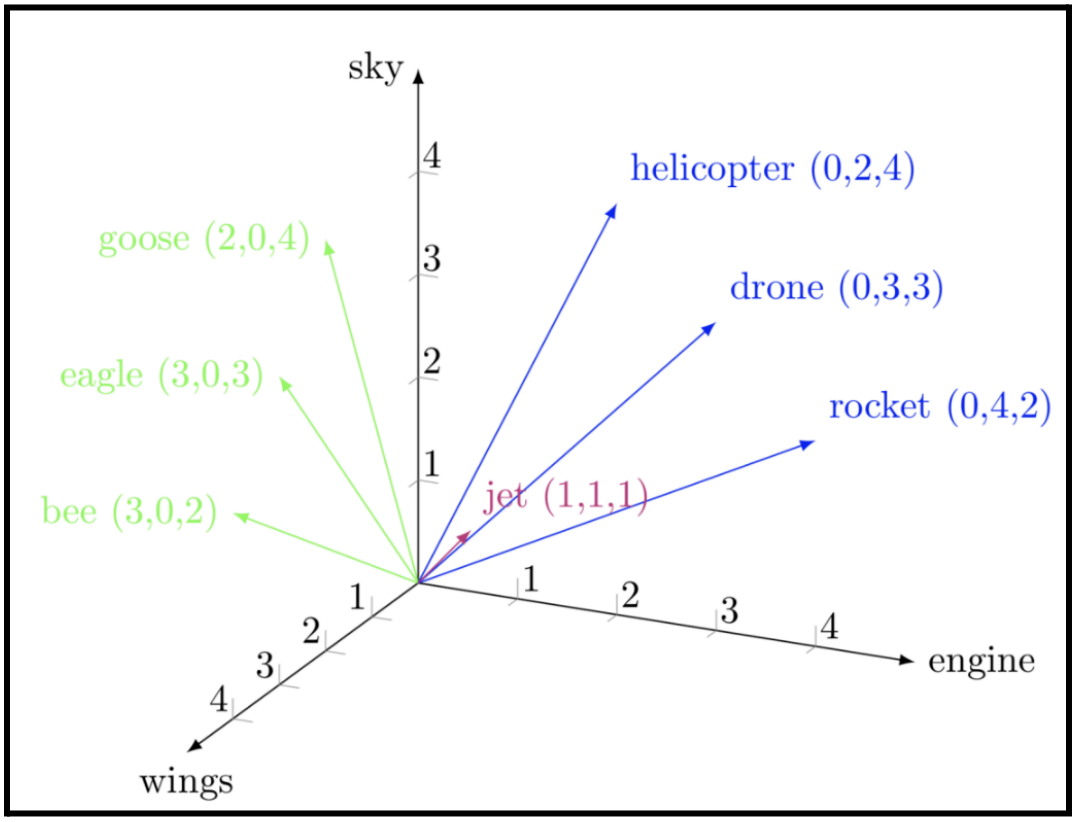
Lecture 3: Word Embeddings

Word Embeddings

A word embedding is a learned representation for text where words that have the same meaning have a similar representation

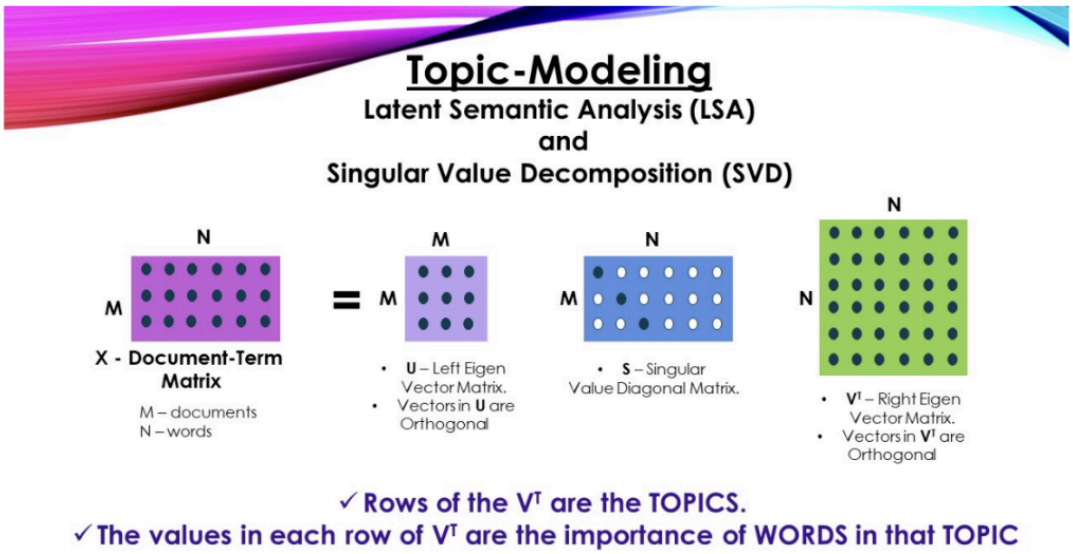


Advantages over Discrete representations

- Able to capture relations between different words.
- Ability to capture context by syntactic and semantic similarity

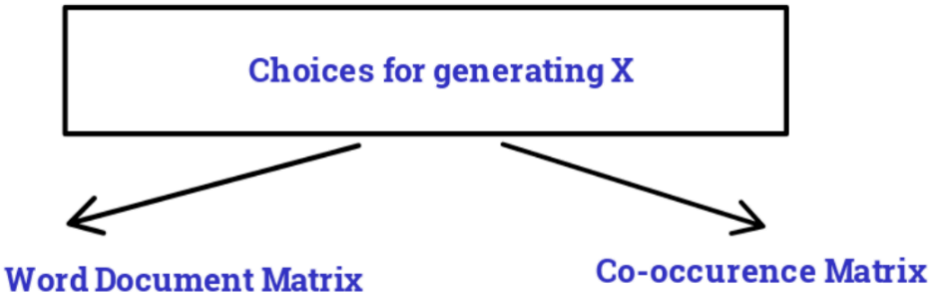
Single Value Decomposition (SVD)

Let A be an m*n matrix. Then there exists a factorisation of A,
A =UΣV
where U is an mm orthogonal matrix, V is an nn orthogonal matrix and Σ is an m*n matrix of the form,



Steps:

1. Looping over the corpus to accumulate word co-occurrence counts in the X matrix.
2. Perform SVD on X to get decomposition.
3. Use rows of U as the word embeddings for all words in our dictionary.



- Assumption (for both): Words that are related will often appear in the same documents and vice versa
- In 2nd choice, X contains word co-occurrences (i.e. affinity matrix)

Advantages

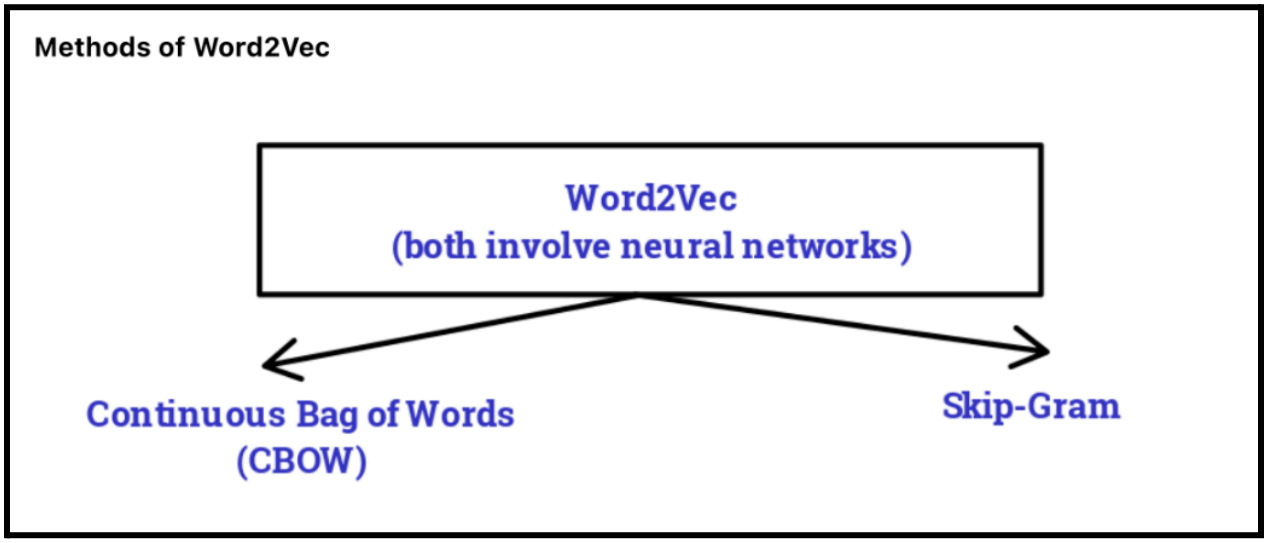
- preserves the semantic relationship
- efficient factorization
- computed only once, can be used multiple times.

Disadvantages

- poor scaling on large matrices (large memory).

Word2Vec

The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text.

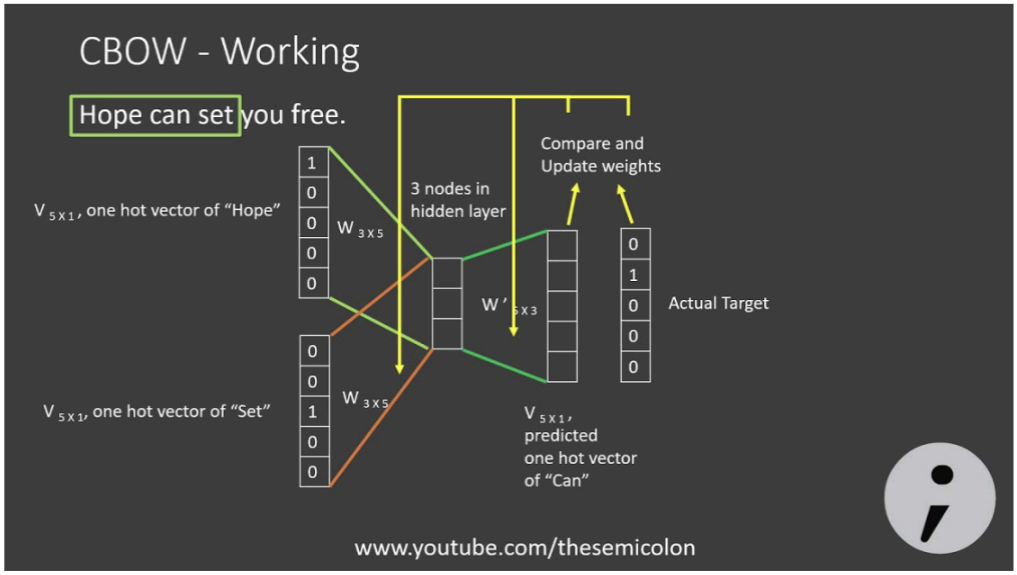


- **Continuous Bag of Words (CBOW)**: predicts the current word from a window of surrounding context ****words.
- **Skip Gram:** the model uses current word to predict the surrounding window of context words.

CBOW Architecture

Input	Hidden Layer	Output
context words in a vector with size of vocabulary.	hyper-parameter which defines shape of word representations	outputs probability of all the words in a vector of size of the vocabulary.

ReLU activation: Input to Hidden Layer
Softmax activation: Hidden to Output Layer



CBOW learns the word representations by reducing the **Cross-Entropy Loss (objective function)** during back-propagation

$$J = -\Sigma_{k=1}^{V} y_k \log \hat{y}_k$$

Disadvantages of CBOW:

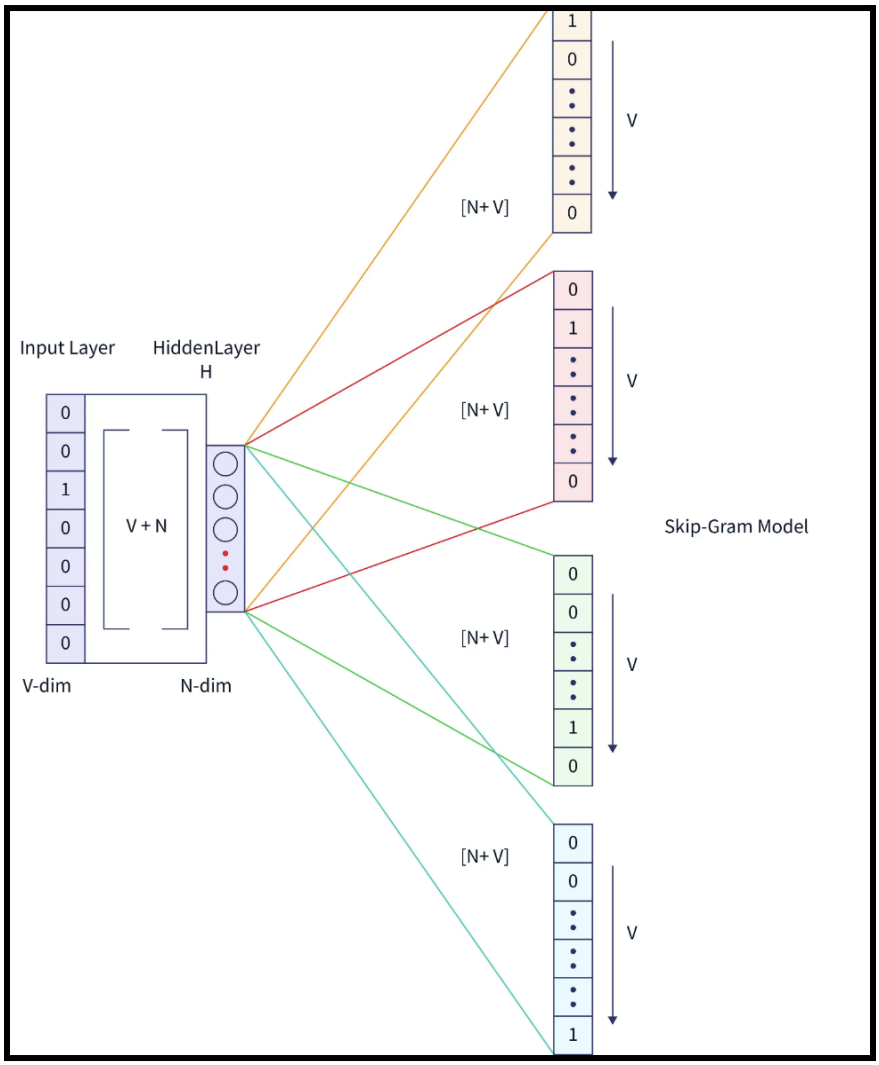
- Overfits on frequent words for context window > 1
- Doesn't produce good representations of rare words.

Skip-Gram Method

Takes in input as center/context word and predicts its surrounding words.

Source Text	Training Samples generated from source text			
I will have orange juice and eggs for breakfast	(will, I)	(will, have)	(will, orange)	
I will have orange juice and eggs for breakfast	(have, I)	(have, will)	(have, orange)	(have, juice)
I will have orange juice and eggs for breakfast	(orange, will)	(orange, have)	(orange, juice)	(orange, and)
I will have orange juice and eggs for breakfast	(juice, have)	(juice, orange)	(juice, and)	(juice, eggs)
I will have orange juice and eggs for breakfast	(and, orange)	(and, juice)	(and, eggs)	(and, for)
I will have orange juice and eggs for breakfast	(eggs, juice)	(eggs, and)	(eggs, for)	(eggs, breakfast)
I will have orange juice and eggs for breakfast	(for, and)	(for, eggs)	(for, breakfast)	

Skip-Gram Architecture



Skip-gram learns word representations via:

1. matching output and true probabilities
2. **Log Likelihood** objective function.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$