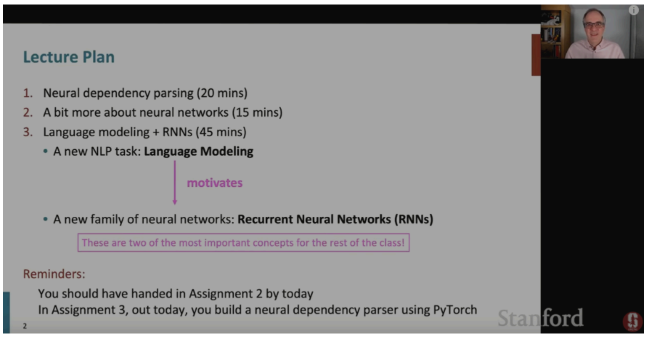
Notes

02 March 2024 07:19

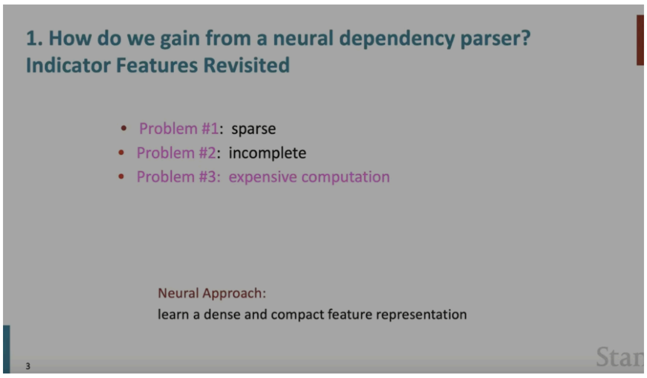
Lecture Outline:



Neural Dependency Parser:

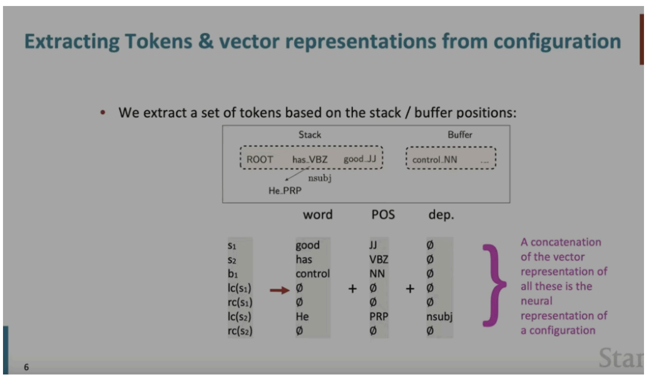
Limitations of Symbolic Dependency parsing (cont. From Lecture – 4)

- Features are generated in a rule based manner, hence they a sparse, Incomplete (difficult to set rules for each sentence in the training data) and involve extensive computations



Symbolic to Neural Classifier:

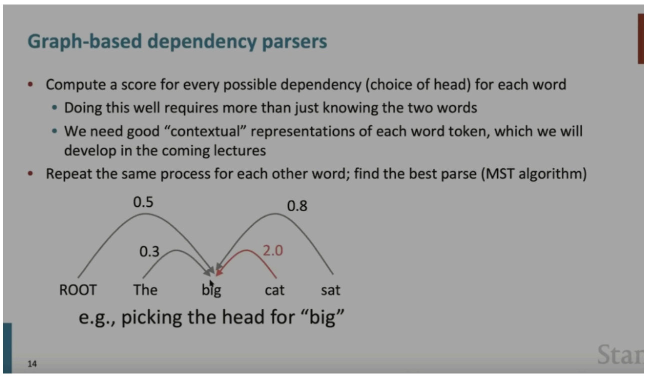
- Sparse symbolic features are replaced by dense vectors of each words
- The Stack – buffer structure is still retained. However the classification Whether to form a dependency or not is now done by a Softmax classifier
- Two step approach:
  - Distributed representations of words + Part of Speech words + dependency labels are represented as vectors



  - Using Softmax classifier

Graph-based dependency Parser:

- Moving on from transition based dependency parsing: Until now we are parsing the sentence from left to right to find the dependencies.
- How about computing every possible dependency in between all the words of the sentence? --- Graph based approaches



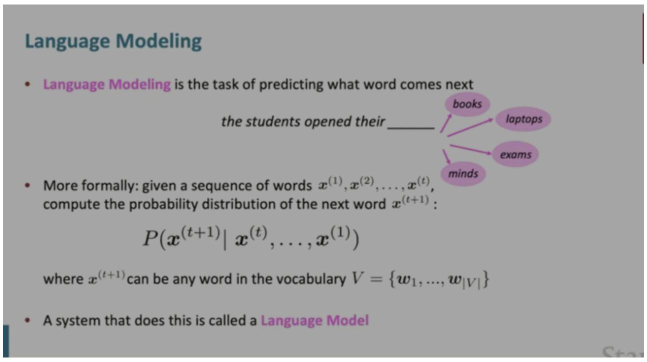
Concepts behind Neural Networks:

This part discussed following concepts: (approx. 15:00 to 45:00 in the video)
- L2 Regularization
- Dropout
- Activation Functions
- Weight Initialization
- Learning rate

Language Models and RNN:

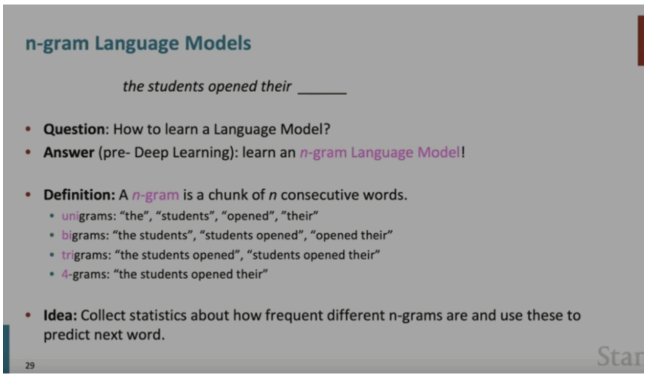
What is language modelling?

- Task of predicting the next word.
- Mathematically, predict the probability of a word occurring next given the context of N previous words



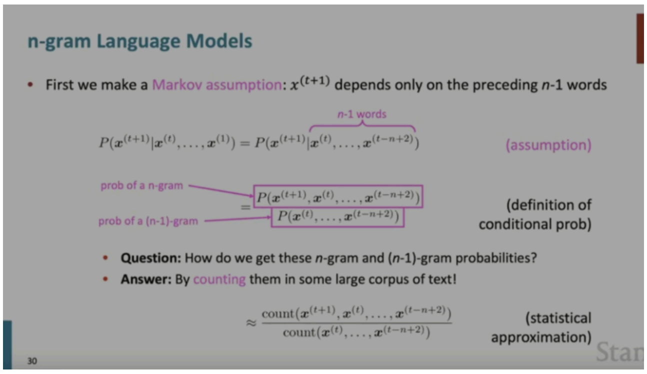
N-gram language Models:

- Models of pre-deep learning era.
- n-grams means collection of N consecutive words-
- Based on how frequent different combinations of word appear

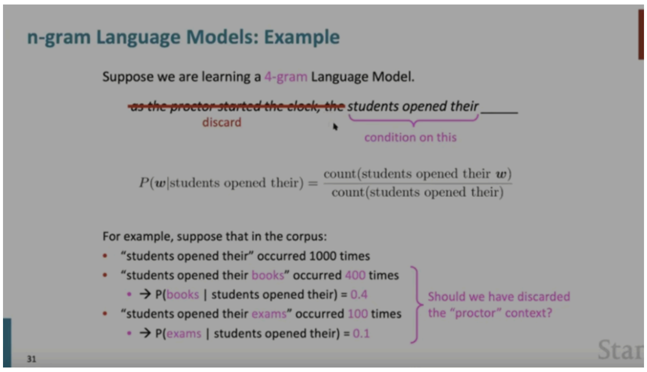


Use Markov Models:

- Markov assumption – a next sample depends on only last (n-1) samples
  Turns this problem into conditional probability problem.
- Order of the Markov model – the "n" in the n-gram

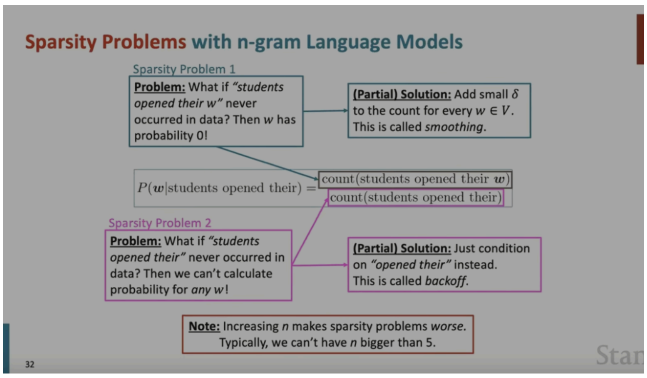


- An example of 4-gram model:
  Disadvantage of n-gram:
  - As we discarded everything before "n" Words, we lost the context of this sentence.
  - The output then depends on stats of the dataset and not context of this sentence.



Sparsity Problem in n-gram models:

- Sparsity – absence of context words from the dataset.
  e.g. Students opened their w – not occurring in data – numerator = 0
  e.g.2 students opened their – not occurring in data – denominator = 0
- Two solutions:
  - Smoothing when numerator is 0.
  - Backoff when denominator is 0.
- Increasing N makes the problem worse: I think we need graph based NN



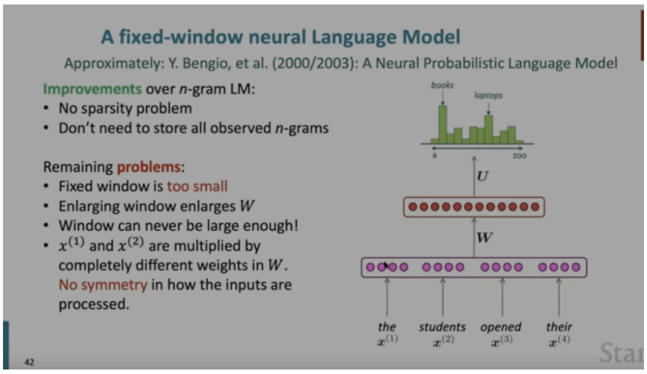
Storage Problems:

- Need of storing all stats of n-gram models in the corpus.

How to build a Neural Language Model?

- Recap: Language modelling is predicting next word given previous N words.
- Idea of Neural language model:
  - Input: sequence of N words
  - Output: Probabilities of next word over whole dataset
- Pros and Cons:

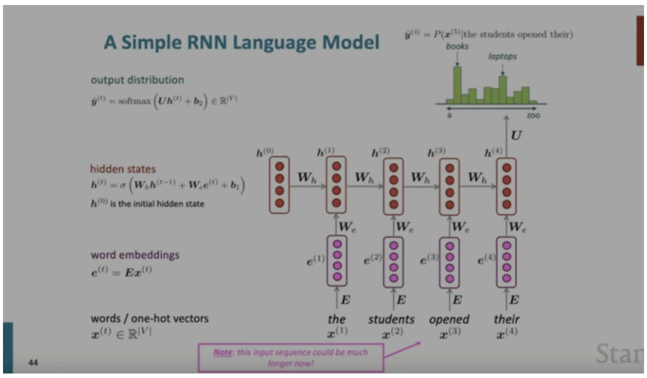
Pros	Cons
No sparsity problems. As system is not count based but considers whole dataset To calculate probabilities	Context is still limited to window of N words
Storage problem is solved as no need to store stats	Can't increase Window as it increase W matrix – computation time increases
	Sequence of word is not strictly considered as x1 and x2 gets multiplied by w1 and w2. so features of x1 and x2 might end of far away from each other



Why RNNs?

- Need or processing variable/larger input lengths
- Need to strictly consider the sequence of words

Simple RNN architecture:



What problems are solved by using RNNs?

- Input length is decoupled from model size, so can use any input length
- Information of N previous steps is available in current hidden states – helps to provide large context for predicting next word
- Each input word vector gets multiplied by same Weight matrix W_e – order of words is preserved