**Paper Link**

[Distributed Representations of Words and Phrases and their Compositionality](https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf)

**Video Summary**

[12.1: What is word2vec? - Programming with Text](https://www.youtube.com/watch?v=LSS_bos_TPI)

Word2Vec is a ML model which produces an embedding, associating words with numbers

Word2Vec quantifies words

Recall that word embeddings can be done in an unsupervised manner

**Pre-requisite papers (Roots)**

Used for embedding inputs which all NLP models need

Skip-gram architecture - efficient method of learning embeddings from unstructured text data, does not involve dense matrix multiplications

[Noice contrastive estimation](https://docs.google.com/document/d/1nL6tMot5scWWQeVQfJLZjz9VNna6G0nOZ3kcTTQTBLM/edit)

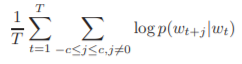
**Notes**

NCE variant is presented for training which speeds up training and gives better vector representations for more frequent words

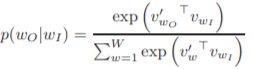
Phrase based models are used instead of word-based, due to idiomatic word meanings

Simple vector addition can produce meaningful results (e.g "germany" + "capital" = "Berlin")

Given a sequence of words, skip gram attempts to to maximize the average log probability

where c is the size of the training context, with a larger c resulting in more examples increasing the accuracy at the expense of training time

The normal skip gram formulation defines the probability distribution using softmax



The input representation and the output embedding generated by the model are both used and must provide a value such that p(wO | wi) is maximized

Remember, we are generating word embeddings here, not the output vocabulary

W is the # of words in the (input) vocabulary, vw and vw' are the input and output vector representations of w

By input and output representations, we mean that vwi is the word embedding for wi and vwo' is the embedding for the output word wo = wi+j where j is in the context

This is not used in practice because the cost of computing the gradient of the log probability is proportional to the vocabulary size W

Therefore, instead we use the hierarchical softmax which only needs to evaluate log2(W) output nodes to obtain the probability distribution

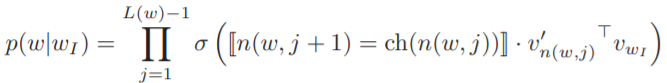
Hierarchical softmax uses binary tree representation of the output layer with W words as its leaves, each node representing the relative probabilities of its child nodes

Each word w can be reached from the root

Let n(w, j) be the j-th node on the path from the root to w, and let L(w) be the length of this path

n(w, 1) = w

Let ch(n) be an arbitrary fixed child of n and let [[x]] be 1 if true and -1 otherwise



The cost of computing the gradient of log p is proportional to L(wO) which is no greater than log W

The hierarchical softmax, unlike the standard softmax formulation, has 1 representation vw for each word w, and one representation vn' for each inner node n of the binary tree

Grouping together words in the tree structure by their frequency in the training data speeds things up

NCE says that a good model should be able to differentiate data from noise through logistic regression

We can simplify NCE so that the word embeddings are still high quality (while using less computation), and we therefore define negative sampling (NEG) with the objective:



This replaces every log P(Wo | wi) term in the Skip-gram objective

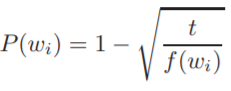
We try and distinguish wO from the noise distrbiution Pn(w) using logistic regression, with k (arbitrary) amount of negative samples

Negative sampling only uses samples, while NCE needs numerical probabilities as well (which is not required for word embeddings)

The unigram distribution raised to the power of ¾ is optimal for the noise distribution

The vector representations of frequent words such as "the" do not change significantly after millions of examples since they co-occur with many words

To counter the imbalance between rare and frequent words, each word wi in the training set is discarded with the probability



where f(wi) is the frequency and t is an arbitrary threshold

The formula aggressively subsamples words with a frequency greater than t, and improves accuracy of the learned embeddings of rare words

**Branches**