deeplearning.ai

NLP and Word Embeddings

Word2Vec

Skip-grams

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

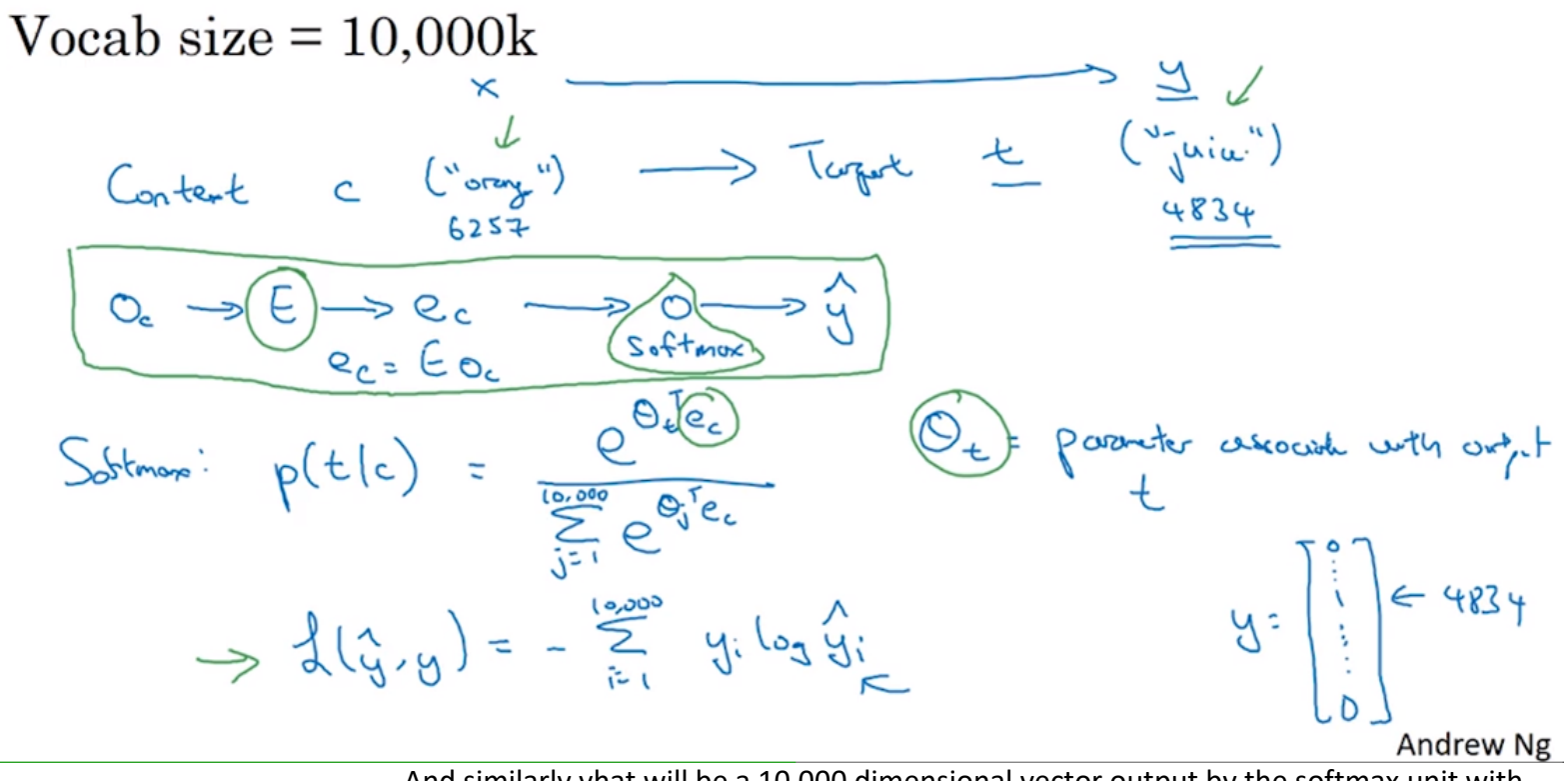
Context	Target
orange	juice
orange	glass
orange	my

[Mikolov et. al., 2013. Efficient estimation of word representations in vector space.]

Andrew Ng

But the goal of setting up this supervised learning problem, isn't to do well on the supervised learning problem per se,is that we want to use this learning problem to learn a good word embeddings.

Model



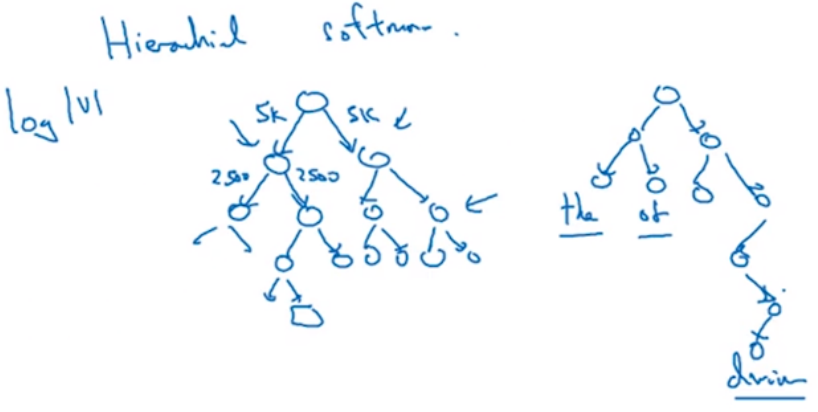
Andrew Ng

And similarly yhat will be a 10,000 dimensional vector output by the softmax unit with probabilities for all 10,000 possible targets words.

Problems with softmax classification

网易云课堂

$$p(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10,000} e^{\theta_j^T e_c}}$$



Andrew Ng

In the next video, we actually talk about a different method called negative sampling, which I think is even simpler. And also works well on speeding up the softmax classifier and the problem of needing the sum over the entire cab size in the denominator.

In practice, the distribution of words pc isn't taken just entirely uniformly at random from the training set corpus, but instead there are different heuristics that you can use in order to balance out sampling from the common words to get the less common words.

So having a tree of classifiers like this, means that each of the interior nodes of the tree can be just a binary classifier like a logistic classifier, and so you do not need to sum over all 10,000 words all the vocab size in order to make a single classification. In facet the computational cost using a tree like this scales like log of the vocab size rather than a linear vocab size.

So this is called a hierarchical softmax classifier.

I should mention in practice, the hierarchical softmax classifier doesn't use a perfectly balanced tree or this perfectly symmetric tree wit h equal numbers of words on the left and right sides of each branch.
In practice, the hierarchical softmax classifier can be developed so that the common words tend to be on top, whereas the less common words like durian can be buried much deeper in the tree. Because you will see the more common words more often and so you might need only a few traversals to get to common words like the "and"\"of". Whereas you see less frequent words like durian much less often, so it's okay that are buried deep in the tree, because you don't need to go that deep as frequently. So there are various heuristics for building the tree that you used to build the hierarchical softmax classifer.

If you read the original paper, you actually find two version of this Word2Vec model, the skip gram was one.
And the other one is called the cBow, the Continuous Bag of Words model, which take the surrounding contexts from the middle word, and uses the surrounding words to try to predict the middle word and that algorithm also works, it has some advantages and disadvantages.

But the key problem with this algorithm with the skip-gram model as presented so far is that the softmax step is very expensive to calculate because needing to sum over your entire vocabulary size into the denominator of the softmax.

In the next video, I will show you an algorithm that modifies the training objective that makes it run much more efficiently and therefore lets apply this maybe to much bigger training sets as well and therefore learn much better word beddings.