Unit 3 – lecture 11: Recurrent Neural Network 2

Objectives:

- Formulate, estimate and sample sequence from Markov models.
- Understand the relation between RNNs and Markov model for generating sequences
- Understand the process of decoding of RNN in generating sequences.

Markov Models:

The kth order Markov model is a simple probability model of giving a distribution of what comes next, based on looking at k steps backwards.

Let $w \in V$ denote the set of possible words/symbols that includes:

- an UNK symbol for any unknown word (out of vocabulary)
- <beg> symbol for specifying the start of a sentence
- <end> symbol for specifying the end of the sentence

<beg> The lecture leaves me UNK <end>

w_0 w_1 w_2 w_3 w_4 w_5 w_6

In a first order Markov model (bigram model) the next symbol only depends on the previous one.

$$P(w_1, \cdots, w_6) = P(w_1 | \underbrace{w_0}_{<beg>}) \cdots P(\underbrace{w_6}_{<end>} | w_5)$$

	ML	course	is	UNK	<end>	
<beg>	0.7	(0.1)	0.1	0.1	0.0	$\rightarrow \Sigma = 1$
ML	0.1	0.5	0.2	0.1	0.1	
course	0.0	0.0	(0.7)	0.1	0.2	
is	0.1	0.3	0.0	(0.6)	0.0	
UNK	0.1	0.2	0.2	0.3	(0.2)	

<beg> Course is UNK great <end>

$P(\text{course} | <beg>)$ $P(\text{is} | \text{course})$ $P(\text{UNK} | \text{is})$ $P(<end> | \text{UNK})$

0.1 0.7 0.6 0.2

<bos> course is UNK <end>

we must look at the row here, probability of distribution over the first word which comes after the beginning symbol.

To sample from it, we throw a weighted 5 way die, weighted heavily towards ML, a little bit less for course and is, and the end symbol has probability 0. Most likely we would get the word ML, but we might also get the word course

- The goal is to maximize the probability that the model can generate all the observed sentences (corpus S)

$$s \in S, \quad s = \{w_1^s, w_2^s, \ldots, w_{|s|}^s\}$$

$$\prod_{s \in S} \left[\prod_{i=1}^{(s)} P(w_i^s \mid w_{i-1}^s) \right]$$

$\prod_{s \in S}$: for all the sentences

$\prod_{i=1}^{(s)} P(w_i^s \mid w_{i-1}^s)$: 1 sentence.

Maximum likelihood estimation.

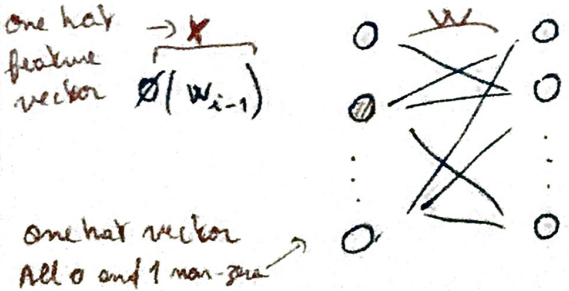
- The ML estimate is obtained as normalized counts of successive word occurences (matching statistics)

$$\text{count}(w, w')$$

$$\hat{P}(w' \mid w) = \frac{\text{count}(w, w')}{\sum_{\tilde{w}} \text{count}(w, \tilde{w})}$$

Markov Models to Feed forward Neural Nets:

We can also represent the Markov model as a feed forward neural network (very extendable)



$P_k = P(w_i = k \mid w_{i-1})$ $\quad P_k \geq 0$ and $\sum_k P_k = 1$

↑ proba of next word

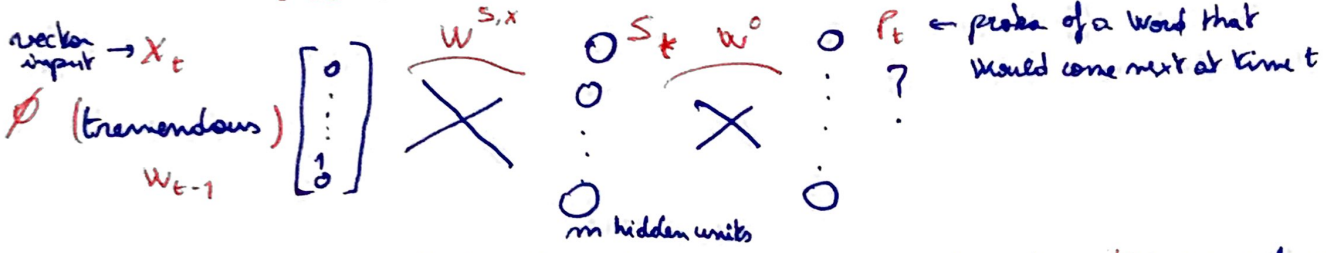
$z_k = \sum_j x_j w_{jk} + w_{0k} \in \mathbb{R}$

Softmax output layer: $P_k = \frac{e^{z_k}}{\sum_j e^{z_j}} \Rightarrow P_k \geq 0, \sum_k P_k = 1$

RNN Deeper Dive:

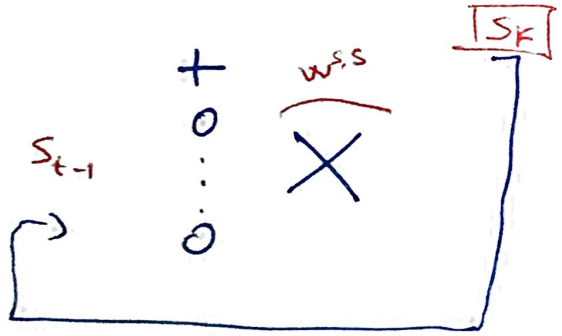
Language modeling: What comes next?

This course has been a tremendous | . . .



We feed in to the model a one hot vector of the previous word, where t now refers to the time or position in the sentence when we are making a prediction.

We make this into a RNN model by feeding in the previous hidden state activation



$s_t = \tanh(W^{s,s} s_{t-1} + W^{s,x} x_t)$ state

$P_t = \text{softmax}(W^o s_t)$ output distribution

Decoding: