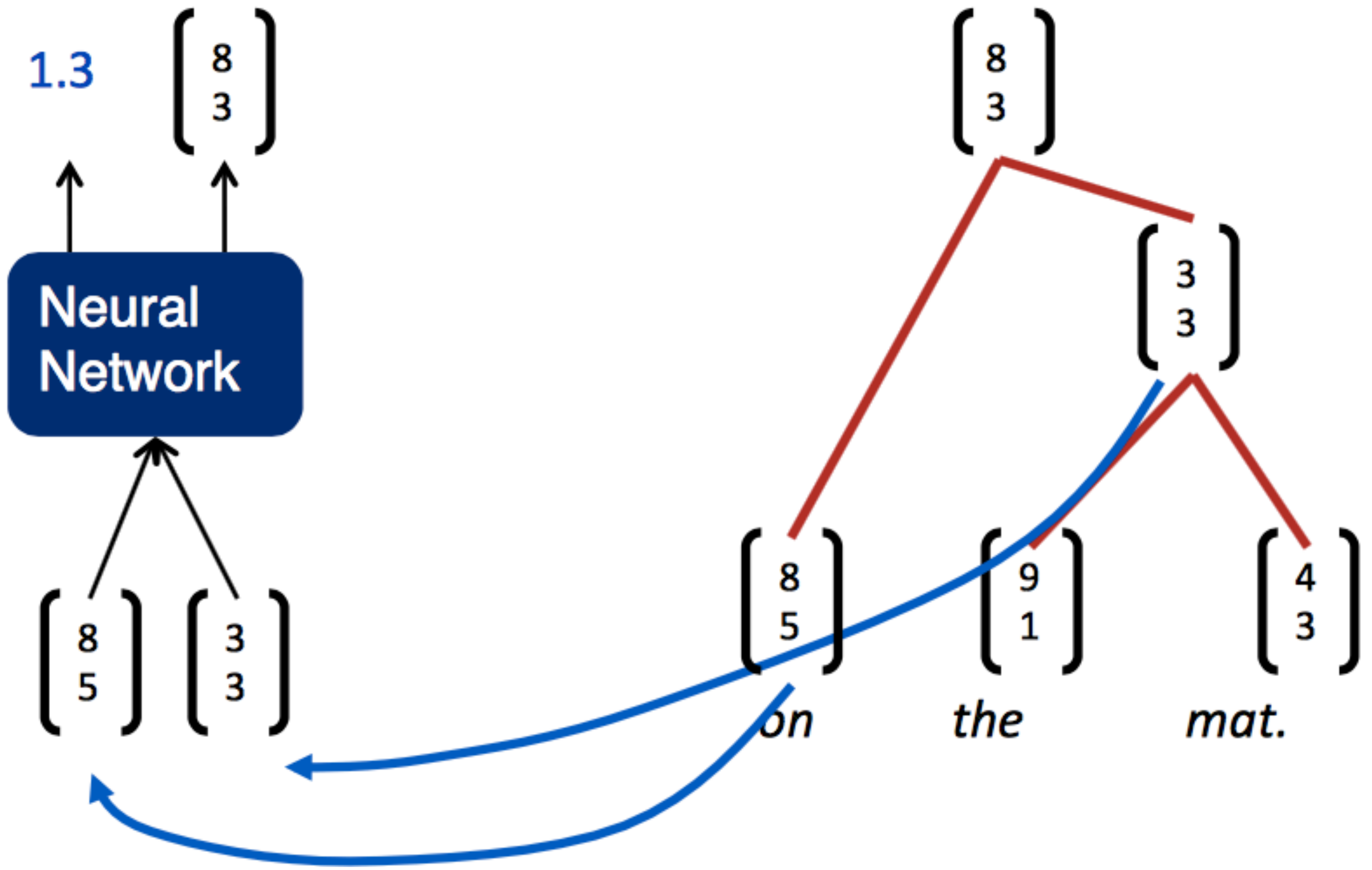
Recursive vs. recurrent

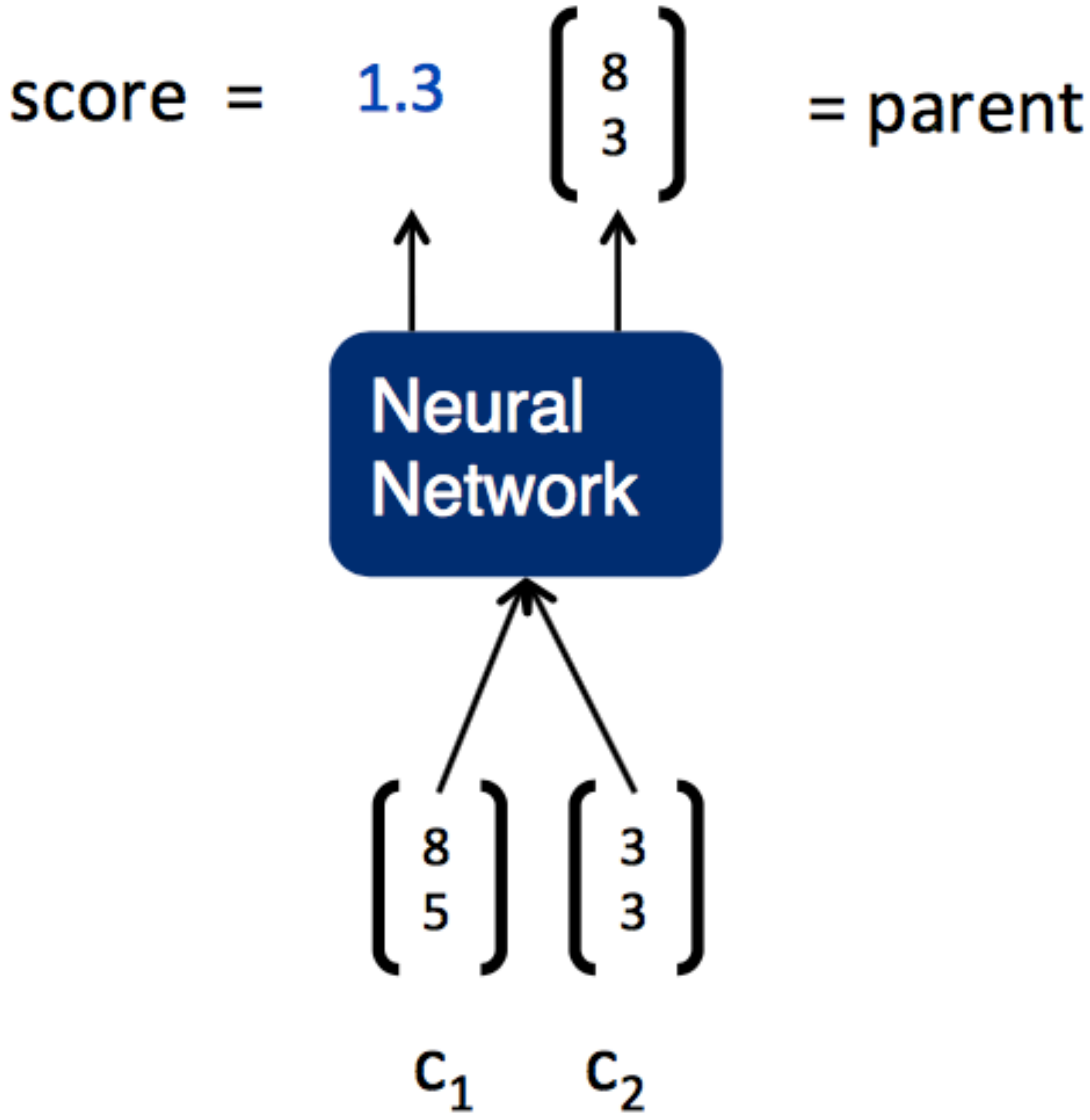




What we want?

Inputs: two candidate children's representations
Outputs:
1. The semantic representation if the two nodes are merged.
2. Score of how plausible the new node would be.

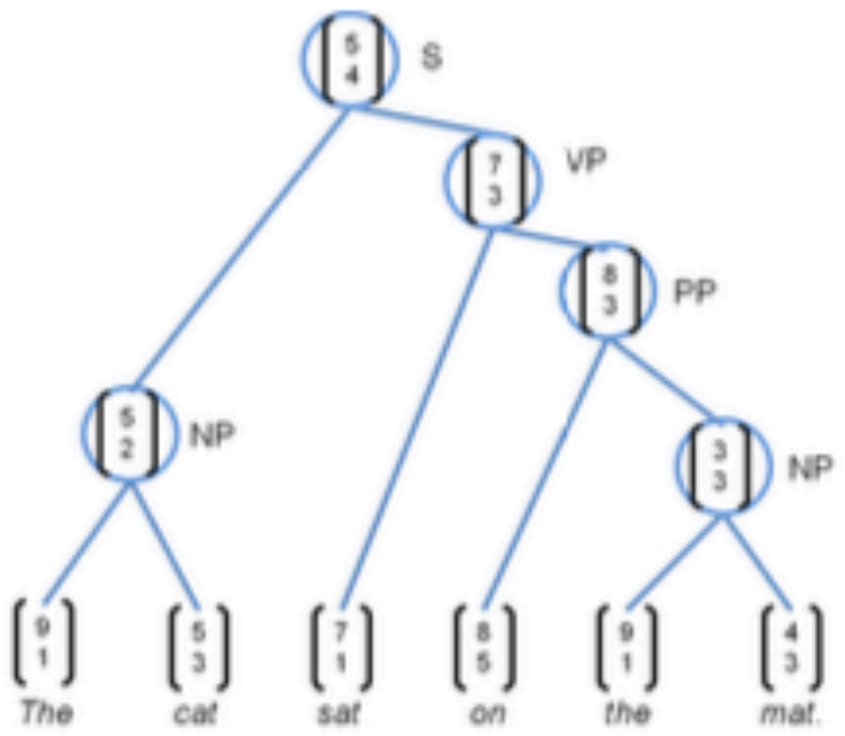




$$\text{score} = U^{\mathsf{T}} p$$

$$\boxed{p = \tanh\left(W \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right),}$$

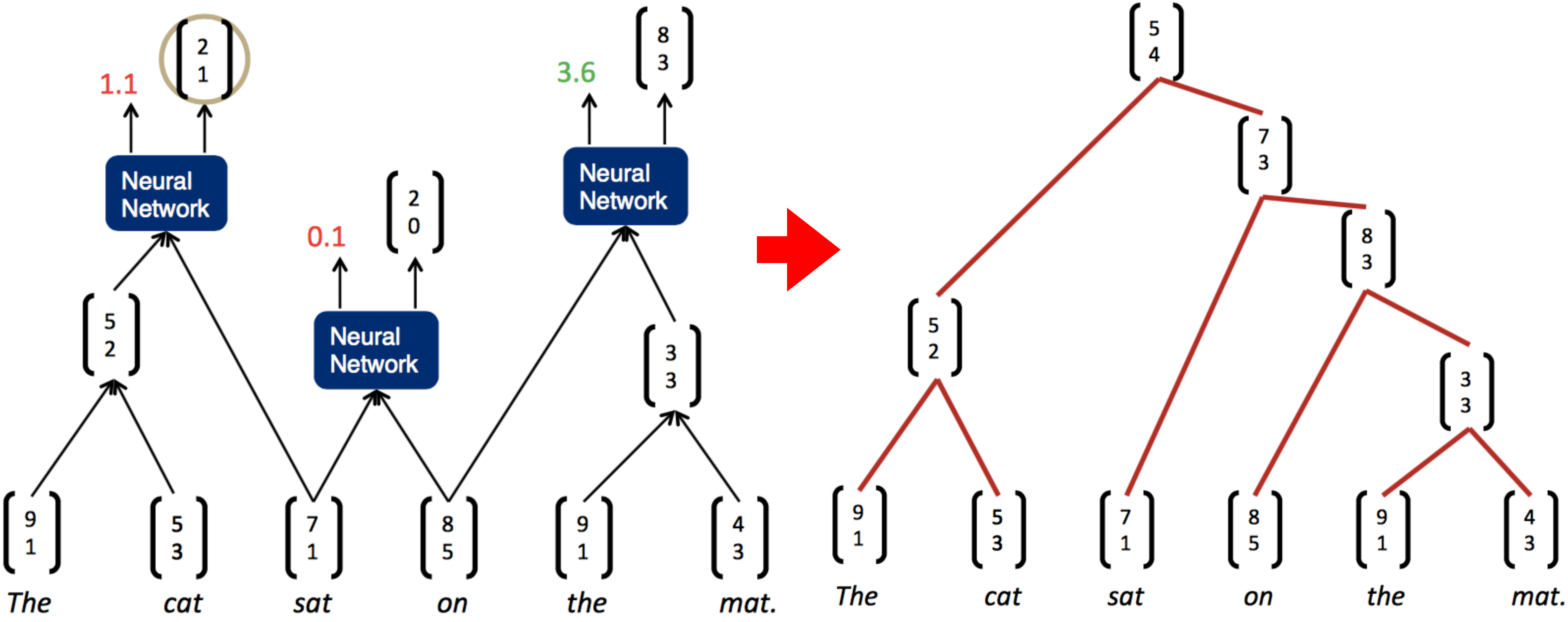
Same W parameters at all nodes of the tree



Example

3.1
5
2
0.3
0
1
0.1
2
0
0.4
1
0
2.3
3
3
Neural Network
Neural Network
Neural Network
Neural Network
Neural Network
9
1
5
3
7
1
8
5
9
1
4
3
The
cat
sat
on
the
mat.
1.1
2
1
Neural Network
5
2
0.1
2
0
0.4
1
0
2.3
3
3
Neural Network
Neural Network
Neural Network
9
1
5
3
7
1
8
5
9
1
4
3
The
cat
sat
on
the
mat.

Example



1.1
2
1
Neural Network
0.1
2
0
Neural Network
3.6
8
3
Neural Network
5
2
3
3
9
1
5
3
7
1
8
5
9
1
4
3
The
cat
sat
on
the
mat.
5
4
7
3
8
3
5
2
3
3
9
1
5
3
7
1
8
5
9
1
4
3
The
cat
sat
on
the
mat.

The score of a tree is computed by the sum of the parsing decision scores at each node:

$$s(x, y) = \sum_{n \in nodes(y)} s_n$$

x is sentence; *y* is parse tree

Backpropagaton Through Structure

Principally the same as general backpropagation

$$\boxed{\delta^{(l)} = \left((W^{(l)})^T \delta^{(l+1)}\right) \circ f'(z^{(l)}),}$$

$$\boxed{\frac{\partial}{\partial W^{(l)}} E_R = \delta^{(l+1)} (a^{(l)})^T + \lambda W^{(l)}}$$

Three differences resulting from the recursion and tree structure:

1. Sum derivatives of *W* from all nodes (like RNN)
2. Split derivatives at each node (for tree)
3. Add error messages from parent + node itself

Sum derivatives of all nodes

You can actually assume it's a different *W* at each node

Intuition via example:

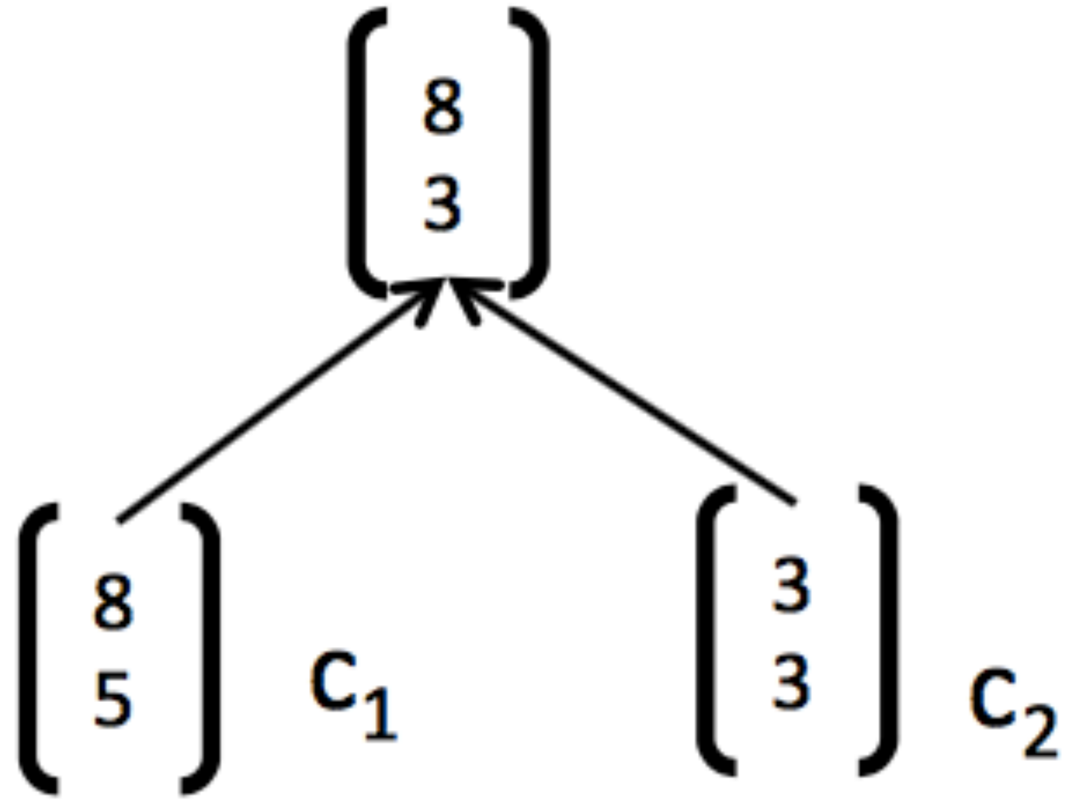
$$\begin{aligned}
&\frac{\partial}{\partial W} f(W(f(Wx)) \\
= \quad & f'(W(f(Wx))\left(\left(\frac{\partial}{\partial W} W\right) f(Wx) + W \frac{\partial}{\partial W} f(Wx)\right) \\
= \quad & f'(W(f(Wx))\left(f(Wx) + W f'(Wx)x\right)
\end{aligned}$$

If we take separate derivatives of each occurrence, we get same:

$$\begin{aligned}
&\frac{\partial}{\partial W_2} f(W_2(f(W_1x)) + \frac{\partial}{\partial W_1} f(W_2(f(W_1x)) \\
= \quad & f'(W_2(f(W_1x))\left(f(W_1x)\right) + f'(W_2(f(W_1x))\left(W_2 f'(W_1x)x\right) \\
= \quad & f'(W_2(f(W_1x))\left(f(W_1x) + W_2 f'(W_1x)x\right) \\
= \quad & f'(W(f(Wx))\left(f(Wx) + W f'(Wx)x\right)
\end{aligned}$$

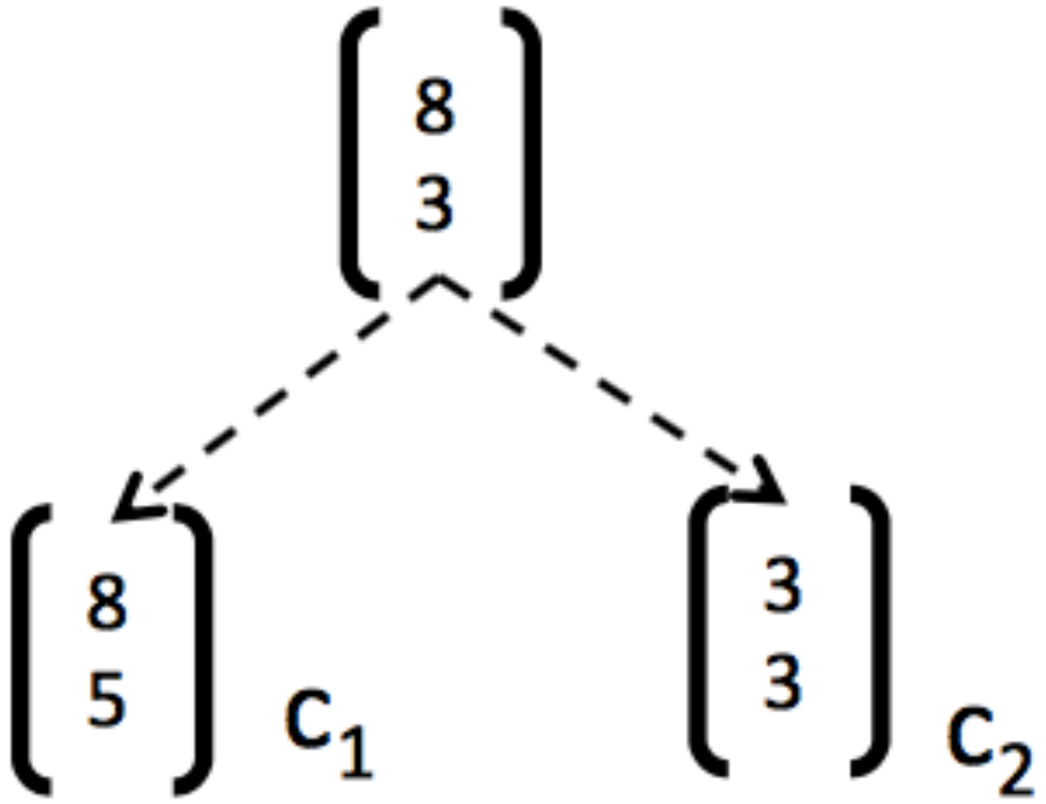
Split derivatives at each node

During forward prop, the parent is computed using 2 children



$$p = \tanh\left(W\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right)$$

Hence, the errors need to be computed wrt each of them:

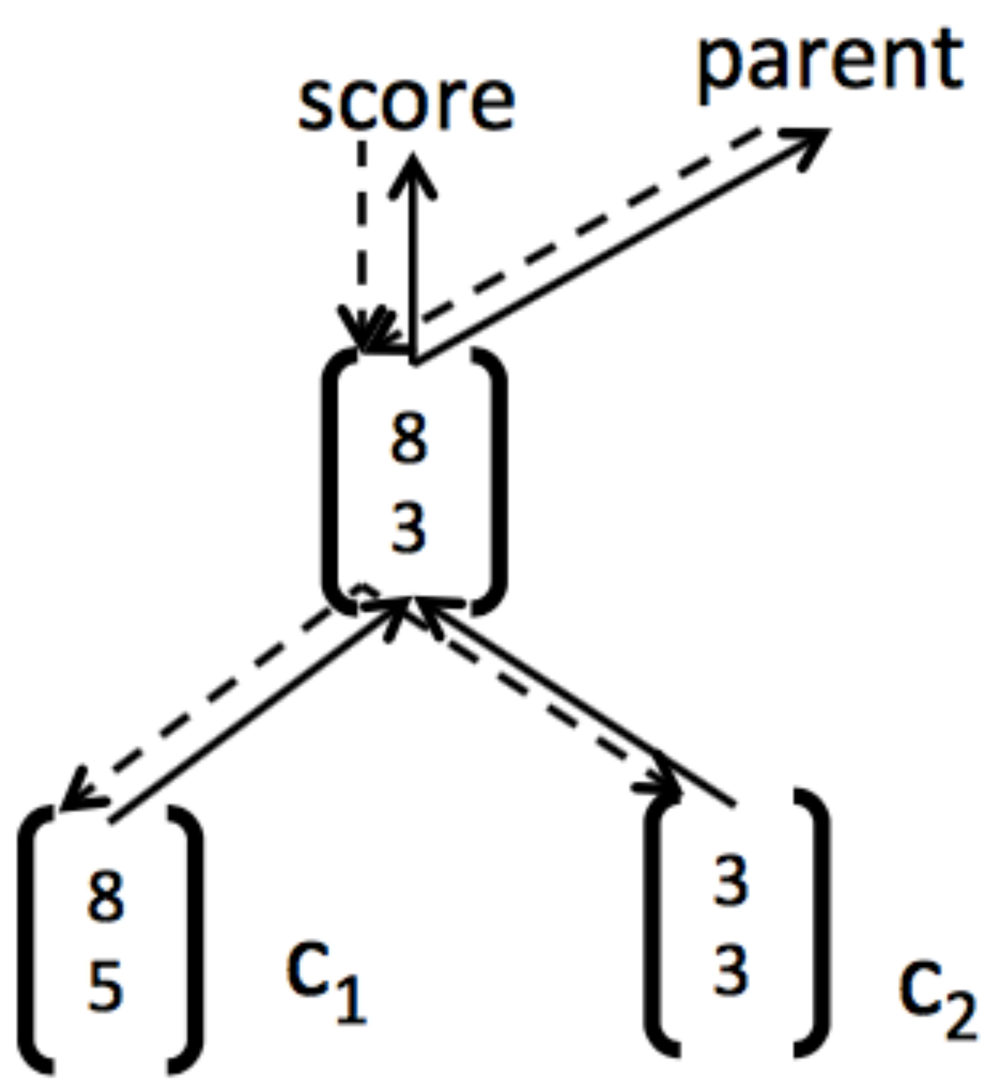


where each child's error is *n*-dimensional

$$\delta_{p\to c_1 c_2} = [\delta_{p\to c_1} \delta_{p\to c_2}]$$

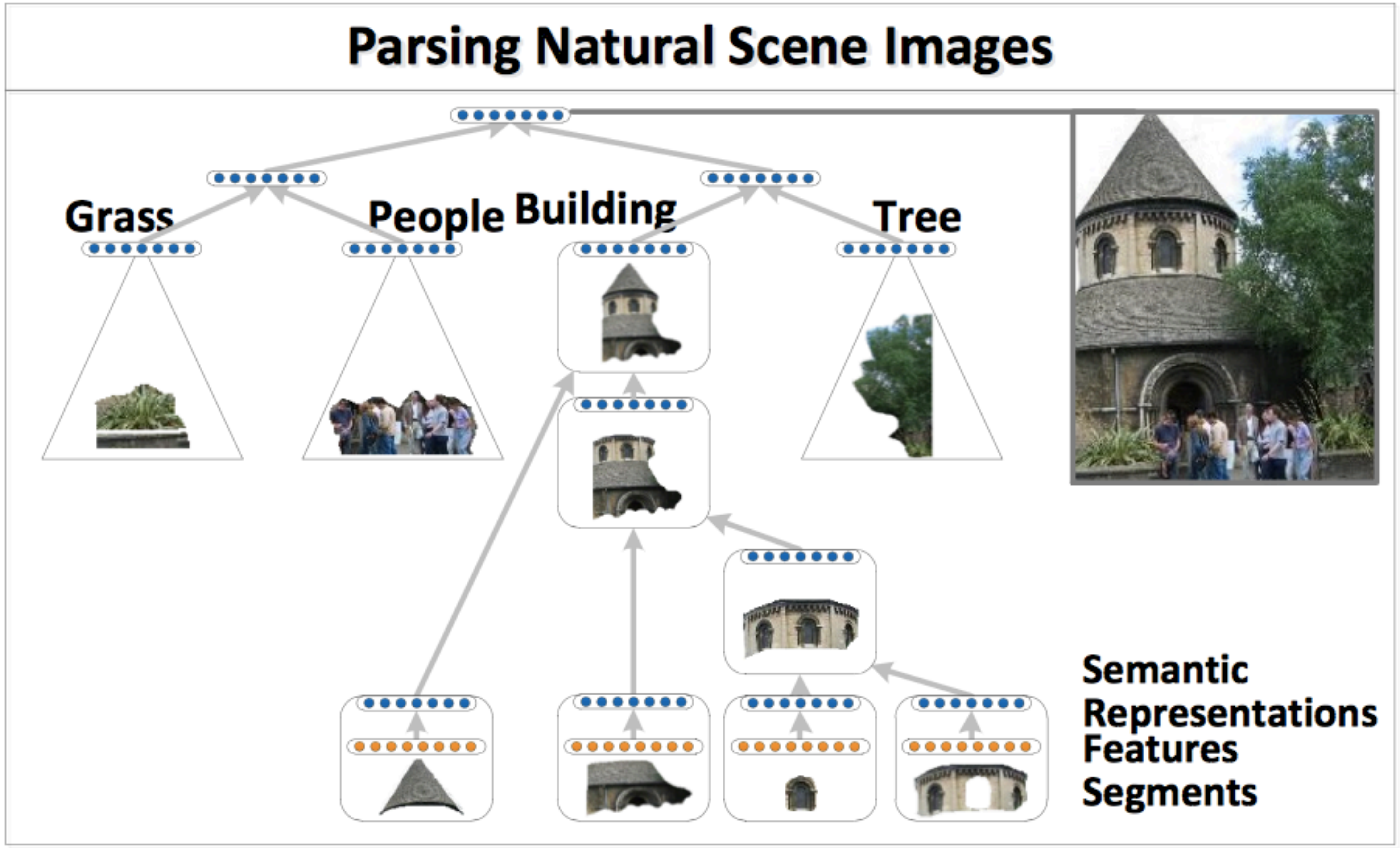
Add error messages

- At each node:
  - What came up (fprop) must come down (bprop)
  - Total error messages = error messages from parent + error message from own score

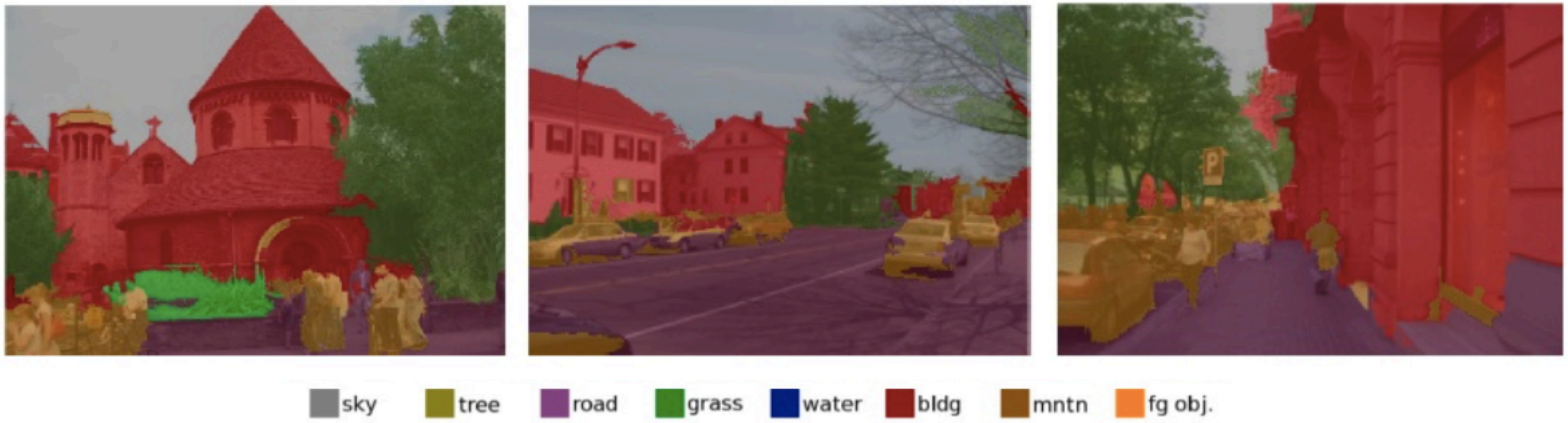


Addition

Same Recursive Neural Network as for natural language parsing!
(Socher et al. ICML 2011)



Addition: Multi-class segmentation



Method	Accuracy
Pixel CRF (Gould et al., ICCV 2009)	74.3
Classifier on superpixel features	75.9
Region-based energy (Gould et al., ICCV 2009)	76.4
Local labelling (Tighe & Lazebnik, ECCV 2010)	76.9
Superpixel MRF (Tighe & Lazebnik, ECCV 2010)	77.5
Simultaneous MRF (Tighe & Lazebnik, ECCV 2010)	77.5
Recursive Neural Network	**78.1**

Stanford Background Dataset (Gould et al. 2009)

Thank you for