STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

CS584 Natural Language Processing

Tree Recursive Neural Networks, Constituency Parsing

Department of Computer Science
Yue Ning
yue.ning@stevens.edu

Late Submission Policy

- **10%** penalty for late submission within **24 hours**.
- **40%** penalty for late submissions within **24-48 hours**.
- After 48 hours, you get **NO** points on the assignment.

Outline

- Recursive Neural Networks
- Constituency Parsing
- Structure prediction with simple Tree RNN
- Backpropagation through Structure
- More complex TreeRNN units

Recursive Neural Networks

- A superset of the previously discussed Recurrent Neural Network
- Recursive Neural Networks (RNNs) are perfect for settings that have nested hierarchy and an intrinsic recursive structure.

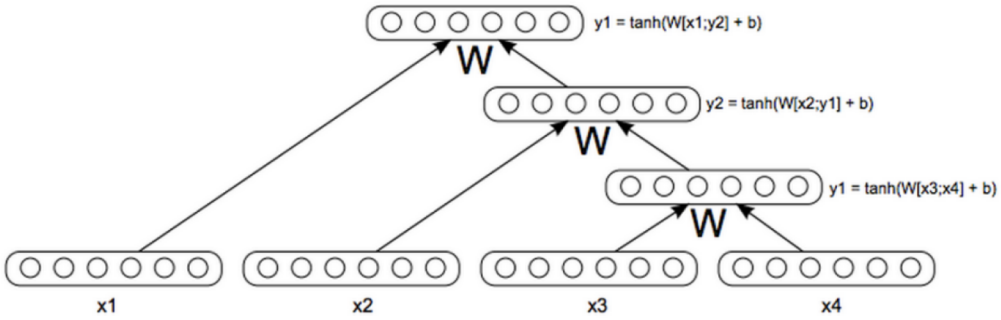
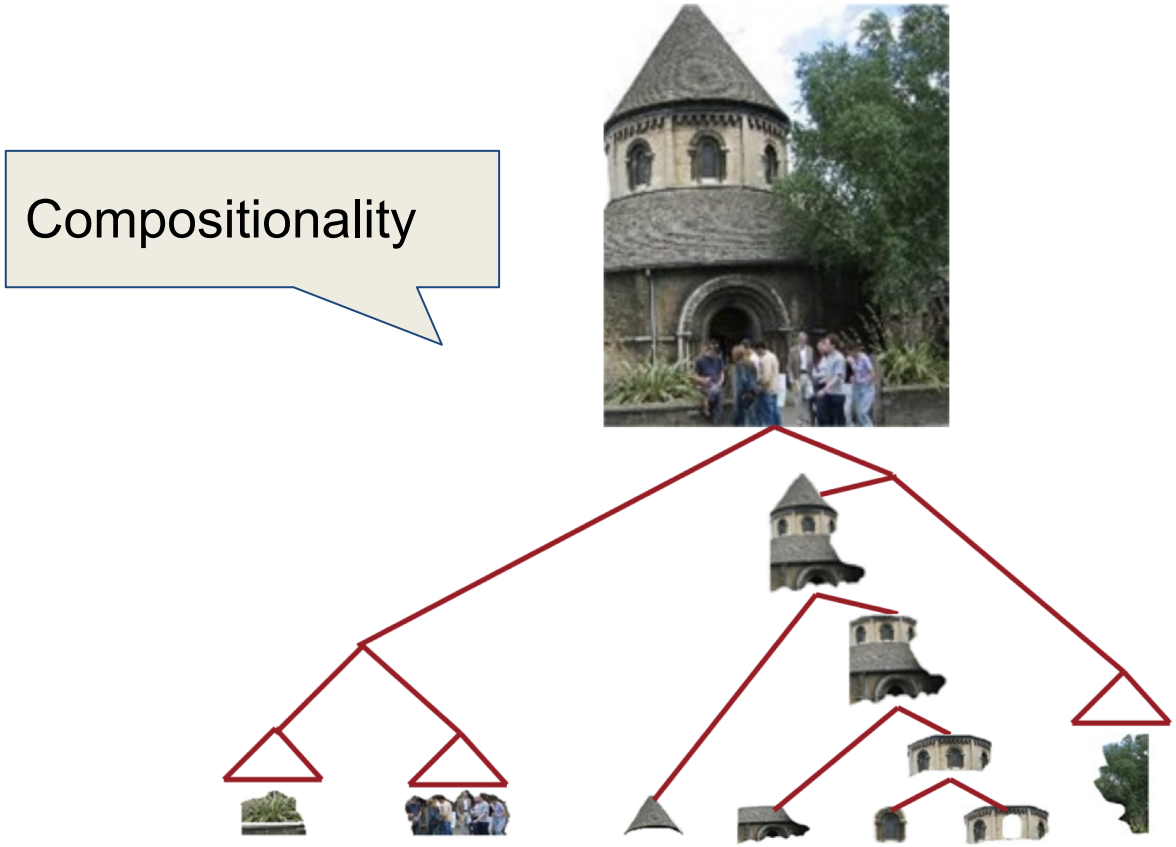


Figure 1: A standard Recursive Neural Network

Recursive Structure

Recursive structures is commonly found in natural scene images



Does a sentence have such a structure?

- Constituency Parse Tree

Constituency Parsing

- Natural Language Understanding requires being able to **extract** meaning from large text units from the understanding of smaller parts.
  - This extraction requires being able to understand how smaller parts are put together

- Two main techniques to analyze the syntactic structure of sentences
  - Dependency parsing and **constituency parsing**

Constituent

- In syntactic analysis, a constituent can be **a single word** or **a phrase** as a single unit within a hierarchical structure.
  - A phrase is a sequence of two or more words working as a unit within a sentence.
  - The phrase can be moved together or replaced

- examples
  - I want to participate in the great camping!
  - The great camping I want to participate in!
  - The great hiking I want to participate in!

Constituency Parsing

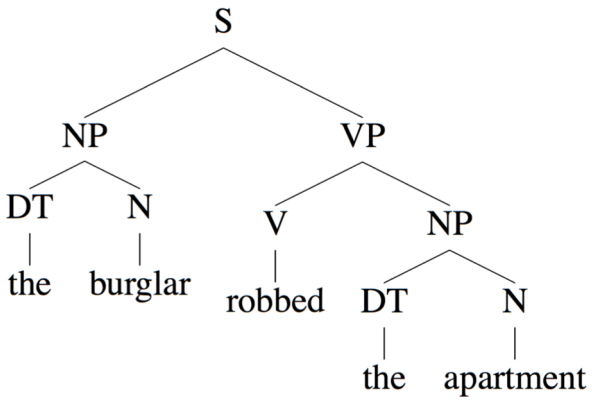
- Constituency Parsing is a way to break a piece of text (e.g. one sentence) into sub-phrases.

- One of its goals:To identify the constituents in the text which would be useful when extracting information from text.

S -> NP VP

- The basic clause structure is understood as a binary division of the clause into
  - subject (noun phrase NP) and
  - predicate (verb phrase VP)

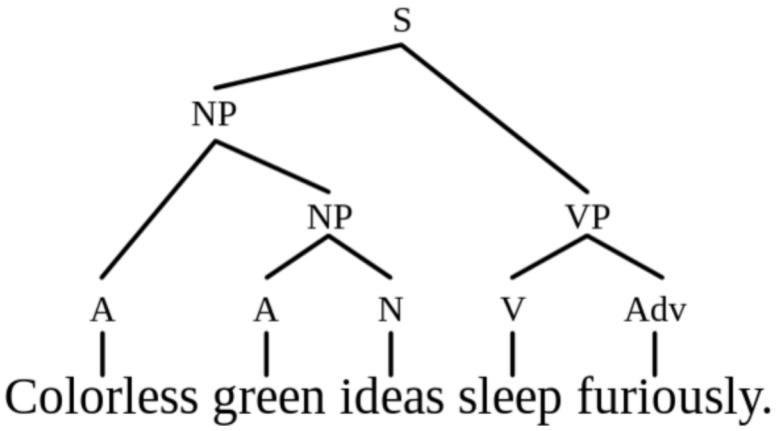
- Process of parsing: Deduce the rules by beginning with the sentence symbol S, and applying the phrase structure rules successively, finally substituting actual words for the abstract symbols.



S -> NP VP

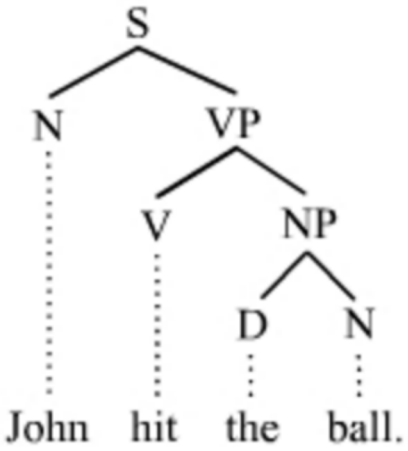
- However, the generated sentences might be syntactically correct but semantically nonsensical

- A well-known example:
  - Colorless green ideas sleep furiously



Constituency Parse Tree

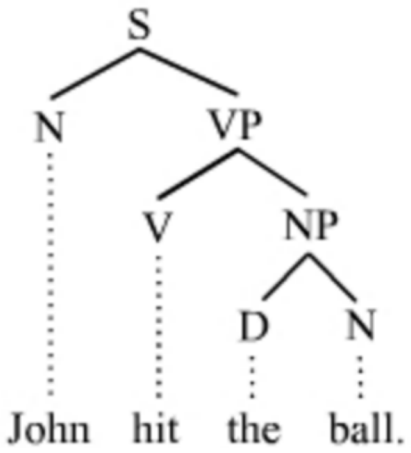
- In natural language, the constituents are likely to be nested inside one another.
- Usually we use a consistency parse tree to display the parsing process.
  - terminal nodes are the exact words
  - non-terminal nodes are phrases(e.g. Noun Phrase)
- “John hits the ball”



Constituency-based parse tree

Constituency Parse Tree

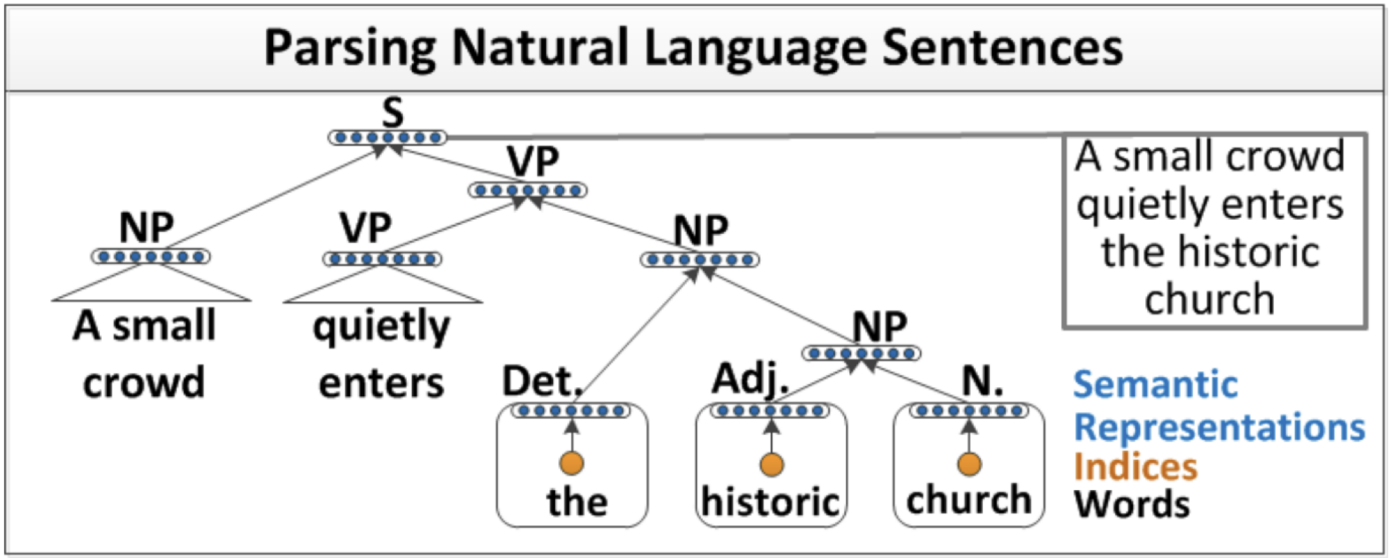
- Some abbreviations
  - S stands for sentence, the top-level structure.
  - NP stands for noun phrase including the subject of the sentence and the object of the sentence.
  - VP stands for verb phrase, which serves as the predicate.
  - V stands for verb.
  - D stands for determiner, such as "the"
  - N stands for noun



Constituency-based parse tree

Does a sentence have such a structure?

- Yes.
- “A small crowd quietly enters the historical church”



- This does seem pretty recursive.

Recursive Neural Networks

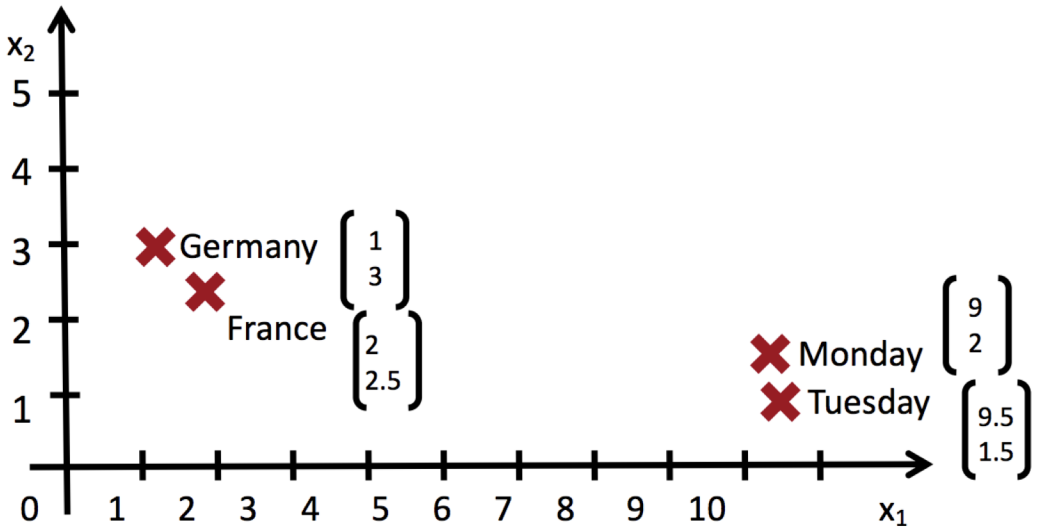
- The syntactic rules of language are highly recursive

- A benefit of modeling sentences with RNN's is that we can now input sentences of arbitrary length
  - which was a huge head scratcher for using Neural Nets in NLP.

- → **Structure prediction with simple Tree Recursive Neural Networks**

Building on Word Vector Space Models

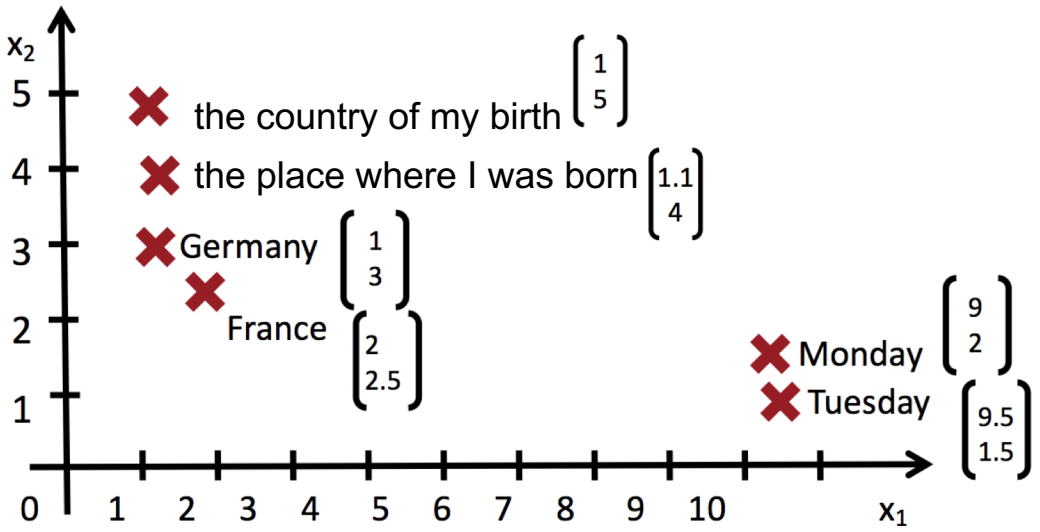
- We have seen ways to train unigram word vector
  - Similar words are close to each other



- How can we represent the meaning of longer phrases?
  - the country of my birth
  - the place where I was born

Building on Word Vector Space Models

- We have seen ways to train unigram word vector
  - Similar words are close to each other



- How can we represent the meaning of longer phrases?
  - the country of my birth
  - the place where I was born
- By mapping them into the same vector space!

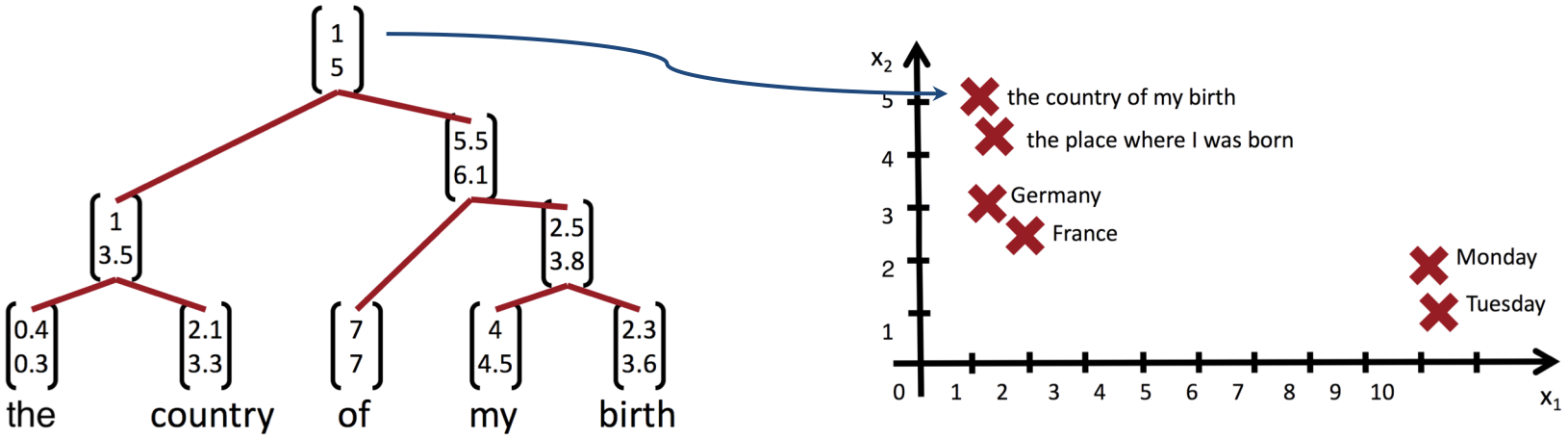
Map Phrases into a Vector Space

- Should we do the same for bigrams, trigrams, etc?

- This may work but there are two major issues
  - There are literally an infinite amount of possible combinations of words. Storing and training an infinite amount of vectors would just be absurd.
  - Some combinations of words while they might be completely reasonable to hear in language, may never be represented in our training corpus. So we would never learn them.

Map Phrases into a Vector Space

- Socher, Manning, and Ng. ICML,2011
- Use principle of compositionality
  - The meaning (vector) of a sentence is determined by
    - the meanings of its words and
    - the rules that combine them.

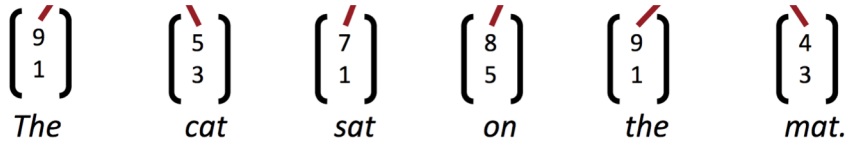


- Models in this section can jointly learn parse trees and compositional vector representations

Constituency Sentence Parsing: What we want

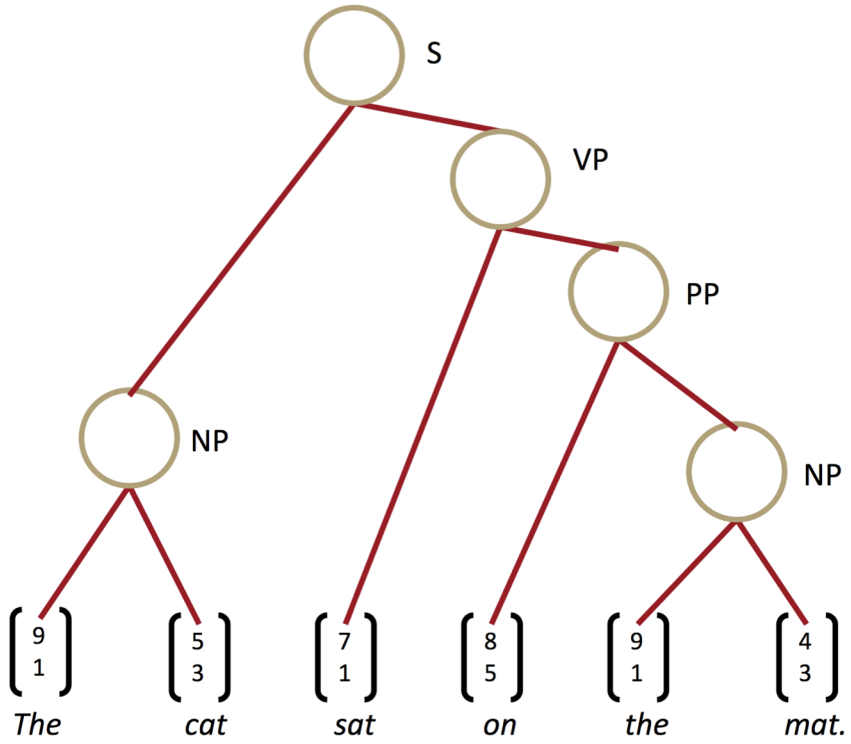
- Parsing:
  - find the right structure of the sentence

- Meaning computation:
  - find the meaning representation of the sentence



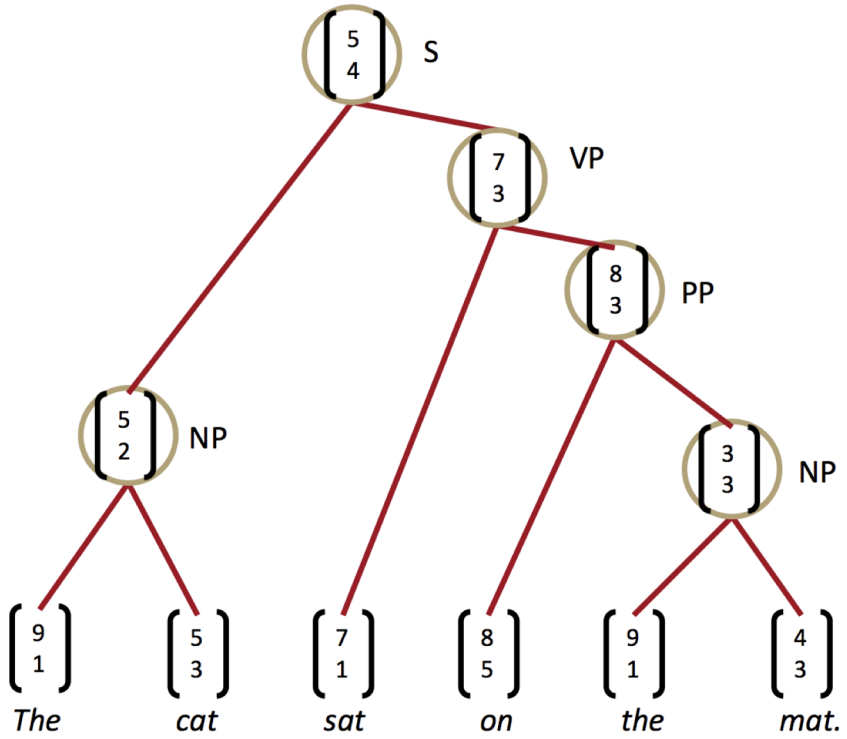
Constituency Sentence Parsing: What we want

- Parsing:
  - find the right structure of the sentence



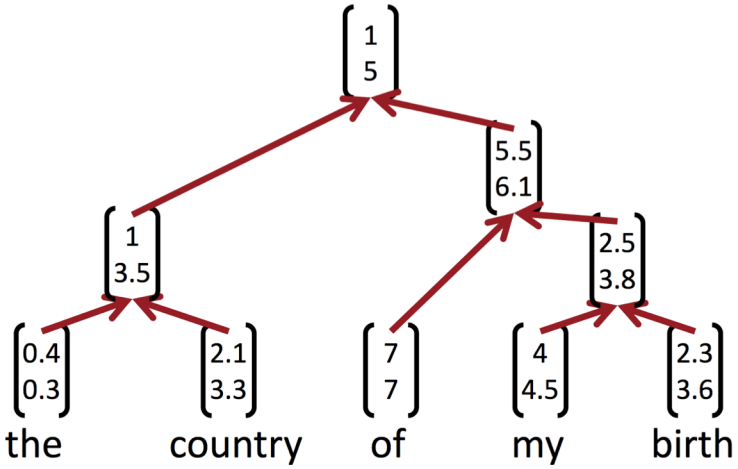
Learn Structure and Representation

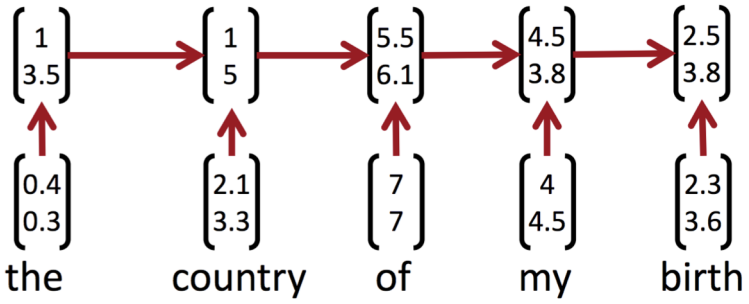
- Meaning computation:
  - find the meaning representation of the sentence



Recursive vs. Recurrent Neural Networks

- Recursive neural nets require a tree structure
- Recurrent neural nets cannot capture phrases without prefix context and often capture too much of last words in final vector





Recursive Neural Networks for Structure Prediction

- Goal: to learn a function f : $\mathcal{X} \rightarrow \mathcal{Y}$

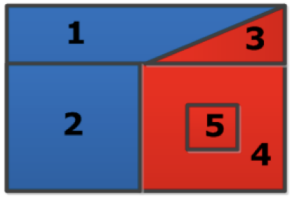
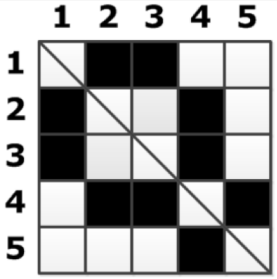
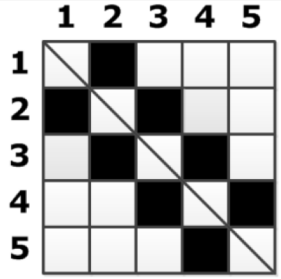
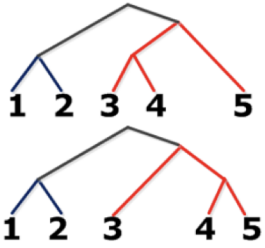
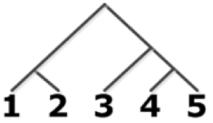
- An input x consists of two parts:
  - A set of activation vectors $\{a_1, \ldots, a_{N_{segs}}\}$ which represent input elements such as image segments or words of a sentence.
  - A symmetric adjacency matrix A, where A(i, j) = 1, if segment i neighbors j.
    - This matrix defines which elements can be merged

- Output Y is the set of all possible binary parse trees.

paper: https://ai.stanford.edu/~ang/papers/icml11-ParsingWithRecursiveNeuralNetworks.pdf

Recursive Neural Networks for Structure Prediction

- Illustration of the RNN training inputs

	Image	Text
Input Instance		The house has 1 2 3 a window 4 5
Adjacency Matrix		
Set of Correct Tree Structures		

Recursive Neural Networks for Structure Prediction

- We denote the set of all possible trees that can be constructed from an input x as $\mathcal{T}(x)$

- When training the visual parser, we have labels l for all segments. We define an equivalence set of correct trees $Y(x, l)$.

Max-Margin Estimation

- Similar to (Taskar et al., 2004), define a structured margin loss $\Delta(x, l, \hat{y})$ for proposing a parse $\hat{y}$ for input x with labels l .
- Check whether the subtree $subTree(d)$ underneath a nonterminal node d in $\hat{y}$ appears in any of the ground truth trees of $Y(x, l)$

$$\Delta(x, l, \hat{y}) = \kappa \sum_{d \in N(\hat{y})} \mathbf{1}\{subTree(d) \notin Y(x, l)\}$$

number of wrong subtrees

where $N(\hat{y})$ is the set of non-terminal nodes and κ is a parameter.

Max-Margin Estimation

- Ideal $\mathrm{s}(\mathrm{RNN}(\theta, x_i, y_i)) \geq \mathrm{s}(\mathrm{RNN}(\theta, x_i, \hat{y})) + \Delta(x_i, l_i, \hat{y})$
- The supervised max-margin objective

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} r_i(\theta) + \frac{\lambda}{2}||\theta||^2$$

number of wrong subtrees

where

$$r_i(\theta) = \max_{\hat{y}\in\mathcal{T}(x_i)} \left(s(\mathrm{RNN}(\theta, x_i, \hat{y})) + \Delta(x_i, l_i, \hat{y})\right) - \max_{y_i\in Y(x_i,l_i)} \left(s(\mathrm{RNN}(\theta, x_i, y_i))\right)$$

- Minimizing this objective maximizes the correct tree's score and minimizes (up to a margin) the score of the highest scoring but incorrect tree.

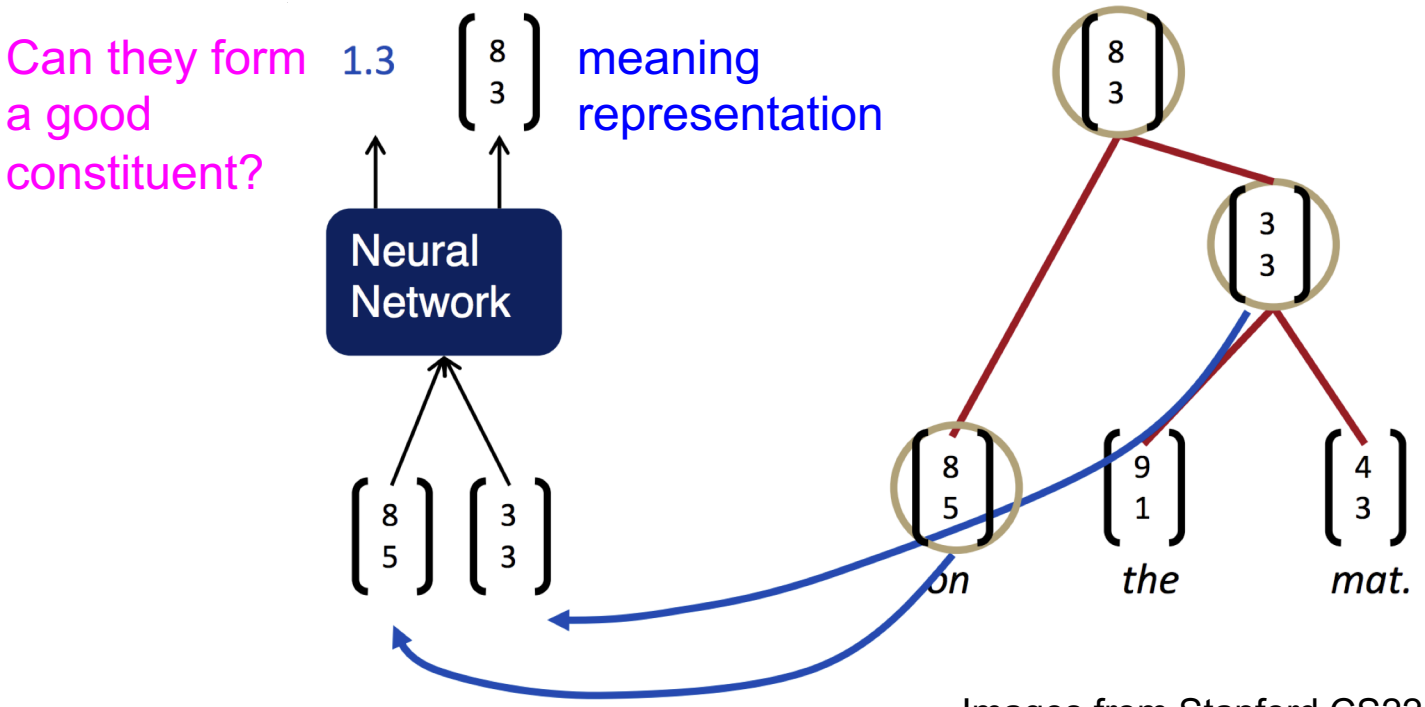
Recursive Neural Networks for Structure Prediction

- We defined the general learning framework
  - Max-Margin Estimation

- We then explain how we predict parse trees and compute their scores with RNNs

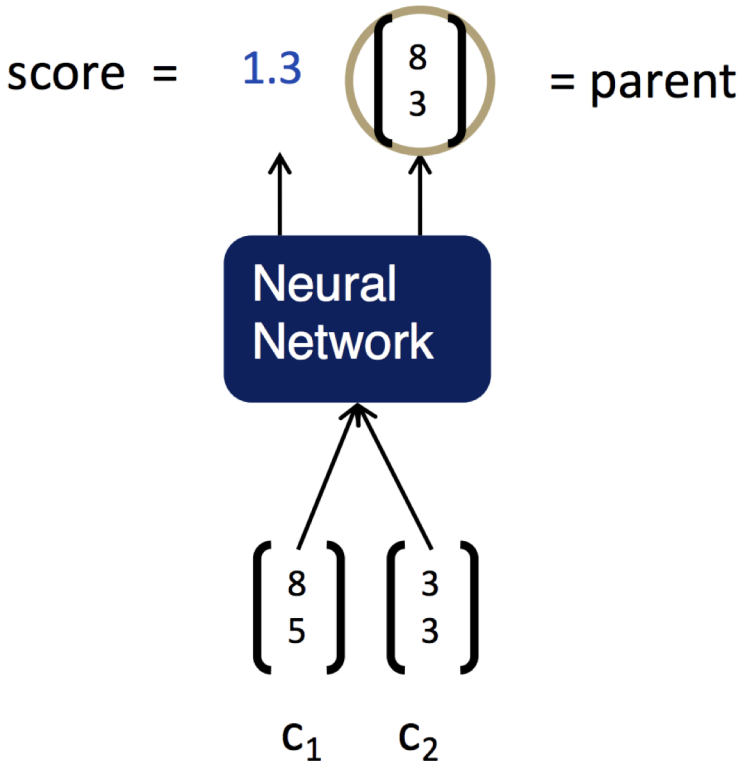
Recursive Neural Networks for Structure Prediction

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



Images from Stanford CS224N Lecture 18 Slides

Recursive Neural Network Definition

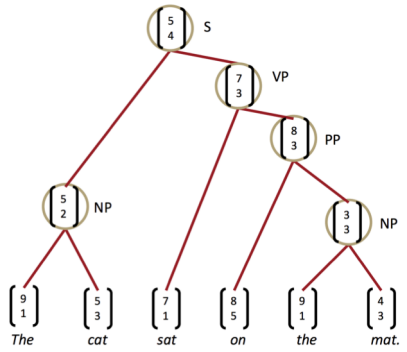


score = $U^T p$

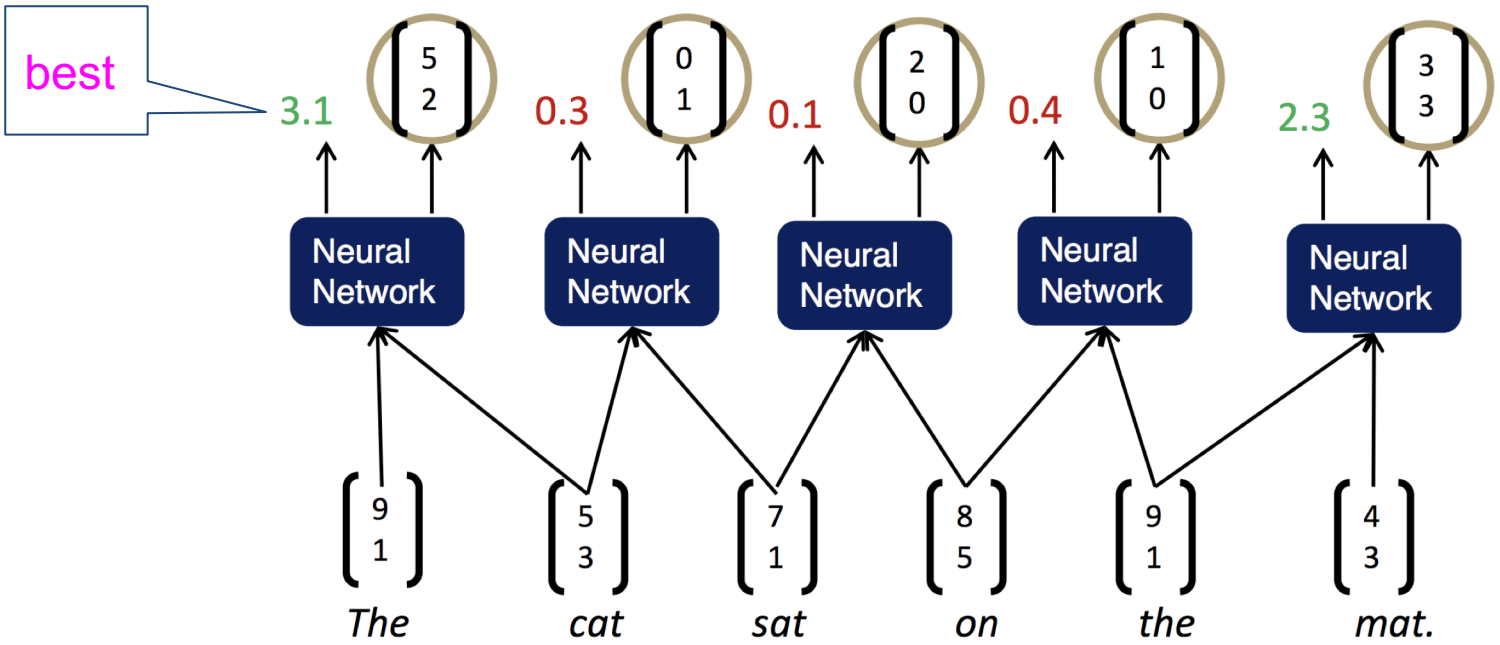
recurrent neural network

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

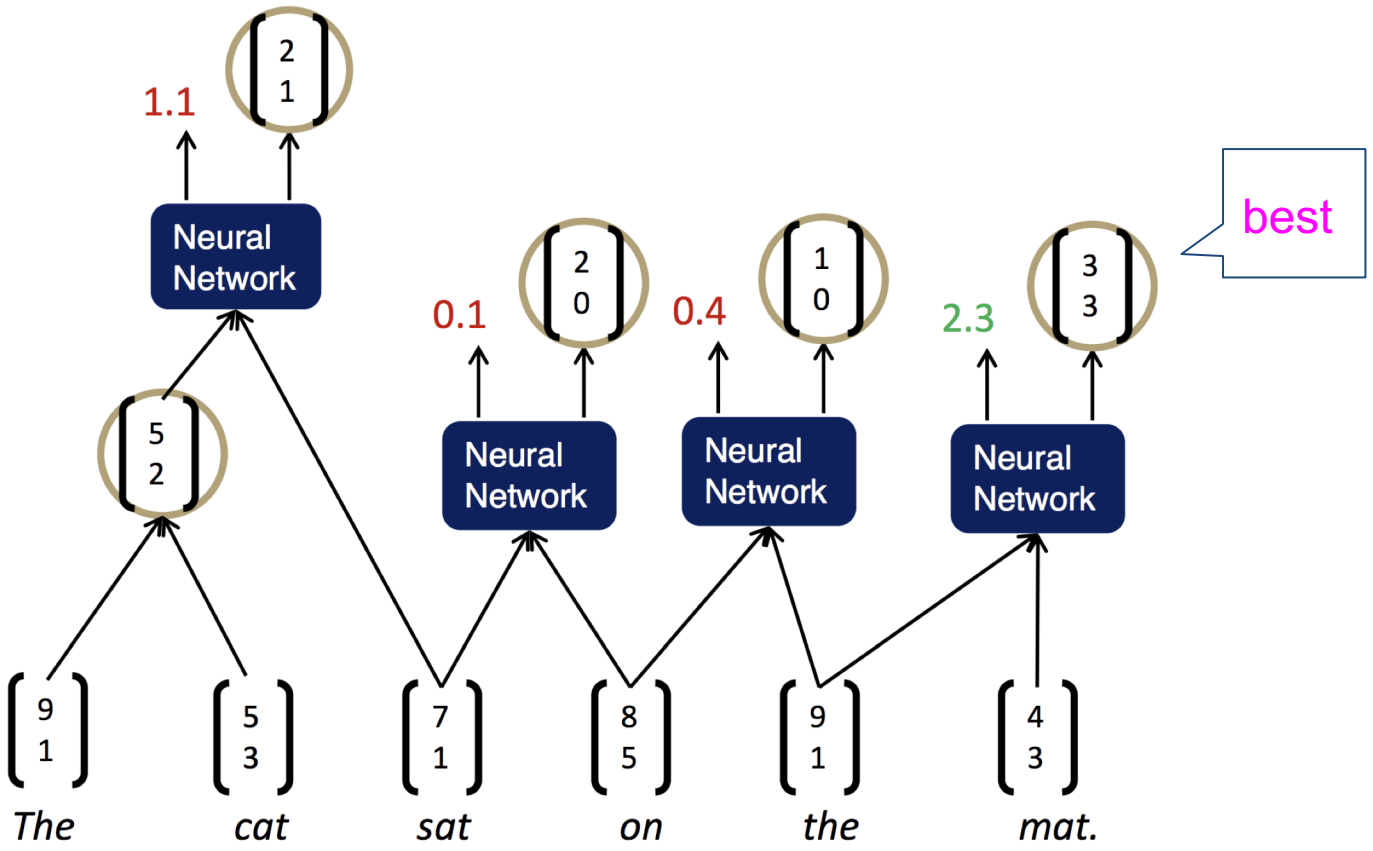
Same W parameters at all nodes of the tree



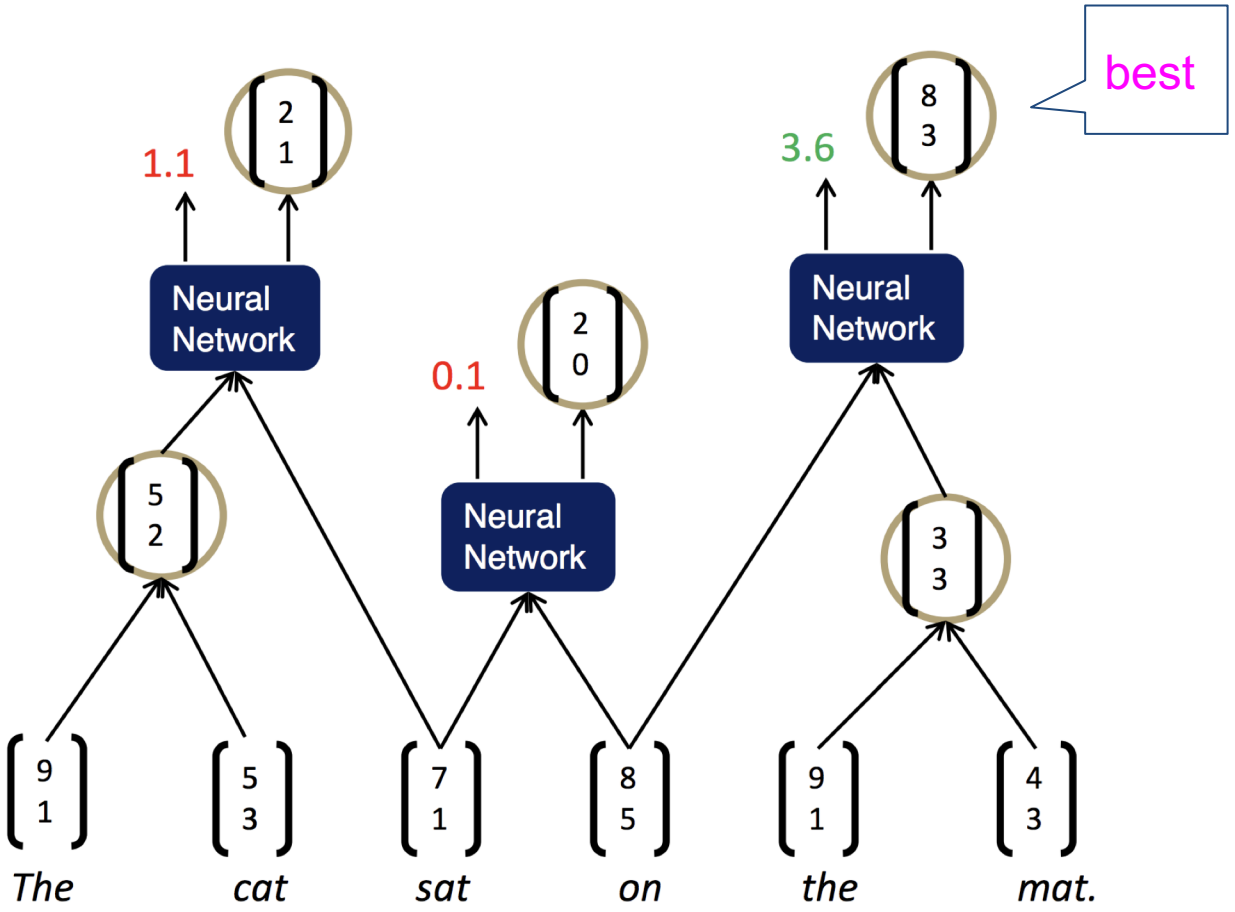
Parsing a sentence with an RNN (greedily)



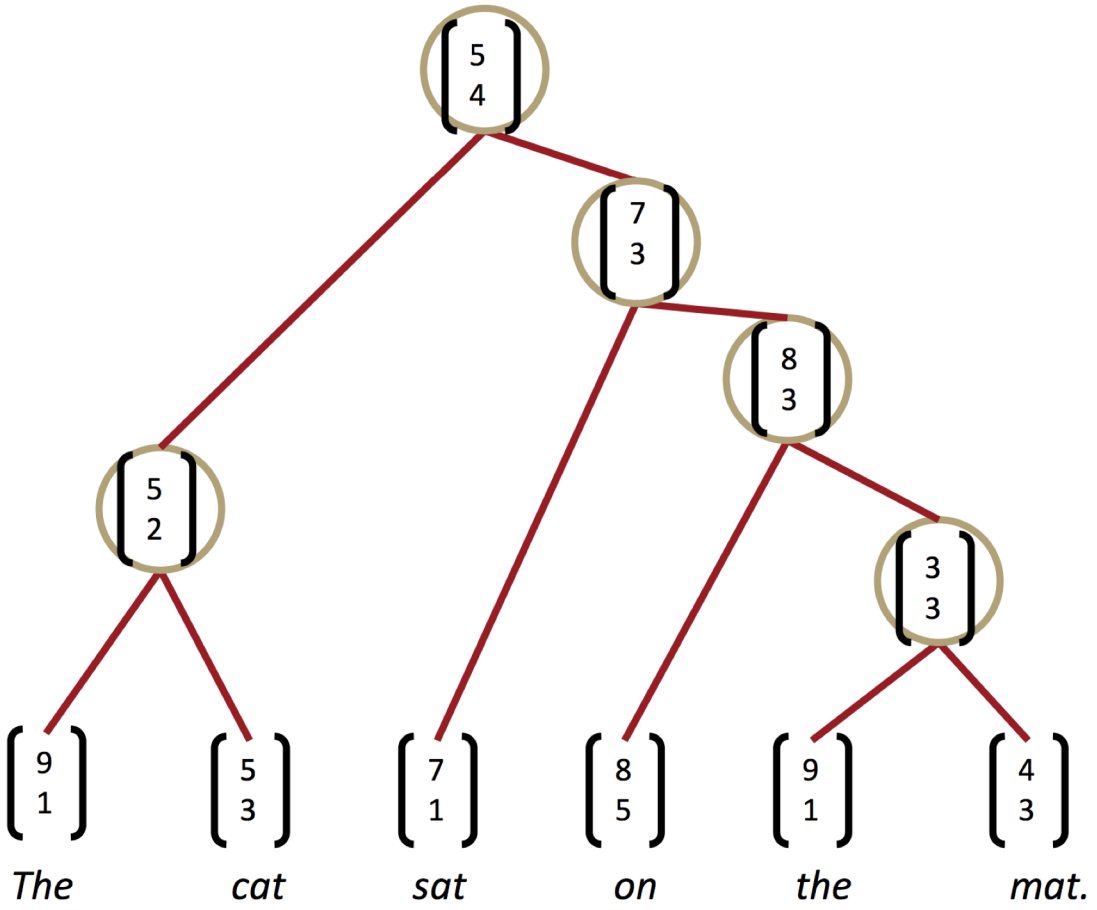
Parsing a sentence



Parsing a sentence



Parsing a sentence



Computing the Scores

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

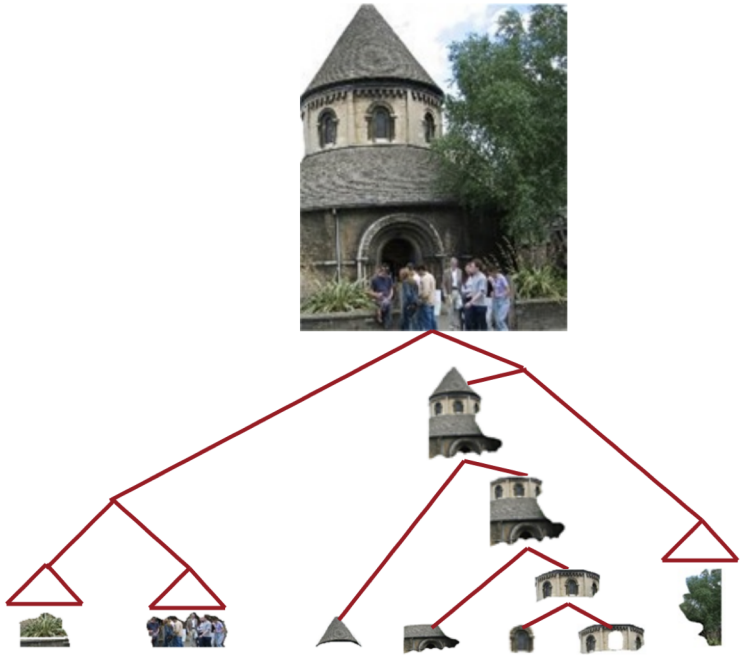
$$s(\mathrm{RNN}(\theta, x_i, \hat{y})) = \sum_{d \in N(\hat{y})} s_d$$

- where $N(\hat{y})$ is the set of non-terminal nodes in parse $\hat{y}$

- Then use Max-Margin Estimation

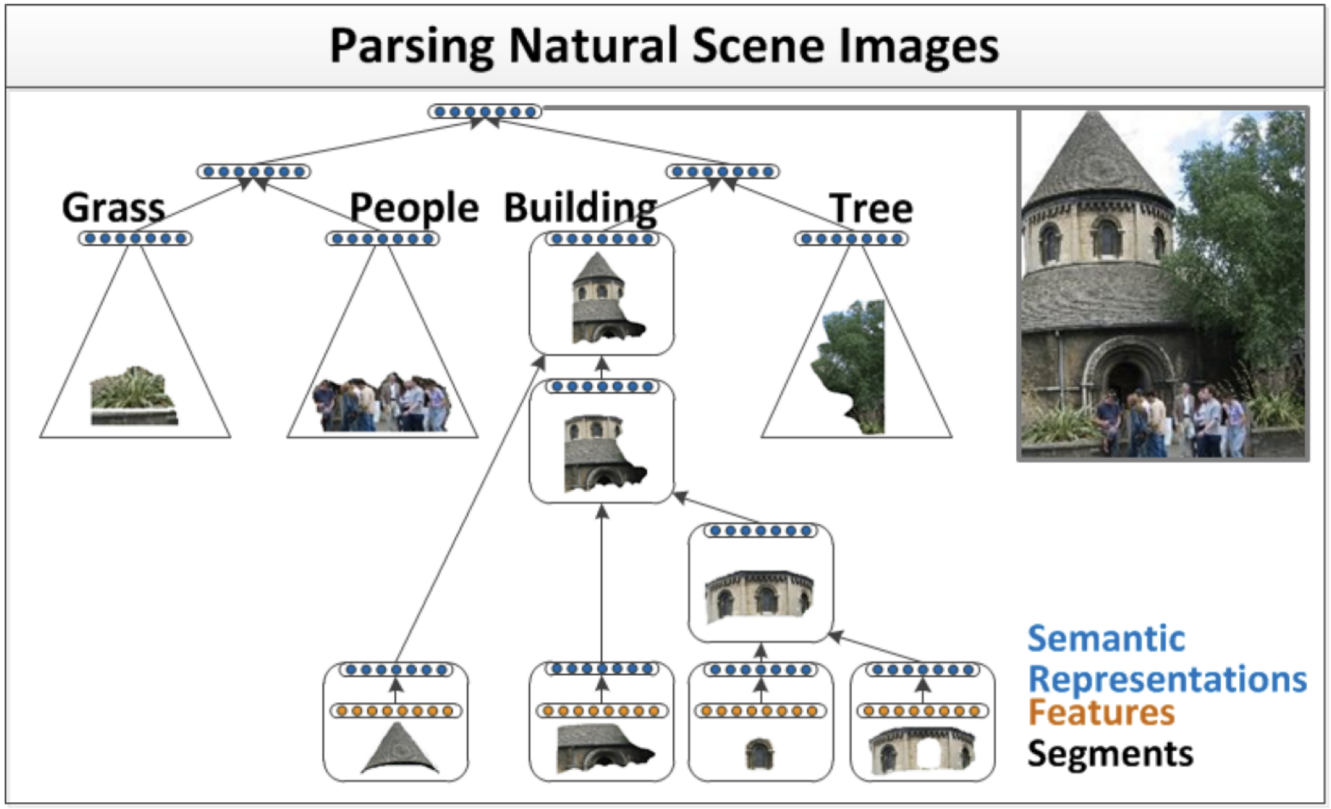
Scene Parsing

- Similar principle of compositionality.
  - The meaning of a scene image is also a function of smaller regions
  - how they combine as parts to form larger objects
  - and how the objects interact.



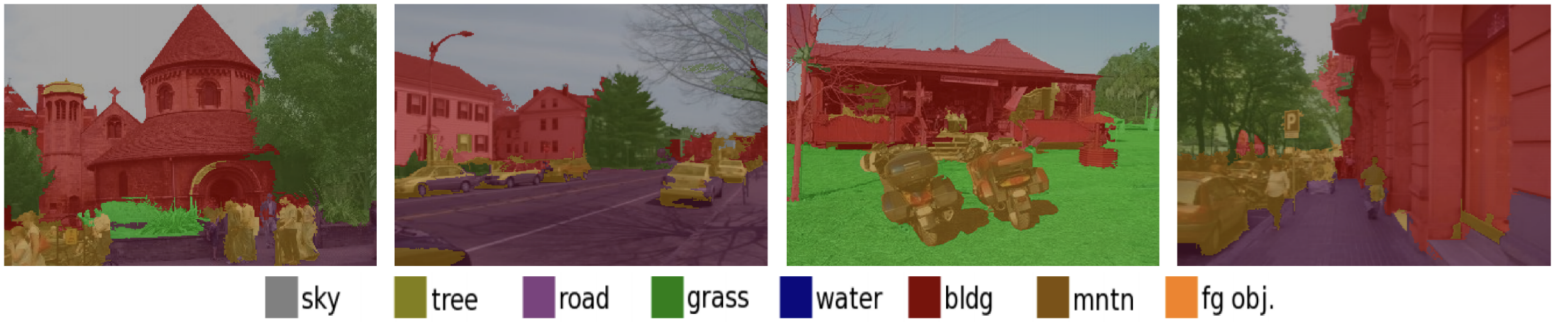
Algorithm for Parsing Images

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



Multi-class segmentation

Pixel level multi-class segmentation accuracy on the Stanford background dataset (Gould et al. 2009)



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**

paper: https://ai.stanford.edu/~ang/papers/icml11-ParsingWithRecursiveNeuralNetworks.pdf

Backpropagation Through Structure

- Introduced by Goller & Küchler (1996)
- Principally the same as general backpropagation

- Calculations 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

BTS: 1) Sum derivatives of all nodes

- You can actually assume it's a different W at each node
- Intuition via example:

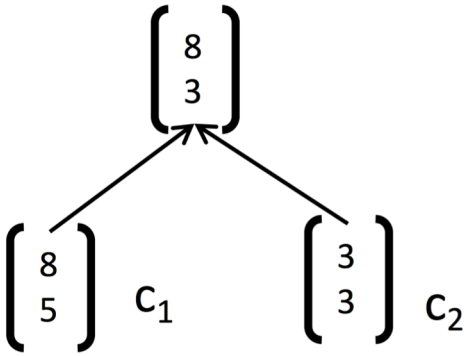
$$
\begin{aligned}
& \frac{\partial}{\partial W} f(W(f(Wx)) \\
= \; & f'(W(f(Wx)) \left(\left(\frac{\partial}{\partial W} W \right) f(Wx) + W \frac{\partial}{\partial W} f(Wx) \right) \\
= \; & f'(W(f(Wx)) \left(f(Wx) + W f'(Wx) x \right)
\end{aligned}
$$

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

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

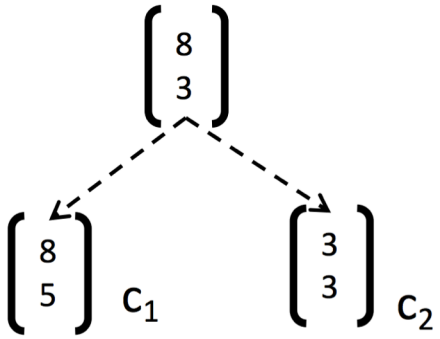
BTS: 2) Split derivatives at each node

- 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:

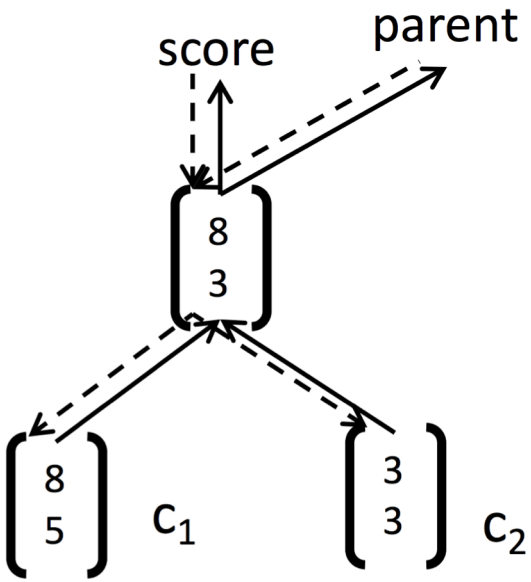


where each child's error is *n*-dimensional

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

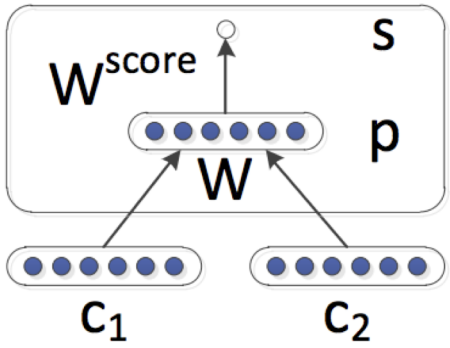
BTS: 3) Add error messages

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



Discussion: Simple TreeRNN

- Decent results with single matrix TreeRNN
- Single weight matrix TreeRNN could capture some phenomena but not adequate for more complex, higher order composition and parsing long sentences
- There is no real interaction between the input words
- The composition function is the same for all syntactic categories, punctuation, etc.



More complex TreeRNN units

- Syntactically-Untied RNN
  - Socher, Bauer, Manning, Ng 2013

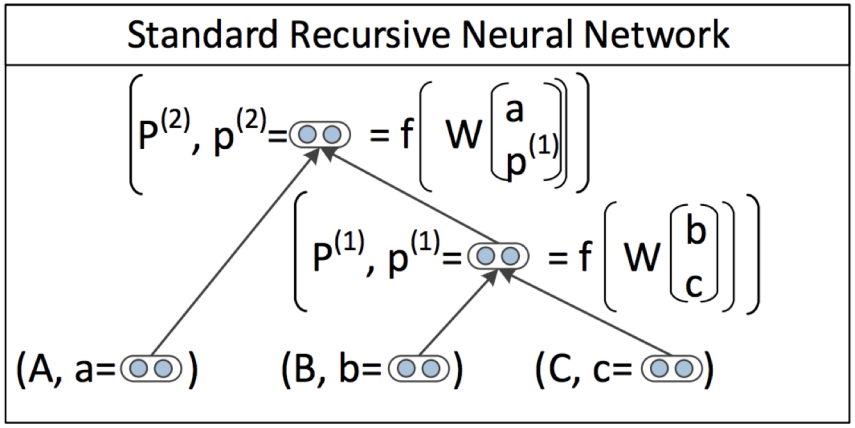
- Compositionality Through Recursive Matrix-Vector Spaces
  - Socher, Huval, Bhat, Manning, & Ng, 2012

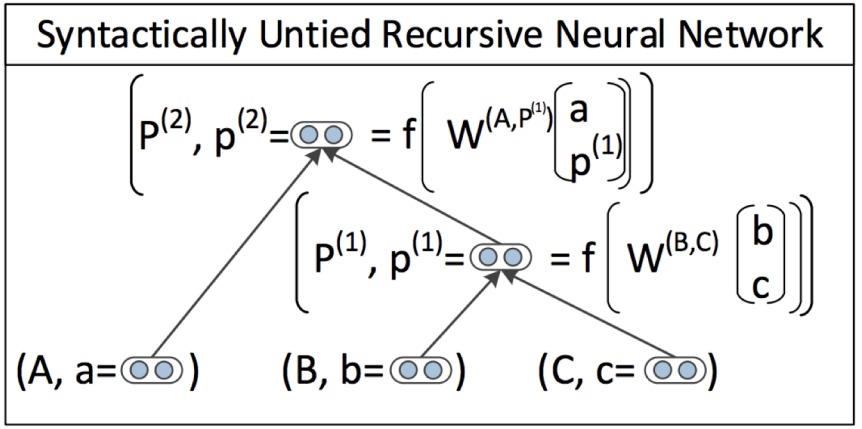
- Recursive Neural Tensor Network
  - Socher, Perelygin, Wu, Chuang, Manning, Ng, and Potts 2013

v2: Syntactically-Untied RNN

[Socher, Bauer, Manning, Ng 2013]

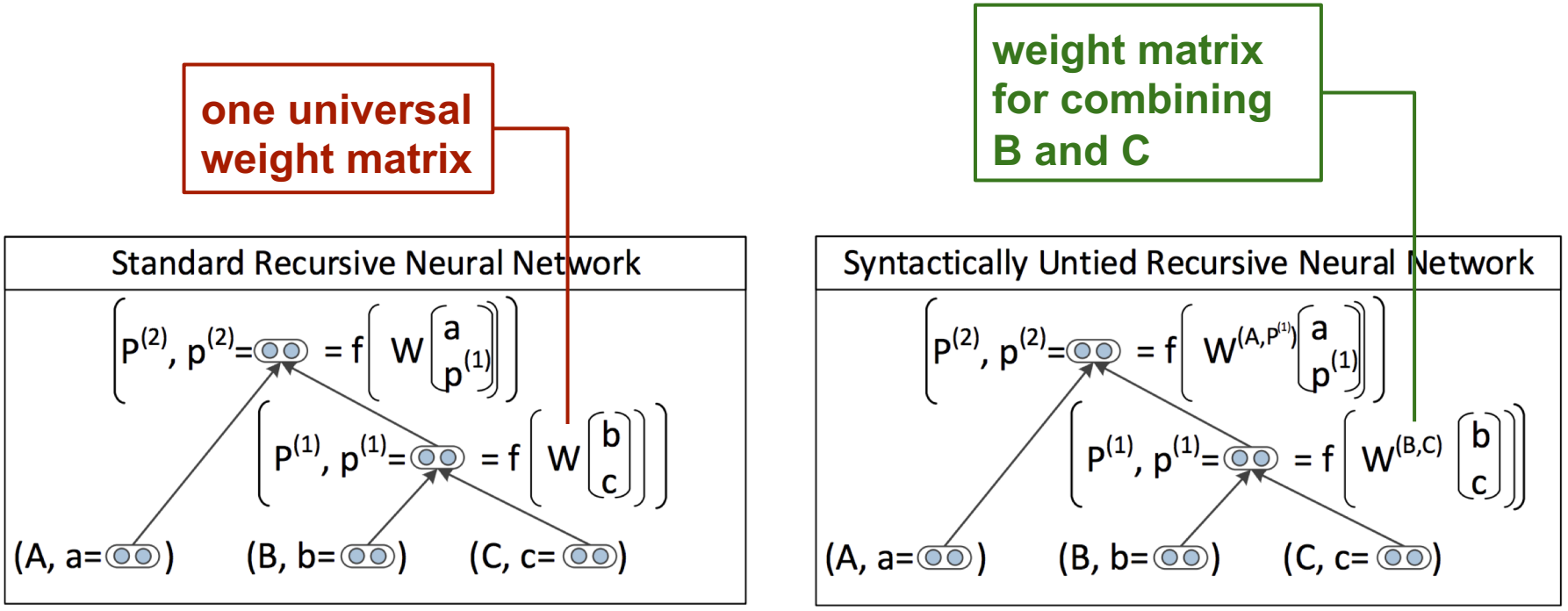
- A symbolic Context-Free Grammar (CFG) backbone is adequate for basic syntactic structure
- We use the discrete syntactic categories of the children to choose the composition matrix
- A TreeRNN can do better with different composition matrix for different syntactic environments
- The result gives us a better semantics





Syntactically-Untied RNN

- A symbolic Context-Free Grammar (CFG) backbone is adequate for basic syntactic structure
- We use the discrete syntactic categories of the children to choose the composition matrix



Compositional Vector Grammars

- Problem:
  - Speed. Every candidate score in beam search needs a matrix-vector product.
- Solution:
  - Compute score only for a subset of trees coming from a simpler, faster model (PCFG)
    - Prunes very unlikely candidates for speed
    - Provides coarse syntactic categories of the children for each beam candidate

- Compositional Vector Grammar = PCFG + TreeRNN

Experiments

- Standard WSJ split, labeled F1
- Based on simple PCFG with fewer states
- Fast pruning of search space, few matrix-vector products
- 3.8% higher F1

Parser	dev (all)	test$\leq$ 40	test (all)
Stanford PCFG	85.8	86.2	85.5
Stanford Factored	87.4	87.2	86.6
Factored PCFGs	89.7	90.1	89.4
Collins			87.7
SSN (Henderson)			89.4
Berkeley Parser			90.1
CVG (RNN)	85.7	85.1	85.0
CVG (SU-RNN)	91.2	91.1	90.4
Charniak-SelfTrain			91.0
Charniak-RS			92.1

SU-RNN / CVG

[Socher, Bauer, Manning, Ng 2013]

- Learns soft notion of head words
- Initialization:

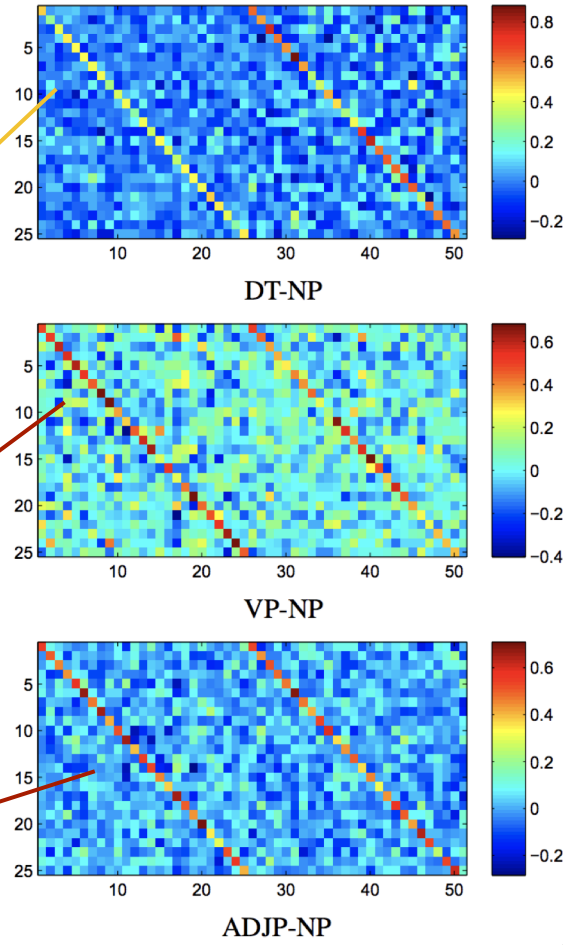
$$W^{(\cdot\cdot)} = 0.5[I_{n\times n} I_{n\times n} 0_{n\times 1}] + \epsilon$$

$$W^{(AB)} \begin{bmatrix} a \\ b \\ 1 \end{bmatrix} = \begin{bmatrix} W^{(A)} W^{(B)} bias \end{bmatrix} \begin{bmatrix} a \\ b \\ 1 \end{bmatrix}$$

$$= W^{(A)} a + W^{(B)} b + bias.$$

- Three binary composition matrices showing that head words dominate the composition.

category-specific W matrices



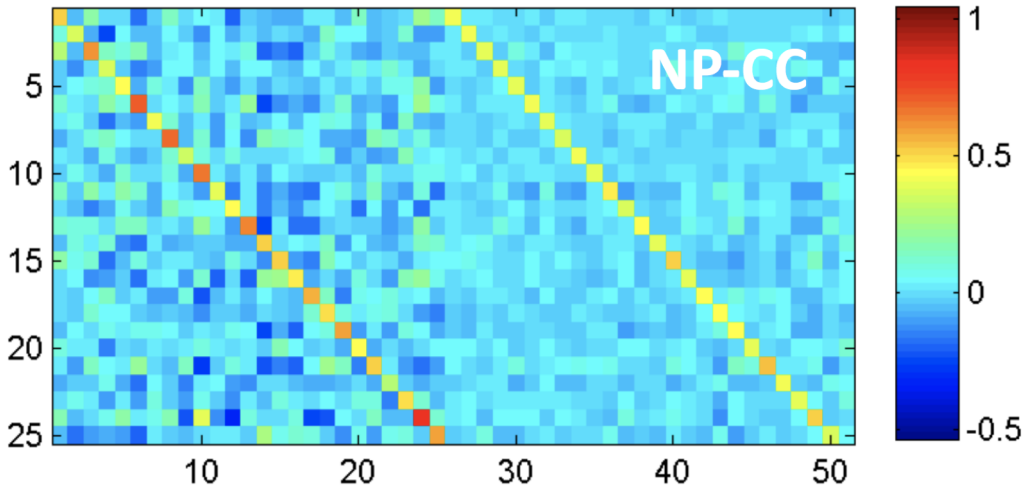
yellow: nothing interesting learned

something interesting about semantics learned

SU-RNN / CVG

- The models are learning which children of a phrase are actually the important one

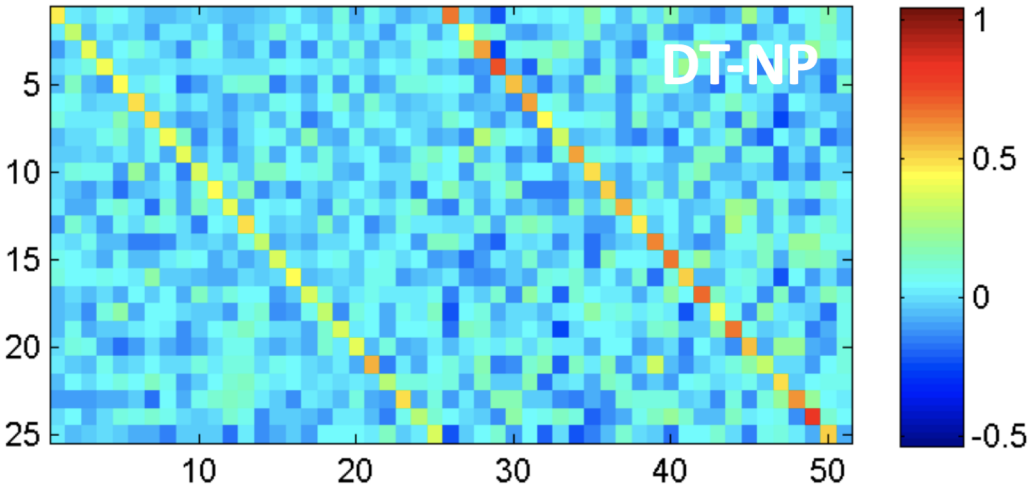
- e.g. “the cat and” NP-CC
  - the most of the semantics have to be found in “the cat”



SU-RNN / CVG

- The models are learning which children of a phrase are actually the important one

- e.g. “the cat” DT-NP
  - the most of the semantics have to be found in “cat”



Analysis of resulting vector representations

- All the figures are adjusted for seasonal variations
  - All the numbers are adjusted for seasonal fluctuations
  - All the figures are adjusted to remove usual seasonal patterns

- Knight-Ridder wouldn't comment on the offer
  - Harsco declined to say what country placed the order
  - Coastal wouldn't disclose the terms

- Sales grew almost 7% to $UNK m. from $UNK m.
  - Sales rose more than 7% to $94.9 m. from $88.3 m.
  - Sales surged 40% to UNK b. yen from UNK b.

v3: Compositionality Through Recursive Matrix-Vector Spaces

[Socher, Huval, Bhat, Manning, & Ng, 2012]

Before:

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

two words didn't interact with each other in their meanings

- One way to make the composition function more powerful was by untying the weights W (SU-RNN)

- But what if words act mostly as an operator, e.g. "very" in "very good".
- Proposal: A new composition function
  - takes in the meaning of good and return a meaning of very good

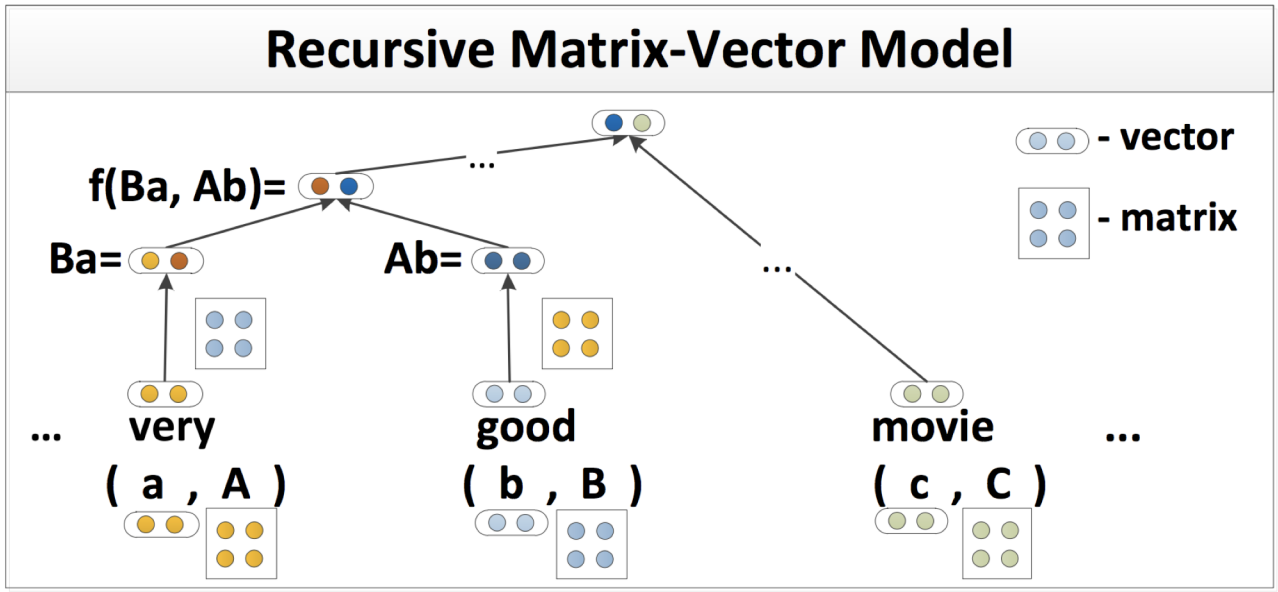
Compositionality Through Recursive Matrix-Vector Spaces

vetcor embedding: pre-trained word embedding

matrix embedding: $X = I + \epsilon$

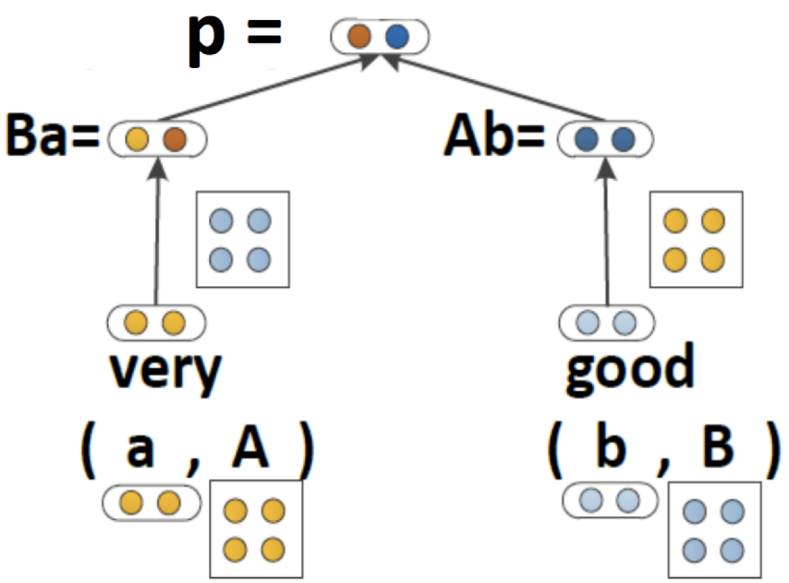
$$p = \tanh\left(W \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right)$$

$$p = \tanh\left(W \begin{bmatrix} C_2 c_1 \\ C_1 c_2 \end{bmatrix} + b\right)$$



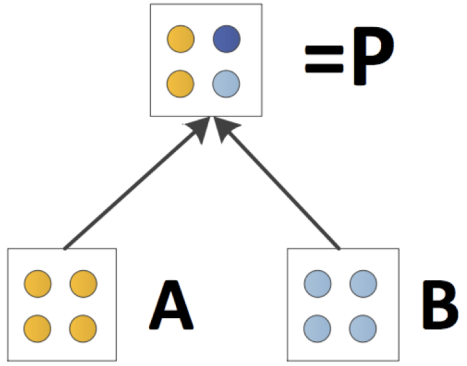
Matrix-vector RNNs

$$p = f\left(W \begin{bmatrix} Ba \\ Ab \end{bmatrix}\right)$$



$$P = g(A, B) = W_M \begin{bmatrix} A \\ B \end{bmatrix}$$

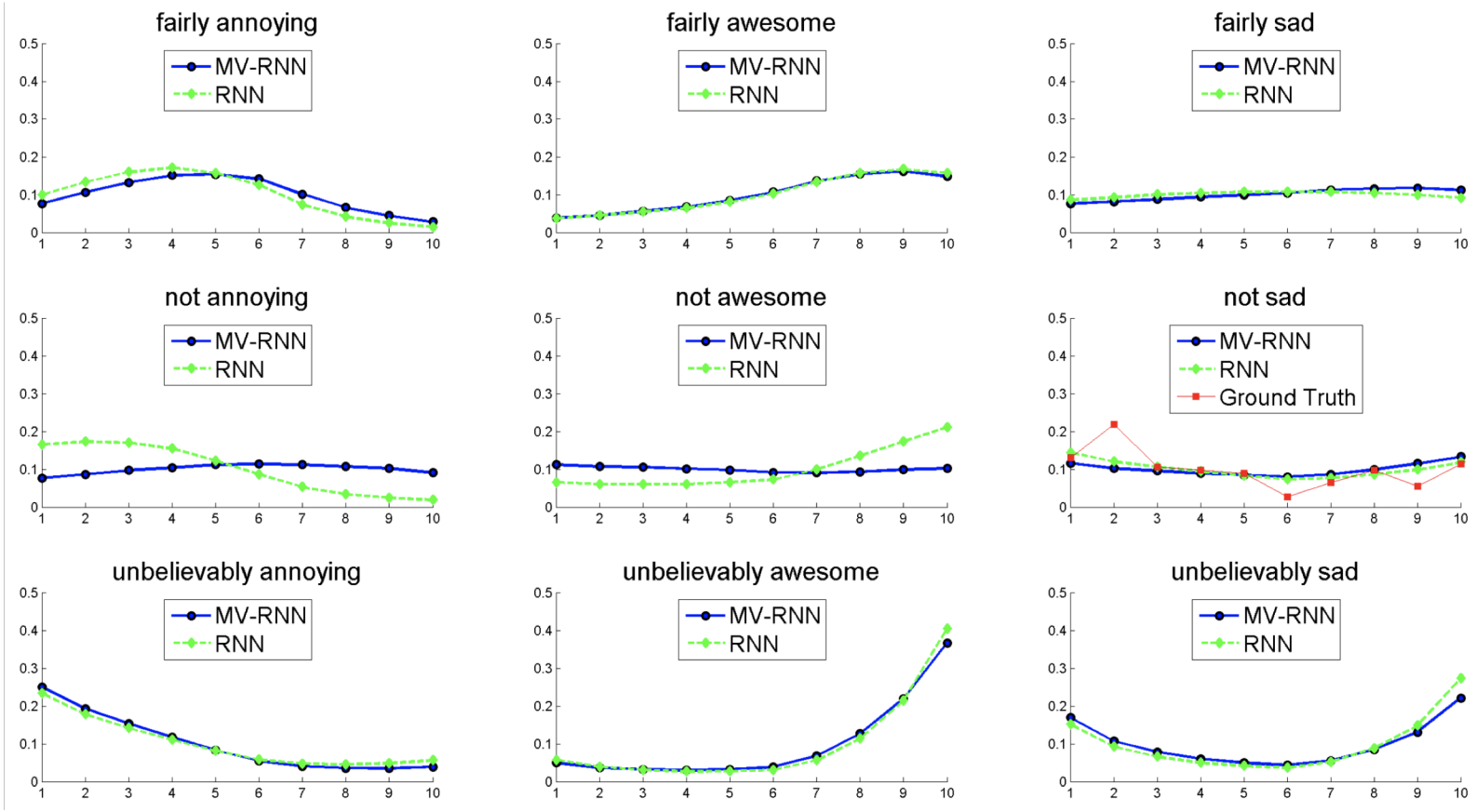
$$W_M \in \mathbb{R}^{n \times 2n}$$



Predicting Sentiment Distributions

Analyze the semantics of an operator modifying another word

- Good example for non-linearity in language



negative to positive

Classification of Semantic Relationships

- Can an MV-RNN learn how a large syntactic context conveys a semantic relationship?

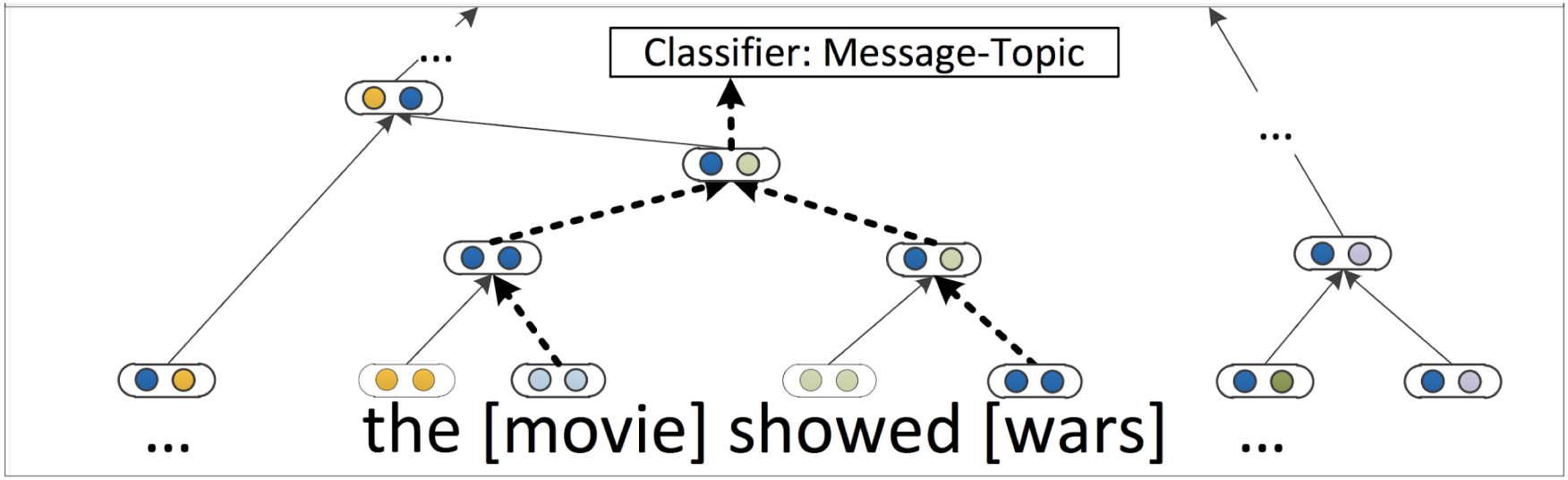
  My [apartment]$_{e1}$ has a pretty large [kitchen]$_{e2}$
  → component-whole relationship (e2,e1)

Classification of Semantic Relationships

- Can an MV-RNN learn how a large syntactic context conveys a semantic relationship?

  My [apartment]$_{e1}$ has a pretty large [kitchen]$_{e2}$
  → component-whole relationship (e2,e1)

- Build a single compositional semantics for the minimal constituent including both terms



Classification of Semantic Relationships

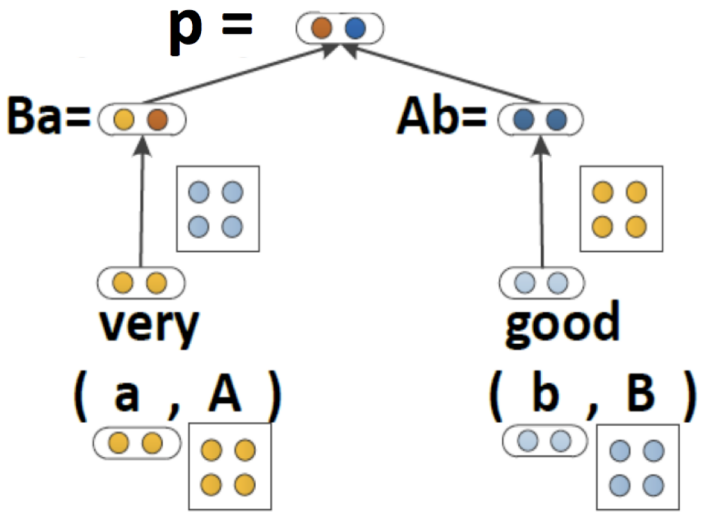
The dataset and evaluation framework is the same as [Hendrickx et al., 2010]

Classifier	Features	F1
SVM	POS, stemming, syntactic patterns	60.1
MaxEnt	POS, WordNet, morphological features, noun compound system, thesauri, Google n-grams	77.6
SVM	POS, WordNet, prefixes, morphological features, dependency parse features, Levin classes, PropBank, FrameNet, NomLex-Plus, Google n-grams, paraphrases, TextRunner	82.2
RNN	–	74.8
MV-RNN	–	79.1
MV-RNN	**POS, WordNet, NER**	**82.4**

Problems of MV-RNN

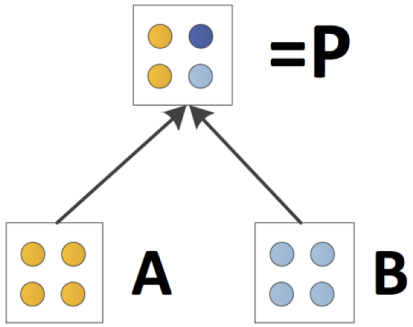
1. introduce a large number of parameters $d \times d \times |V|$
   a. vector size d=1024, matrix size 1024*1024
2. there are not very good ways of building up matrix meaning of the bigger matrix

$$p = f\left(W \begin{bmatrix} Ba \\ Ab \end{bmatrix}\right)$$



$$P = g(A, B) = W_M \begin{bmatrix} A \\ B \end{bmatrix}$$

$$W_M \in \mathbb{R}^{n \times 2n}$$



Beyond the bag of words: Sentiment detection

- Is the tone of a piece of text positive, negative, or neutral?
- Sentiment is sometimes “easy”. Detection accuracy for longer documents ~90%

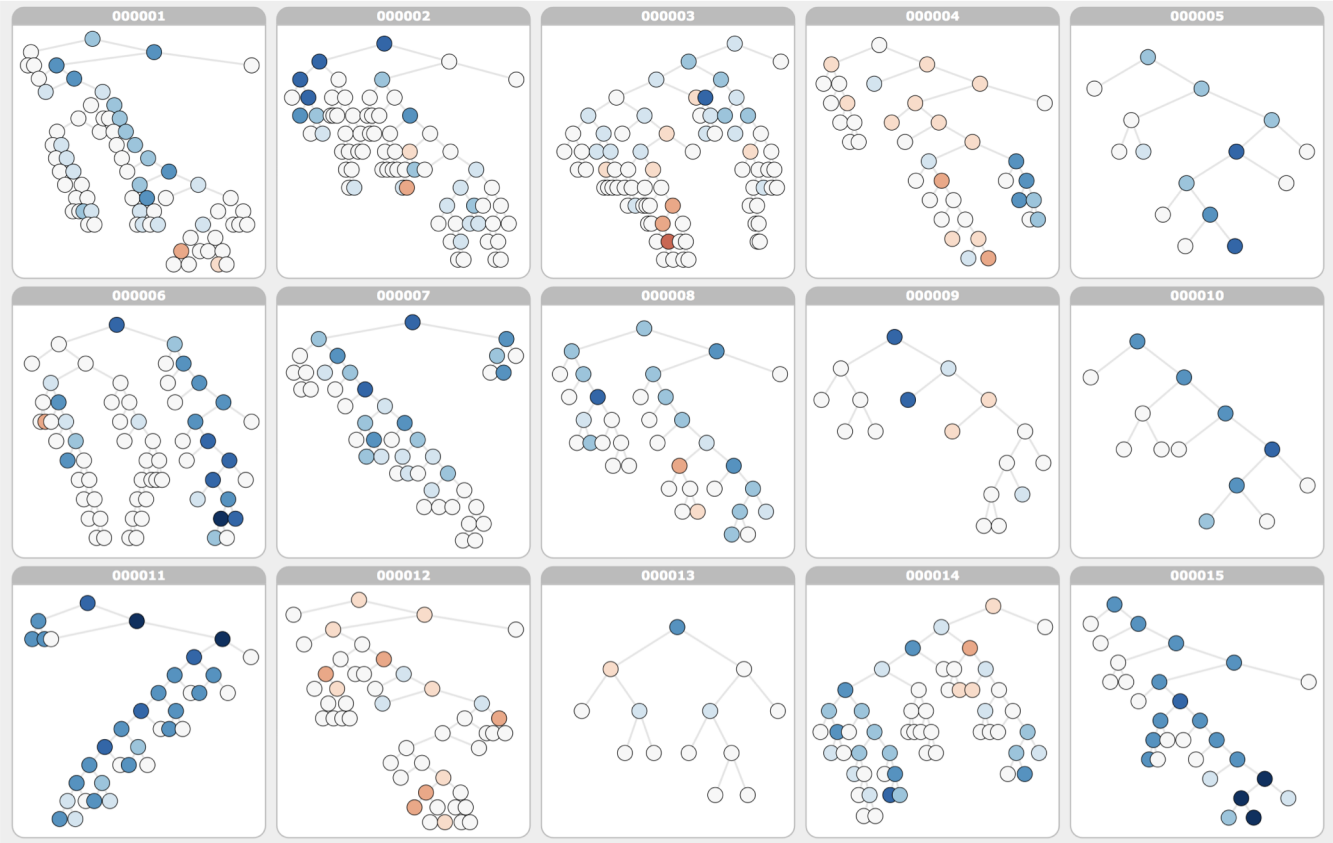
  … … loved … … … … … great … … … … … … impressed … … … … … … marvelous … … … …

- BUT

With this cast, and this subject matter, the movie should have been funnier and more entertaining.

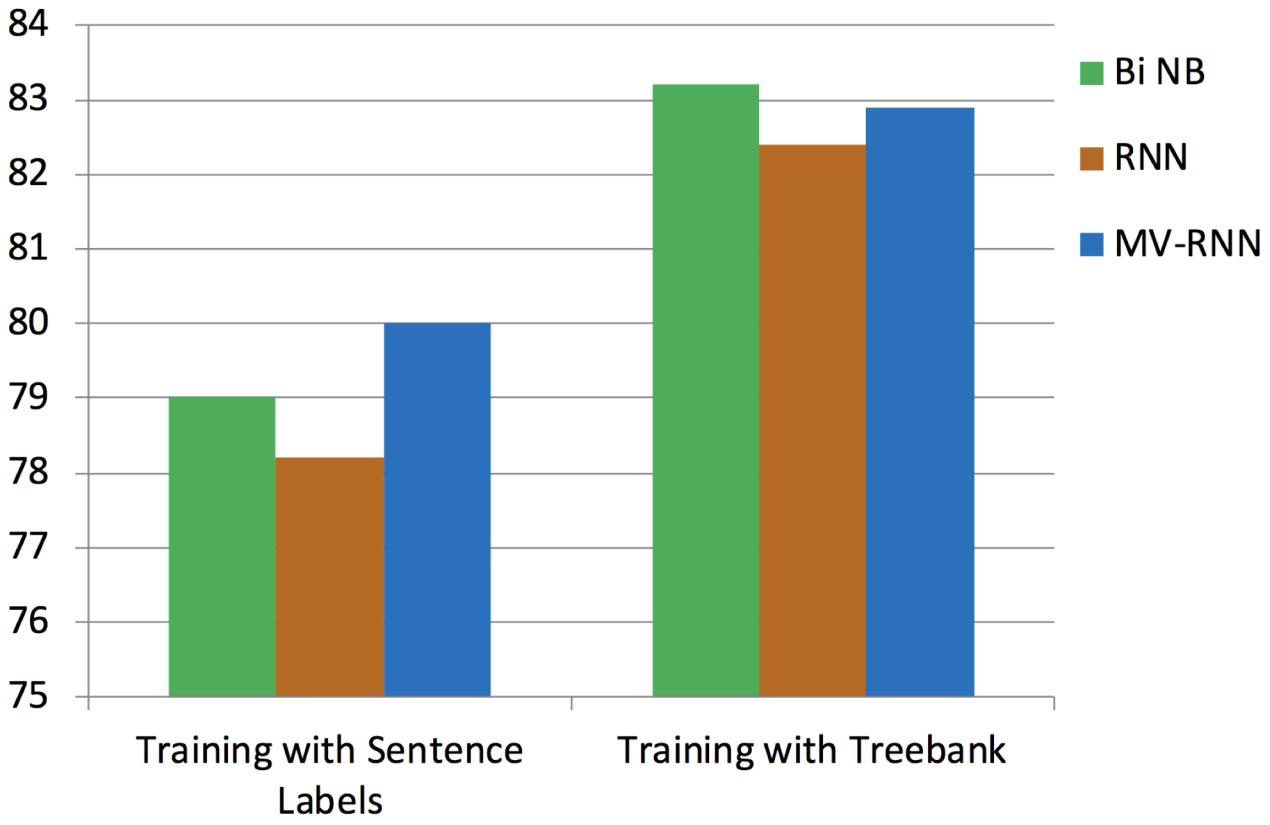
Stanford Sentiment Treebank

- 215,154 phrases labeled in 11,855 sentences
- Can actually train and test compositions



https://nlp.stanford.edu/sentiment/treebank.html

Better Dataset Helped All Models

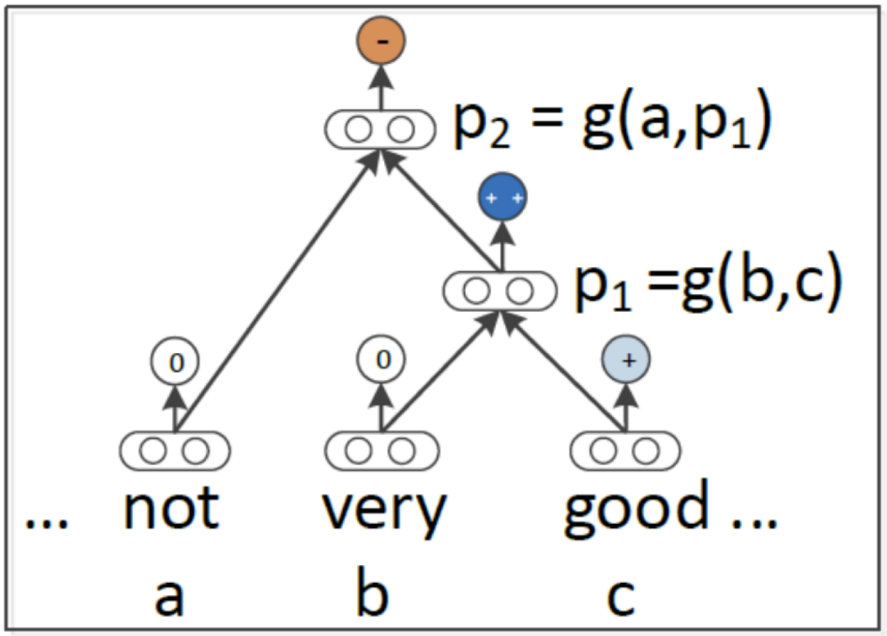


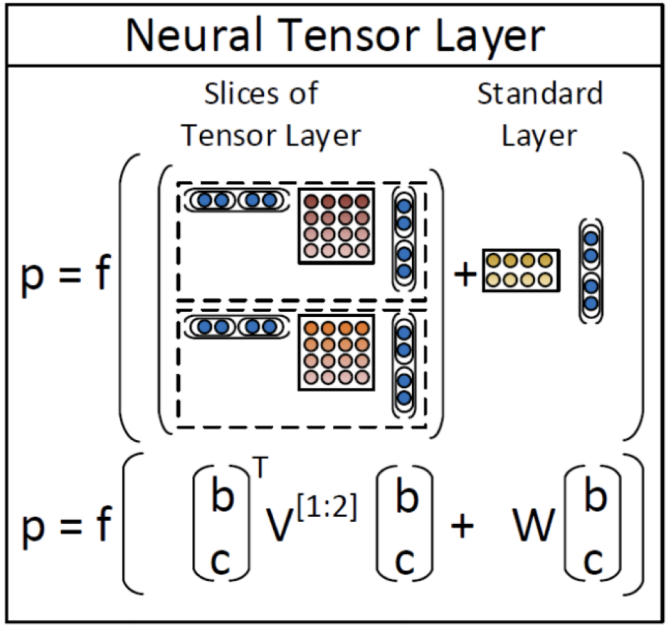
- Hard negation cases are still mostly incorrect
- We also need a more powerful model!

v4: Recursive Neural Tensor Network

[Socher, Perelygin, Wu, Chuang, et al. 2013]

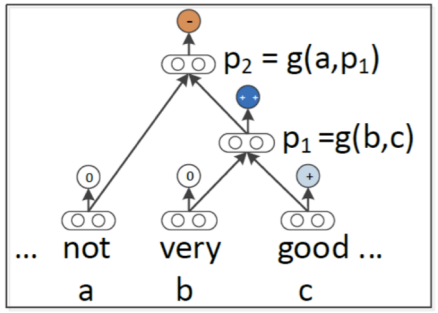
- Less parameters than MV-RNN
- Allows the two word or phrase vectors to interact multiplicatively



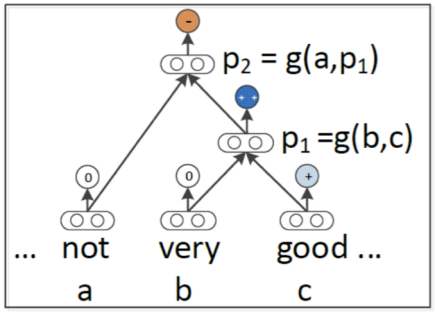


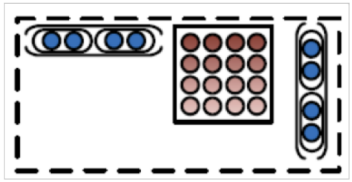
Recursive Neural Tensor Network

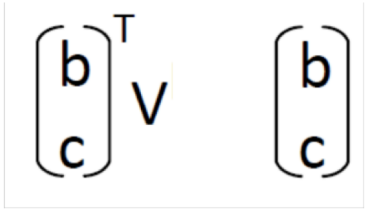
- Idea: Allow both additive and mediated multiplicative interactions of vectors



Recursive Neural Tensor Network

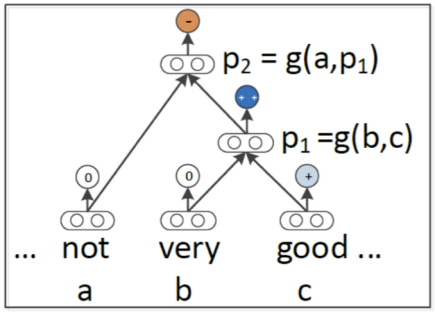


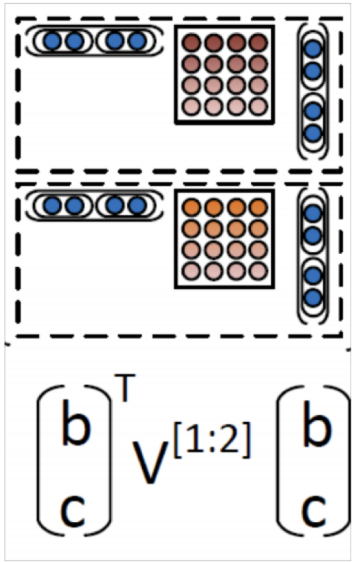




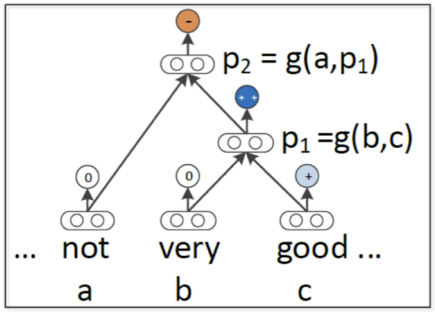
Recursive Neural Tensor Network

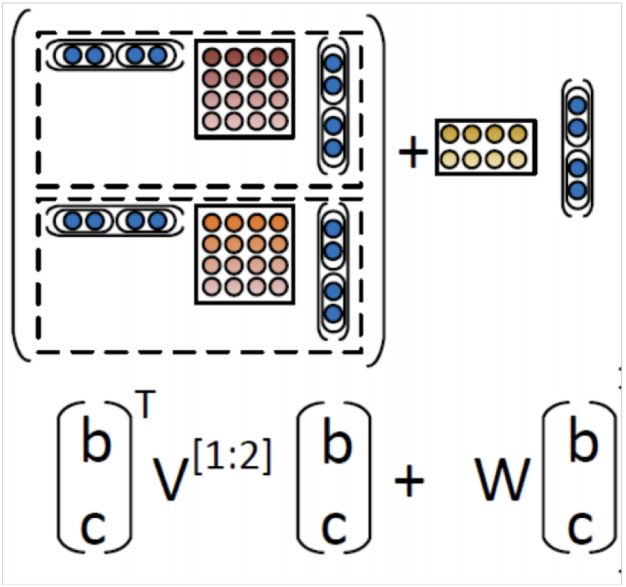
- Idea: Allow both additive and mediated multiplicative interactions of vectors





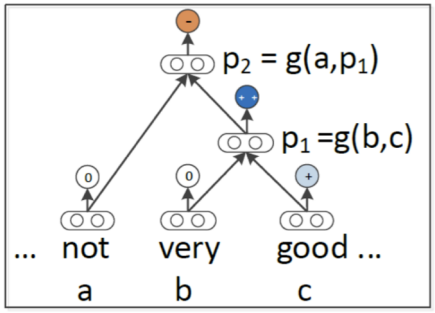
Recursive Neural Tensor Network

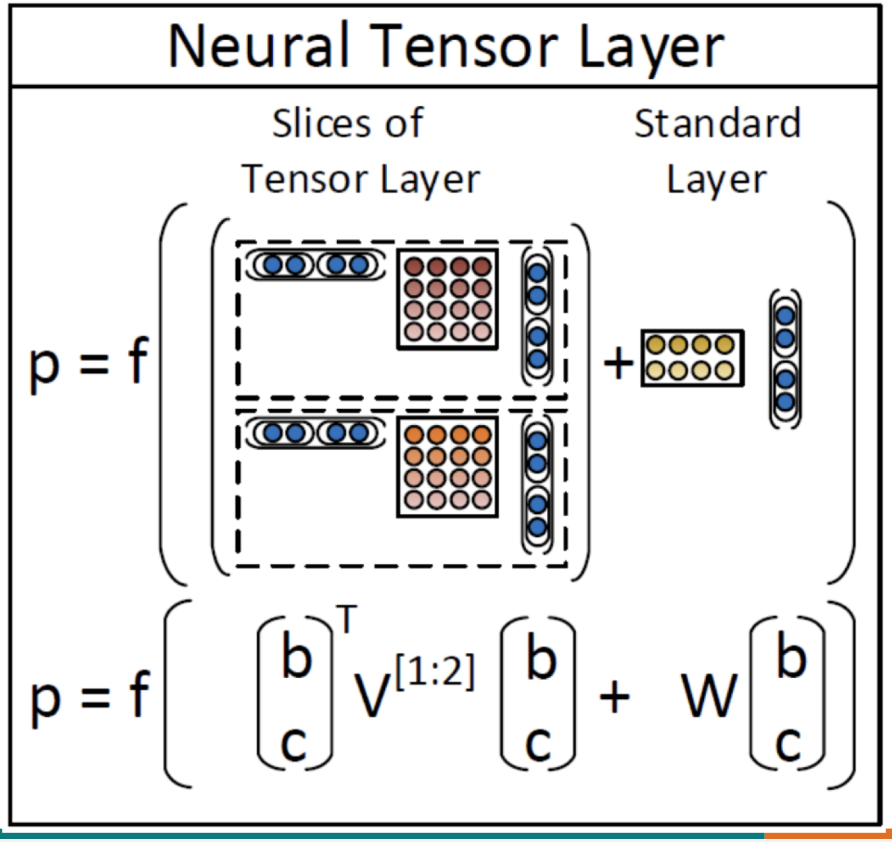




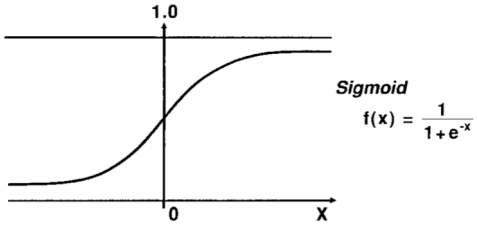
Recursive Neural Tensor Network

- Use resulting vectors in tree as input to a classifier like logistic regression
- Train all weights jointly with gradient descent





$$V^{[1:d]} \in \mathbb{R}^{2d \times 2d \times d}$$



Recursive Neural Tensor Network

- Formally, let $V^{[1:d]} \in \mathbb{R}^{2d \times 2d \times d}$ where $V^{[i]} \in \mathbb{R}^{d \times d}$ indicates each tensor slices
- Then the composition rule $h \in \mathbb{R}^d$ for children (b, c) are given by

$$h = \begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix}; h_i = \begin{bmatrix} b \\ c \end{bmatrix}^T V^{[i]} \begin{bmatrix} b \\ c \end{bmatrix}$$

- and the parent p_1

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

- It reduced the # of parameters from from $d \times d \times |V|$ to $2d \times 2d \times d$

Recursive Neural Tensor Network

- Main advantage over the previous RNN
  - When V is set to 0, directly model input vectors
  - Intuitively, we can interpret each slice of the tensor as capturing a specific type of composition

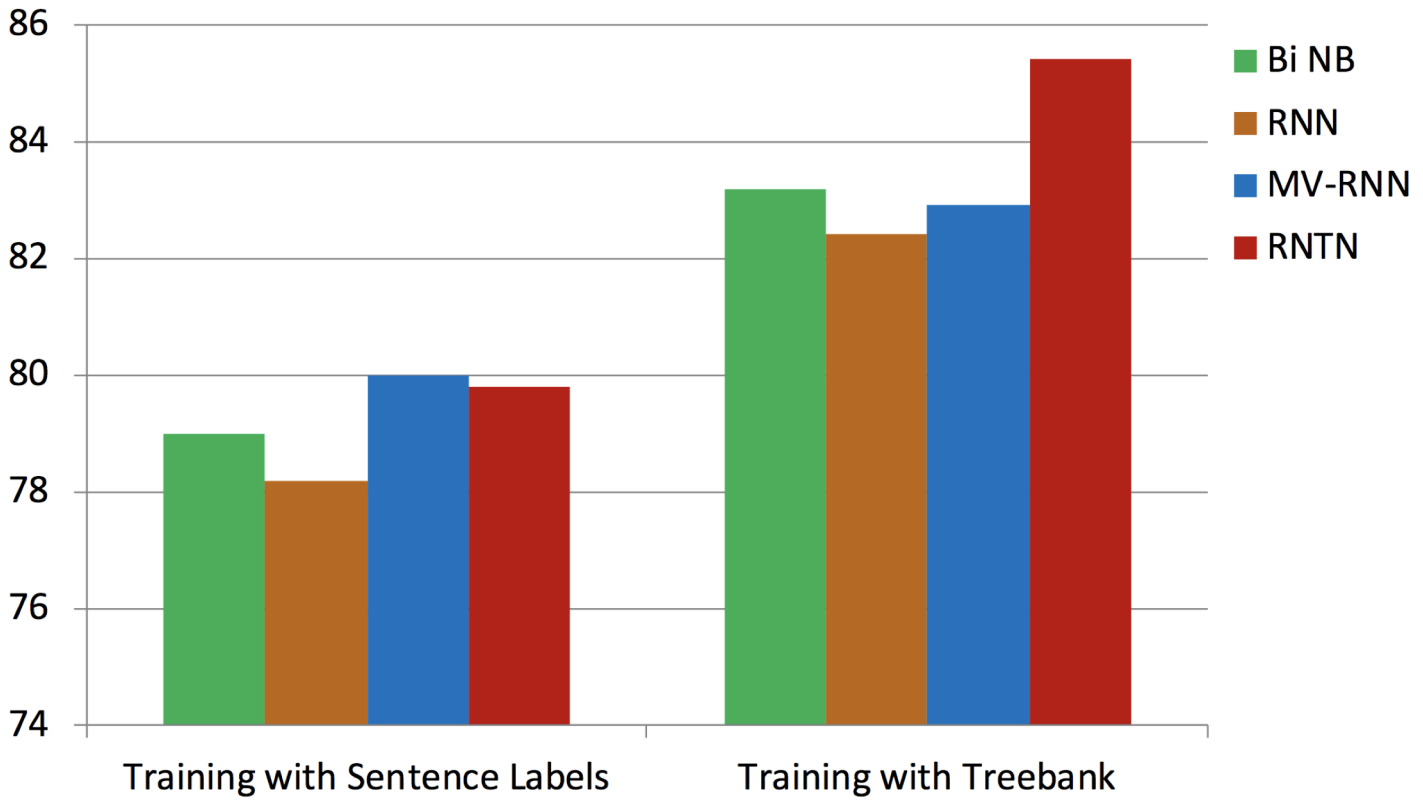
$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix} + W \begin{bmatrix} b \\ c \end{bmatrix}\right)$$

capturing specific type of compositions

standard RNN

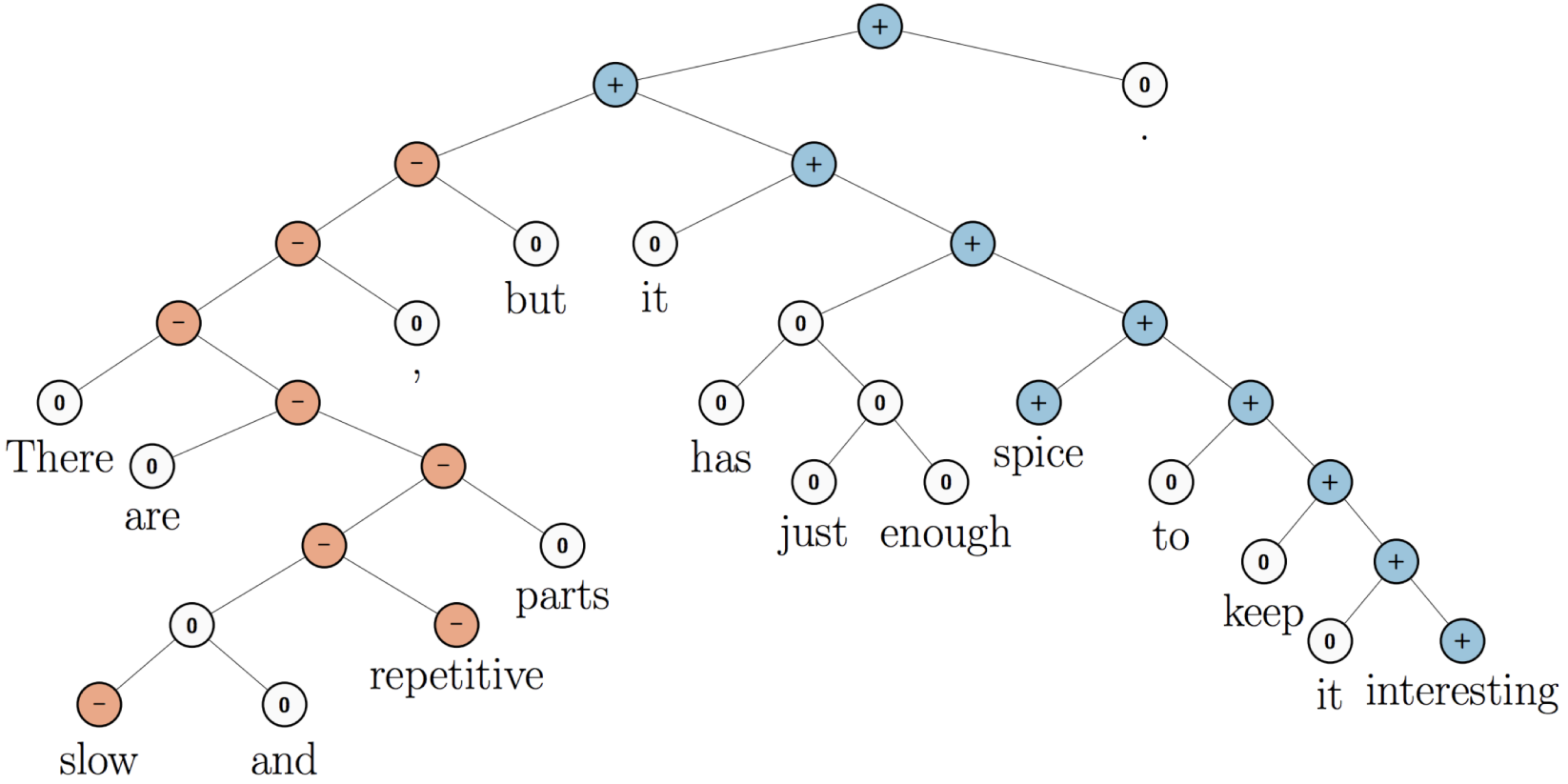
Positive/Negative Results on Treebank

- Classifying Sentences: Accuracy improves to 85.4



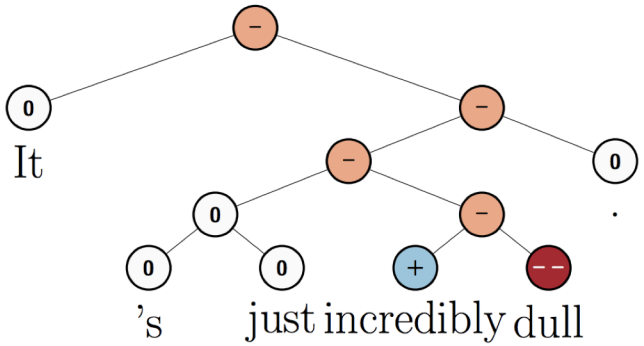
Experimental Results on Treebank

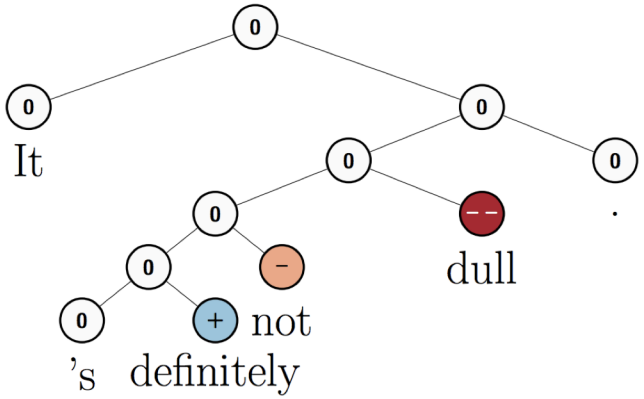
- RNTN can capture constructions like “X but Y”
- RNTN accuracy of 72%, compared to MV-RNN (65%), biword NB (58%) and RNN (54%)

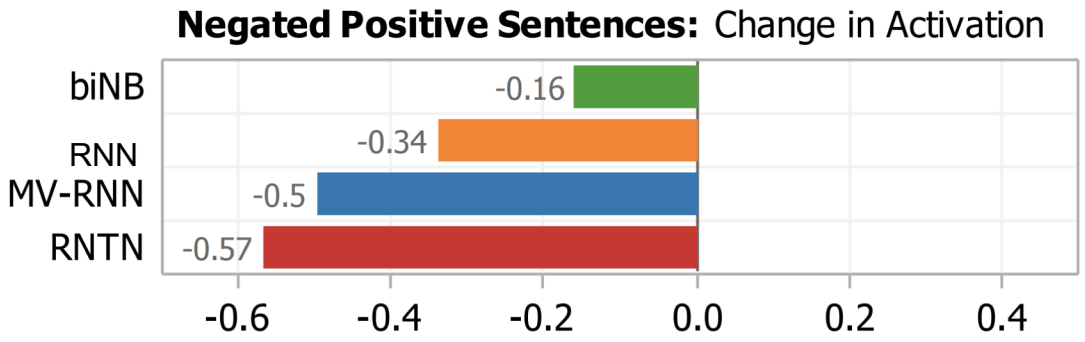


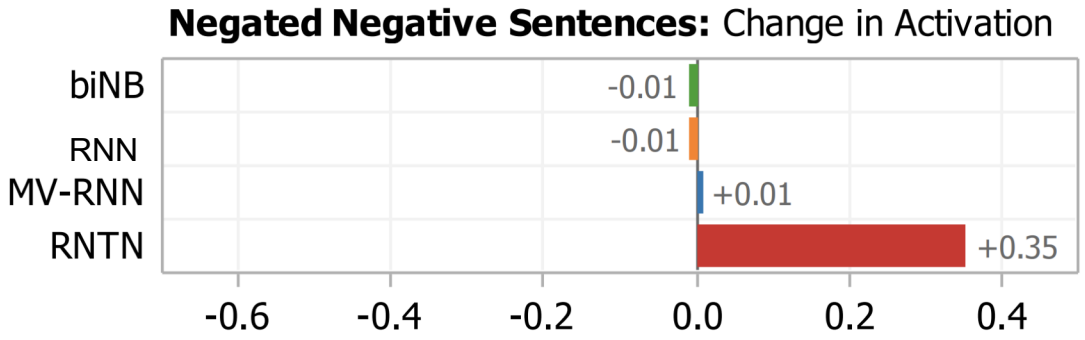
Negation Results

- When negating negatives, positive activation should increase!









Other more advanced/newer model

- Improving Deep Learning Semantic Representations using a TreeLSTM
  - Tai et al., ACL 2015; also Zhu et al. ICML 2015
- QCD-Aware Recursive Neural Networks for Jet Physics
  - Louppe, Cho, Becot, Kyle Cranmer 2017
- Tree-to-tree Neural Networks for Program Translation
  - Chen, Liu, Song, NeurIPS 2018
- etc.

Summary

- Constituency Parsing
  - Constituency Parsing Tree

- Standard Recursive Neural Networks

- More complex TreeRNN units
  - Syntactically-Untied RNN (SU-RNN)
  - Compositionality Through Recursive Matrix-Vector Spaces (MV-RNN)
  - Recursive Neural Tensor Network (RNTN)

STEVENS

INSTITUTE *of* TECHNOLOGY

THE INNOVATION UNIVERSITY®

stevens.edu

Thank You