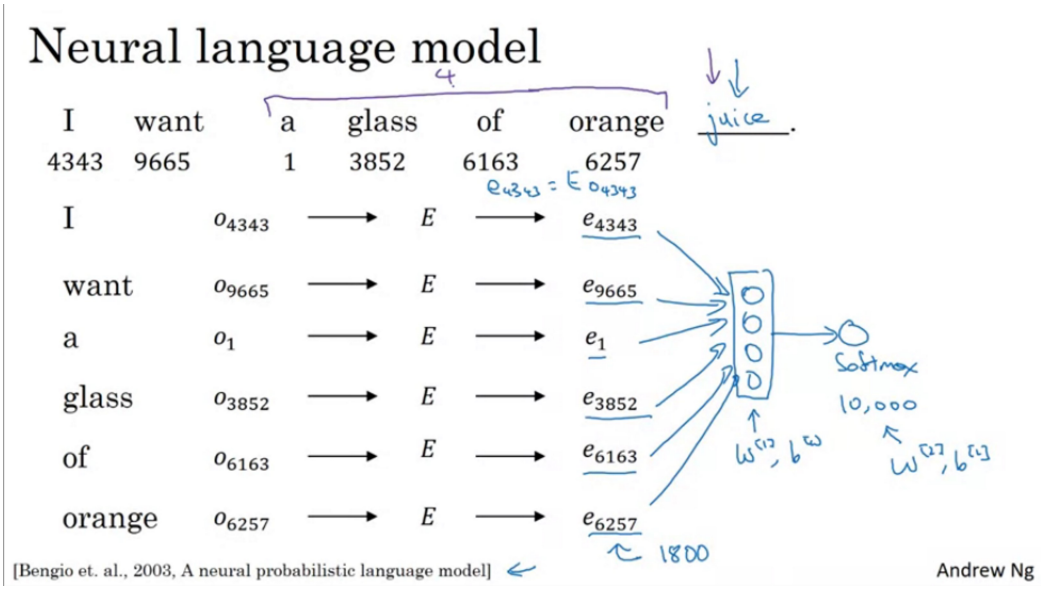
Learning Word Embeddings: Word2vec & GloVe

Wednesday, September 30, 2020 3:11 PM

① Learning word embeddings

- The newer models are simpler than initial models
- We will learn the initial models first to gain intuition
- Neural language model



→ $e_j = E \cdot o_j$

→ The embeddings are passed to NN layer followed by softmax layer

→ Input size is $(300*6, 1)$ if all 6 previous words are used (window size = 6)

- Other context / target pairs

eg: I want a glass of orange juice to go along with my cereal

Context: Last 4 words → a glass of orange ___
4 words both sides → a glass of orange ___ to go along with

Last 1 word → orange ___
Nearby 1 word → glass ___
↳ skip gram model (works remarkably well)

Nearby 1 word → glass ___
↳ skip gram model (works remarkably well for learning embeddings)

② Word2Vec

- Skip grams
  - → We choose context and randomly a target within a specific window (eg: +10/-10 words)

context	target
Orange	juice (next word)
orange	glass (2 words back)
orange	my (6 words after)

  - → Now this is used as a supervised learning problem

- Word2vec model
  - → Vocabulary size = 10000 words
  - → we need to learn context (c) to target(t)
  - → $e_c = E \cdot O_c$
  - → soft max is used to get $P(t|c) = \hat{y}$

$$O_c \rightarrow E \rightarrow e_c \rightarrow \underbrace{O \longrightarrow \hat{y}}_{\text{softmax}}$$

Softmax: $P(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10000} e^{\theta_j^T e_c}}$ θ_t = parameter associated with t

  - → Cross entropy loss is used

$$\mathcal{L}(\hat{y}, y) = -\sum_{i=1}^{10000} y_i \log \hat{y}_i \qquad y = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \rightarrow \text{one hot vector}$$

- Problems with softmax classifier
  - → The soft max layer

$$P(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{i=1}^{10000} e^{\theta_j^T e_c}}$$

is very computationally intensive as we are summing through our entire vocabulary

  - → This can be solved by using a hierarchical softmax classifier which uses a tree like structure with more common words at the top and rare words in the deeper nodes

- To sample context c
  - → Can be chosen randomly but frequent words like a, the, my :.. will dominate
  - → In reality, it is chosen randomly but not uniformly to balance the common words and non-common words
- 2 versions of Word2vec are used skip grams & CBoW (continuous bag of words)

③ Negative Sampling

- Negative Sampling is similar to skip gram model but much more efficient algorithm
  - For the example
    I want a glass of orange juice to go along with my cereal

    The sampling would be

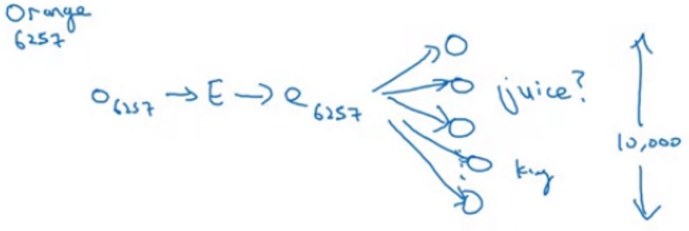
context	word	target	
orange	juice	1	→ positive example
orange	king	0	} k negative contexts
orange	book	0	
orange	the	0	
orange	of	0	

↳ it is fine if same negative samples appearin same sentence

→ Positive example is chosen by skip-gram technique
→ Negative examples (K → 5 to 20 acc. to original paper) are picked randomly from vocabulary

- The model is defined as
  - → target is the probability that a pair of words are a context target pair
  - → we use logistic regression model

$$P(y=1 \mid c, t) = \sigma(\theta_t^T e_c)$$



10,000 binary classification problems
↳ we train k+1 classifiers each iteration

- To select negative examples
  - → We can sample according to word frequencies but results in frequent words like the, of, and ...
  - → Hence, we use (acc. to original paper)

$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=1}^{1000} f(w_j)^{3/4}} \qquad f \to \text{freq.}$$

④ GloVe word vectors

- Used less frequently, known for simplicity stands for Global vectors
- we will choose context and target similar to previous models.
- we define

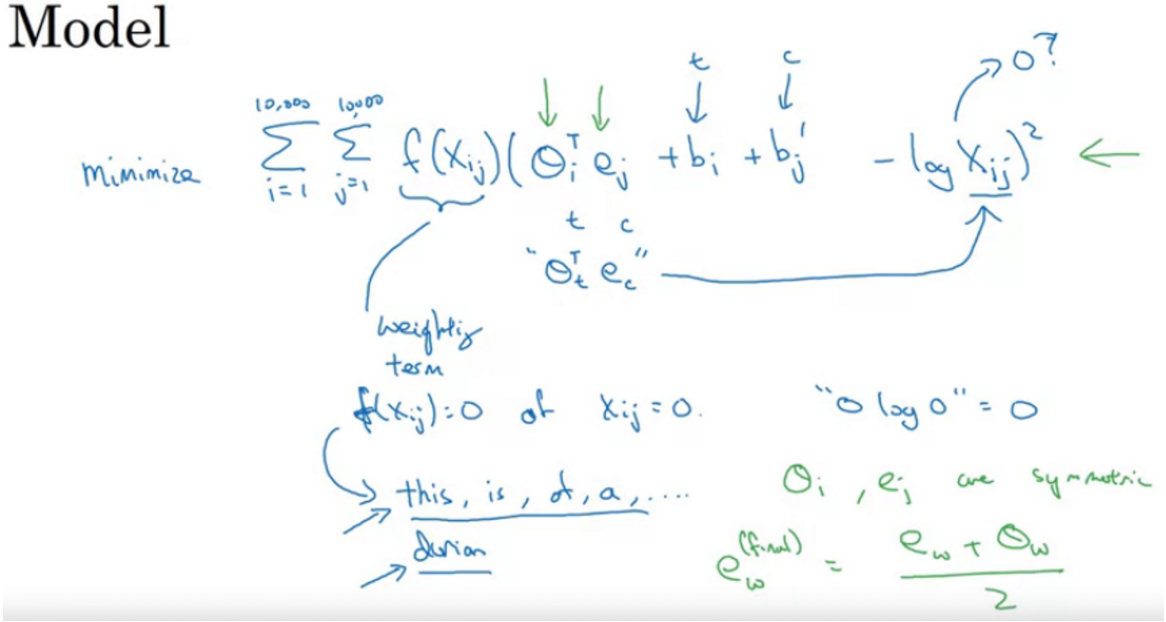
$$X_{ct} = \text{No. of time } t \text{ appears in context of } c$$

- $X_{ct} = X_{tc}$ is also possible if we choose a window pair
- The model is defined as

$$\text{minimize} \to \sum_{i=1}^{10000} \sum_{j=1}^{10000} f(X_{ij}) \left(\theta_i^T e_j + b_i + b_j - \log X_{ij}\right)^2$$

  - → $f(X_{ij})$ is weighting term used to avoid calculations if $X_{ij} = 0$, hence for solving the log0 problem and to not give too much weight to frequent words and not too little to infrequent words
  - → θ & e are symmetrical and swappable unlike previous models

Model


Minimize
10,000
10,000
$\sum_{i=1}^{10,000} \sum_{j=1}^{10,000} f(X_{ij})(\theta_i^T e_j + b_i + b_j' - \log X_{ij})^2$
$\theta_t^T e_c$
weighting term
$f(x_{ij}) = 0$ if $x_{ij} = 0$.
"$0 \log 0$" = 0
this, is, if, a, ...
θ_i, e_j are symmetric
$e_w^{(final)} = \frac{e_w + \theta_w}{2}$