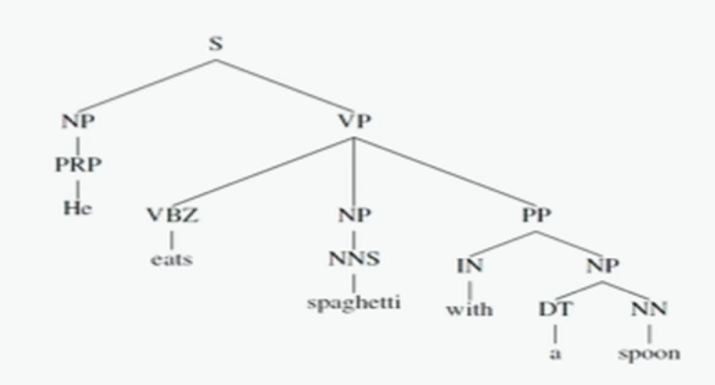
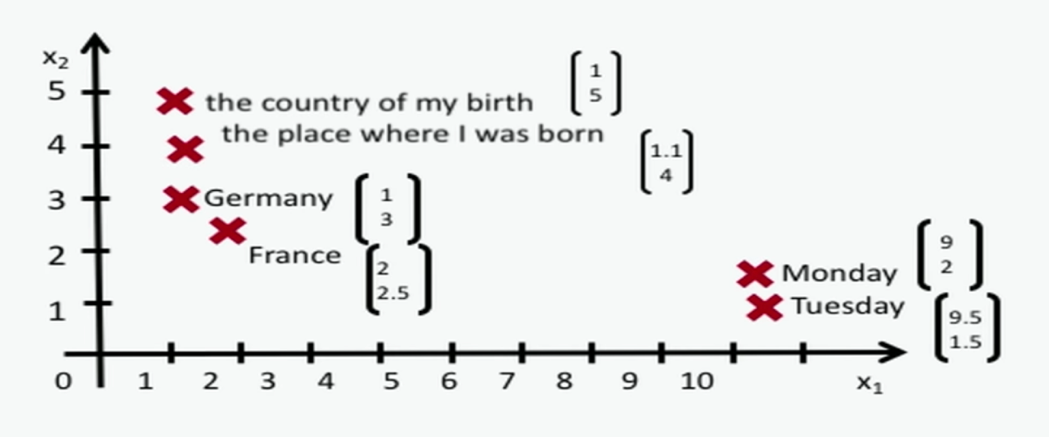
Lecture 14 | Tree Recursive Neural Networks and Constituency Parsing

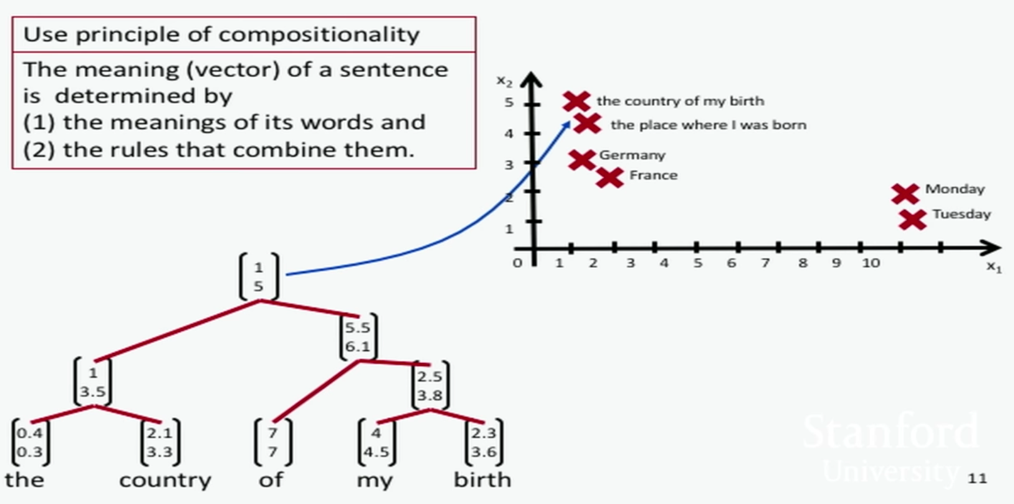
* Humans have this ability, which you see through language, to have recursion. Recursion is a natural way to describe language
* Noun phrase containing a noun phrase containing a noun phrase



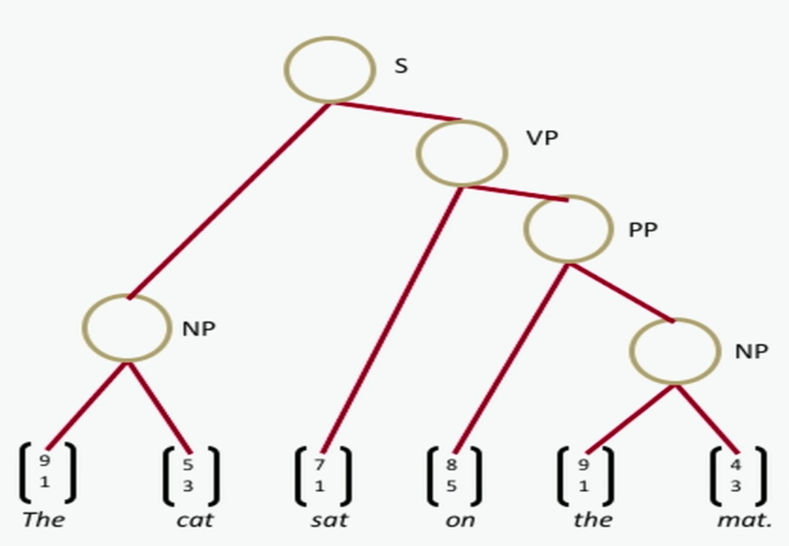
* Recursion is useful when there’s some tasks that require you to refer to specific phrases
  + For example: John and Jane went to a big festival. They enjoyed the trip and the music there
    - “They” refers to “John and Jane”
    - “the trip” refers to “went to a big festival”
    - “there” refers to “big festival”
* We can represent the meaning of longer phrases through mapping them into the same vector space



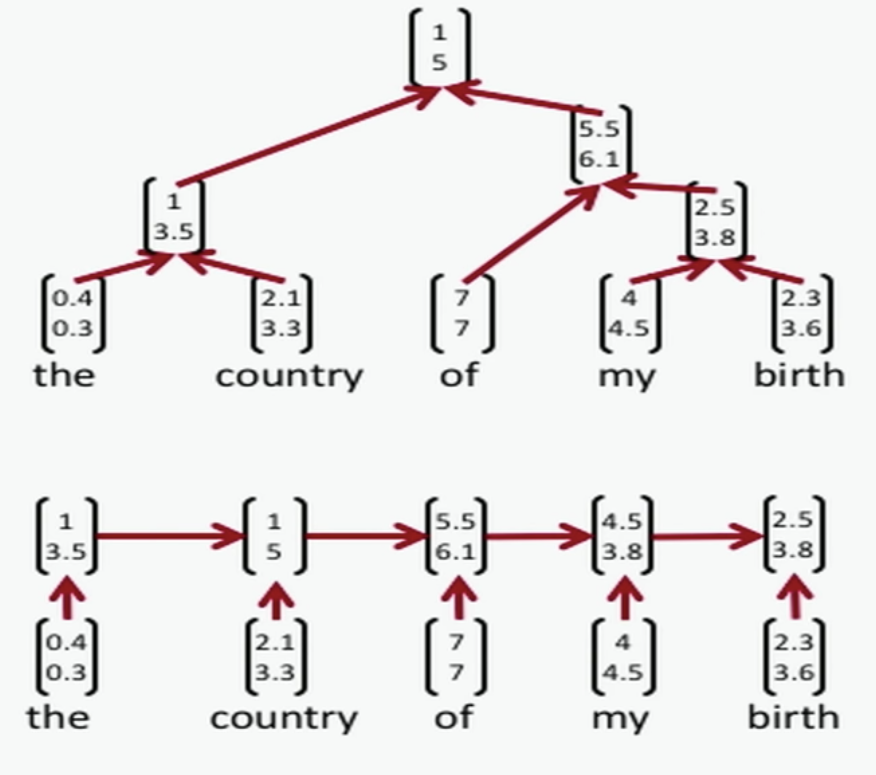
* + How should we map phrases into vector space?



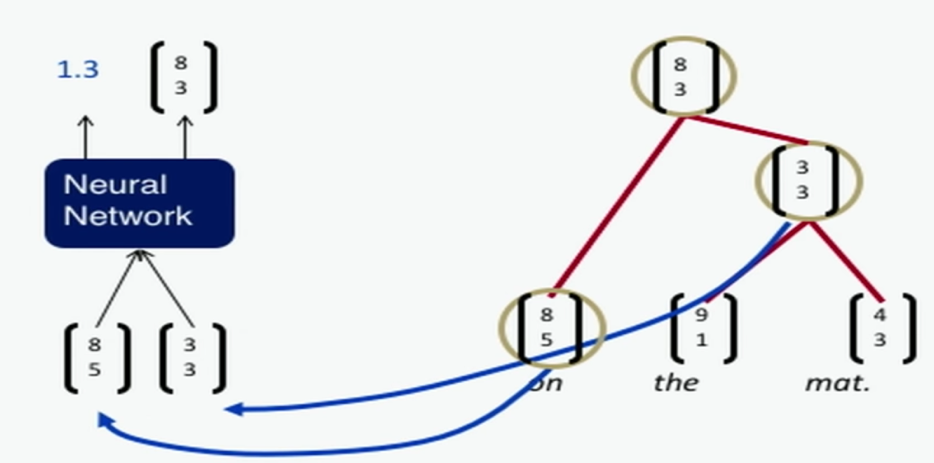
* Constituency sentence parsing
  + We start from individual word vectors and start building phrase structure and end up with a syntactic phrase structure of the tree



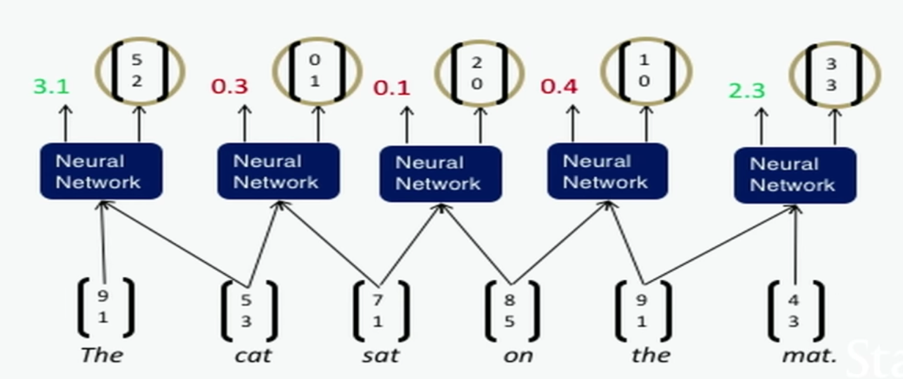
* Difference between recursive and recurrent neural networks



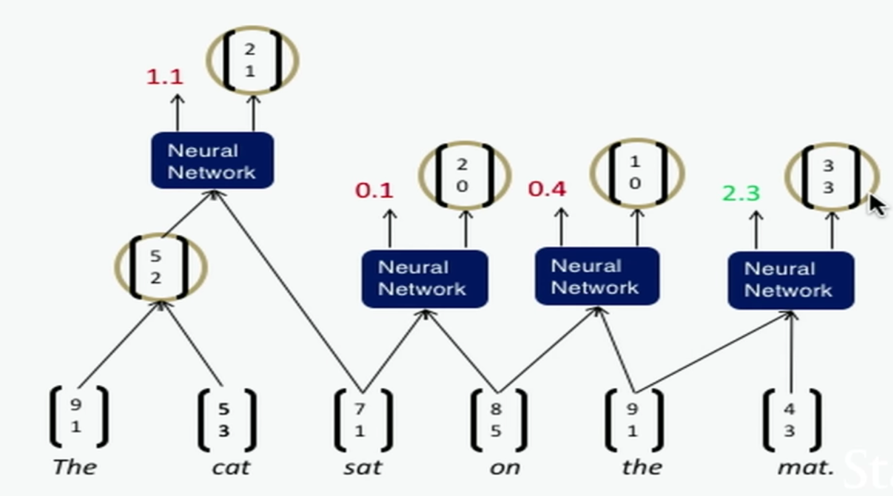
* + Recursive neural nets require a parser to get tree structure. A recurrent neural net cannot capture phrases without prefix context and often capture too much of last words in final vector
* Tree RNNs compute compositional vectors for grammatical phrases only whereas a CNN computes vectors for every possible phrase, regardless of whether it makes grammatical sense (you don’t need a parser tho)!
* Recursive NN for structure prediction
  + When we want to build a representation of a larger unit, we want to take the representation of its children and put them into some neural network and output two things, a vector of the meaning of the larger unit and a score to measure the quality of constituent

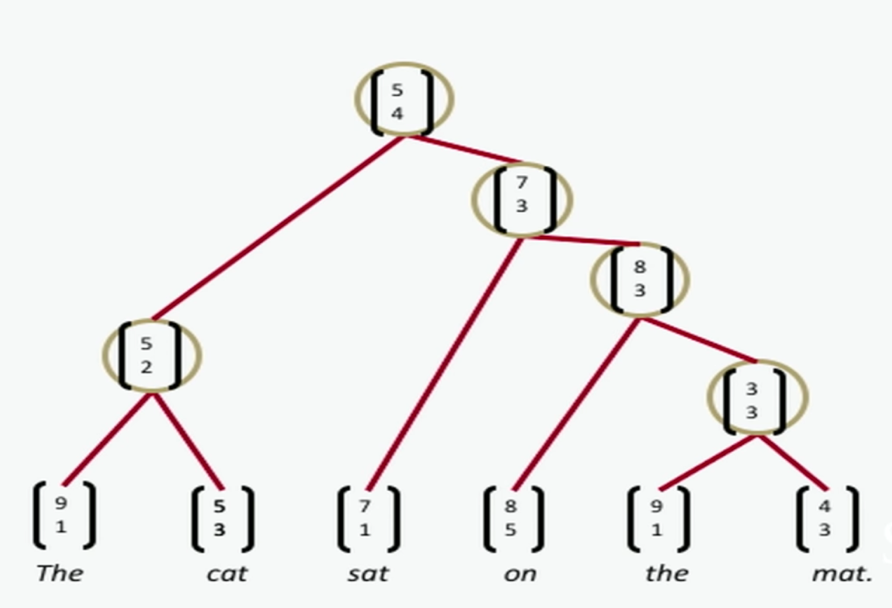


* The two outputs allow us to parse a sentence as follows!

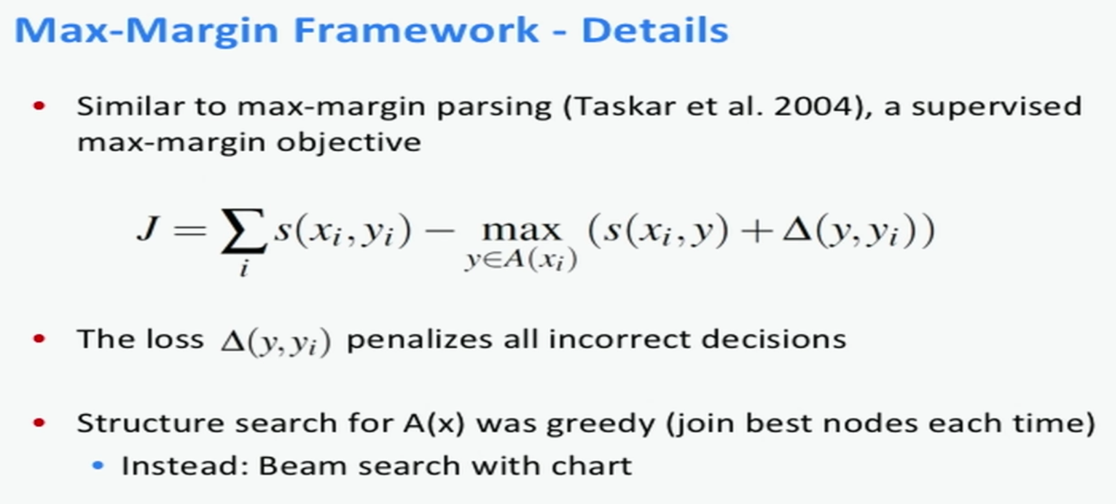


* + As shown as above, now that we have a score metric, we can choose the highest score that tells us the best phrase together (hard commit to those). Repeat this process and you will have the following:

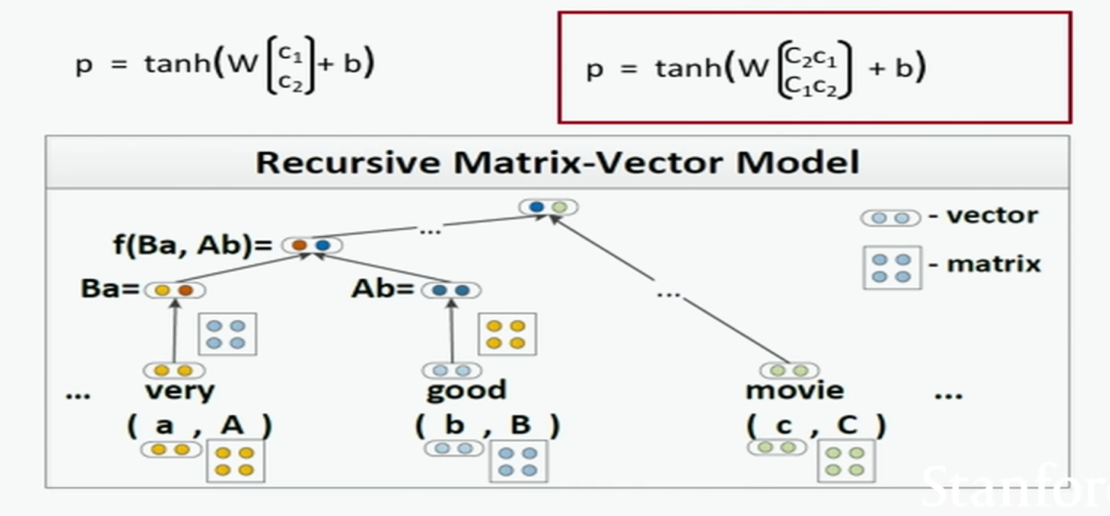




* + The score of a tree is computed by the sum of the parsing decision scores at each node



* + Backpropagation through tree structure is similar to the normal backprop except the following differences:
    - Sum derivatives of W from all nodes (like RNN)
    - Split derivatives at each node (for tree)
    - Add error messages from parent + node itself
* Compositionality through recursive matrix-vector spaces
  + One way to make the composition function more powerful was by unying the weights W – proposing a new composition function



* Classification of semantic relationships
  + Can an MV-RNN learn how a large syntactic context conveys a semantic relationship
  + E.g. My [apartment] has a pretty large [kitchen], where e1 = apartment and e2 = kitchen
  + The relationship (e2, e1) is a component-whole relationship