Natural Language Processing
CSE 325/425

Sihong Xie

Lecture 21:

- Recussive neural networks

Motivation

- Tweet sentiment analysis
  - customer satisfaction analysis
  - voting in election

Loves the German bakeries in Sydney. Together with my imported honey it feels like home	Positive
@VivaLaLauren Mine is broken too! I miss my sidekick	Negative
Finished fixing my twitter...I had to unfollow and follow everyone again	Negative
@DinahLady I too, liked the movie! I want to buy the DVD when it comes out	Positive
@frugaldougal So sad to hear about @OscarTheCat	Negative
@Mofette briliant! May the fourth be with you #starwarsday #starwars	Positive
Good morning thespians a bright and sunny day in UK, Spring at last	Positive
@DowneyisDOWNEY Me neither! My laptop's new, has dvd burning/ripping software but I just can't copy the files somehow!	Negative

Motivation

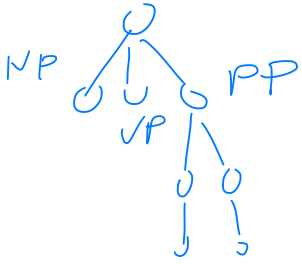
- Representing semantics of a sentence.
  - basing on individual word vectors are not sufficient.
    - negation
    - word order are lost
  - recurrent NN is sequential and ignores constituency structures
  - need to follow the constituency structure, which is recursive.
    - scopes with different
      - lengths (unigram, bigram, trigram, …, whole-sentence).
      - synatactic type (NP, VP, PP, …).

Average word vectors in a sentence.

--- not bad ---

--- bad -- not good --

[start, end]

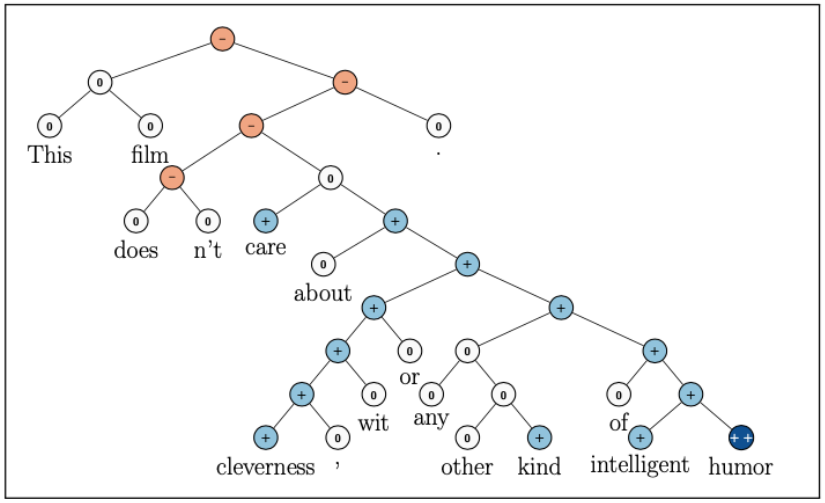


Question: Given a parsing tree of a sentence $\vec{O} = [o_1, \ldots, o_T]$

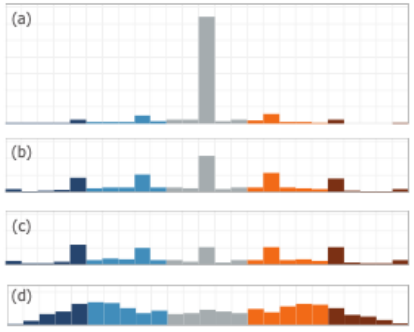
derive semantic vector of $\vec{O}$

Stanford sentiment treebank

- Hand-labeled phrases derived from syntactic parsing trees.
  - parsing tree with sentiments on nodes;
  - Some statistics
    - longer phrases have more distinguishable orientations



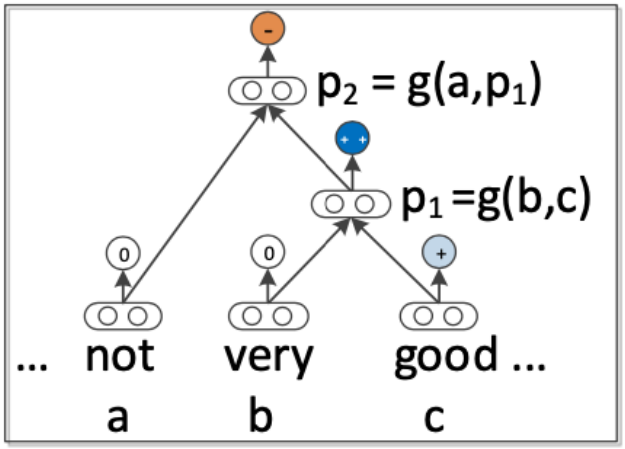
Distributions of sentiment values for (a) unigrams, (b) 10-grams, (c) 20-grams, and (d) full sentences.



Recussive neural networks

(invented 1990's)

- Compute a vector for each node of a parsing tree.
- Compute parent vector based on children vectors
  - using the same non-linear function *g*;
  - bottom-up and recursive;
  - parsing trees define neural network architectures:
    - different from Recurrent neural network and MLP.
  - Trainable parameters: word vectors and *g*.



$g(\vec{u}, \vec{v}; \vec{\theta}_g) \rightarrow \vec{w}$

- Each node can be classified with sentiments, using a multi-class logistic regression model $y^a = \text{softmax}(W_s a)$

$y^a \in \{--, -, 0, +, ++\}$

5 classes

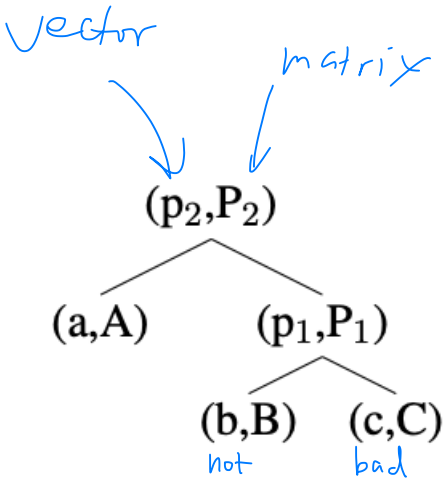
Variants of Recursive NN

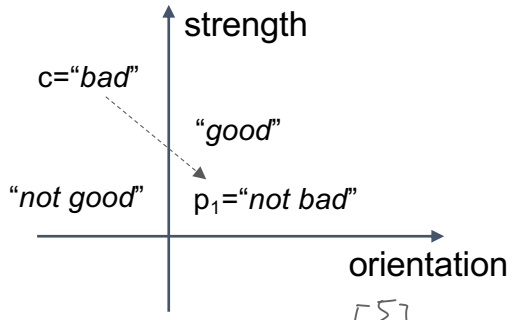
- Matrix-Vector Recursive NN
  - Each node has a vector and a matrix
    - vector: word/phrase semantics
    - matrix: linear operator mapping from vector to vector
  - Computing the parent representation

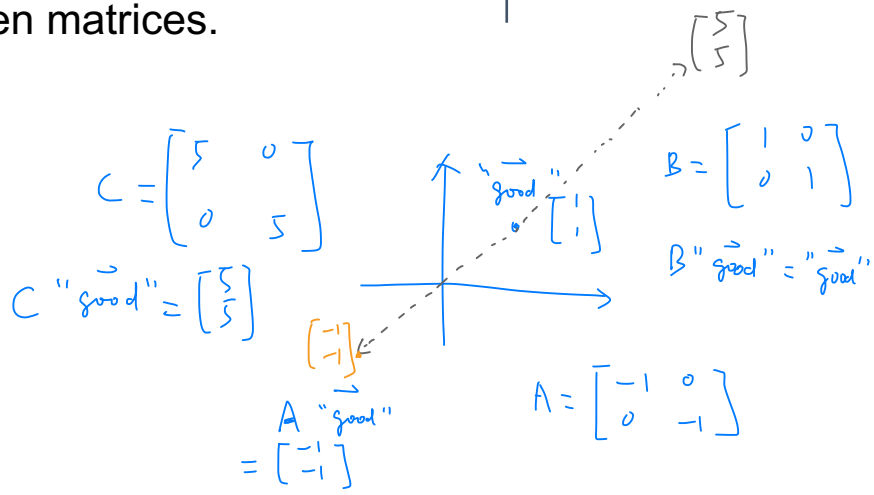
$$p_1 = f\left(W\begin{bmatrix} Cb \\ Bc \end{bmatrix}\right), P_1 = f\left(W_M\begin{bmatrix} B \\ C \end{bmatrix}\right)$$

tanh

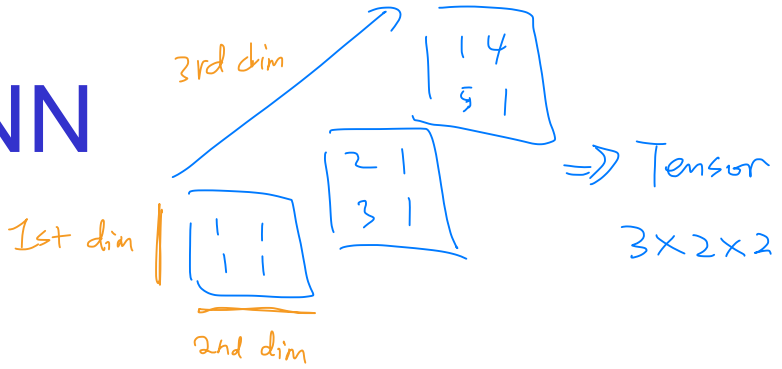
    - composing semantic vector by children interactions;
    - composing operator matrix from children matrices.
  - Too many parameters.



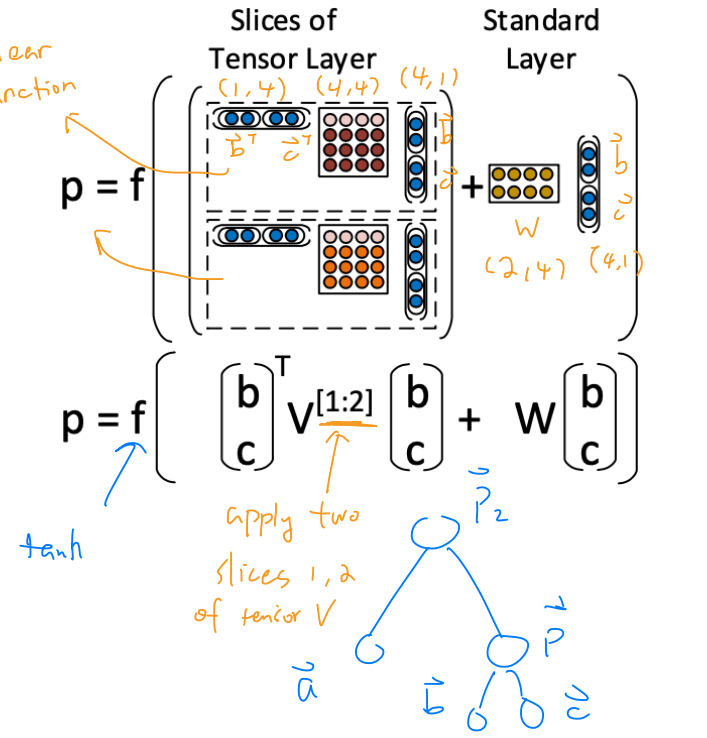




Variants of Recursive NN



- Recursive neural "tensor" network
  - What is a tensor?
    - just a fancy term that means a stack of multiple matrices of the same shape.
    - think of a tensor as a 3D array (in general can have more dimensions).
  - Each slice of the parameter tensor is a trainable bilinear function to combine two sementic children vectors.
    - go through each slides to combine children vectors into vector *h*;
    - capture multiple ways of interactions between children;
    - Use the same parameter recursively when going up the tree.



Variants of Recursive NN

- Model training.

log-likelihood.

$$E(\theta) = \sum_i \sum_j t_j^i \log y_j^i + \lambda \|\theta\|^2$$

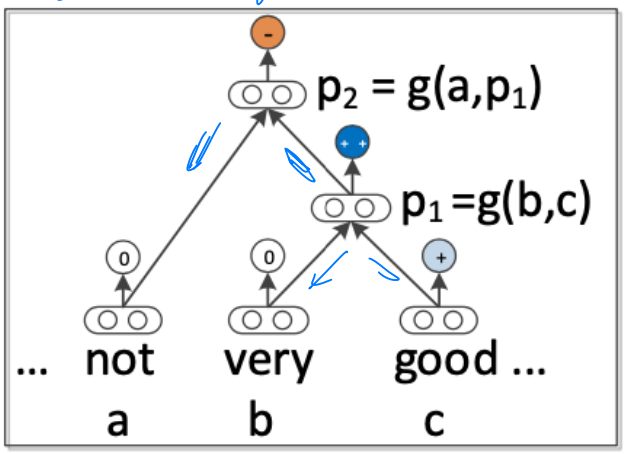
→ going through all nodes of tree i.

Going through all trees

L-2 regularization to avoid overfitting

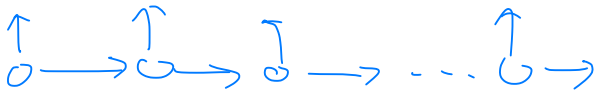
$y_j^i = \text{softmax}(W_s \vec{a}_j^i)$

$\vec{a}_j^i$: vector of the j-th node of the i-th tree.



$\max_\theta E(\theta)$ $\theta = \{\text{word vectors}, W_s, V\}$

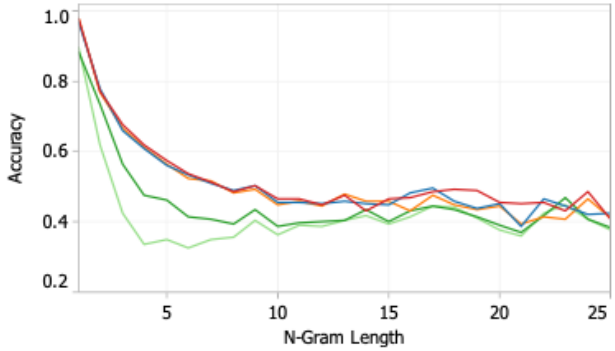
$\theta^{(t+1)} \leftarrow \theta^{(t)} + \eta \frac{\partial E(\theta)}{\partial \theta}$

RNN : 

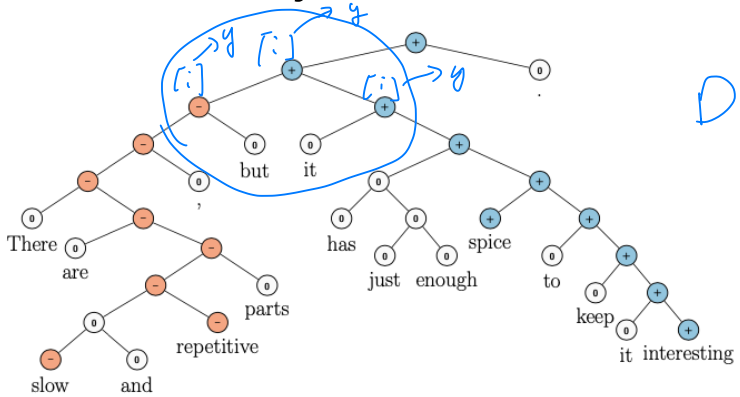
(BPTT)

Results

- Longer phrases/sentences are harder to predict



- Handling contrastive conjunction *but*



Dynamic Programming

Results

- Handling negations of positive and negative sentences.