Bag-of-Words Model

Legacy Techniques: counting is everything

One-hot Vector

- Map each word to an unique ID
- ID can be the index of the word in the whole vocabulary.

The cat and the dog play
The cat is on the mat

corpus

and, the, cat, dog, play, on, mat, is

vocab.

and	0
the	1
cat	2
dog	3
play	4
on	5
mat	6
is	7

One-hot Vector

- The ID can determine the one-hot word vector

- A vector filled with 0s, except for a 1 at the position of the ID

cat	*mat*
0	0
0	0
1	0
0	0
0	0
0	0
0	1
0	0

One-hot Vector

- Pros
  - Simple
  - Easily computed and suitable for parallel computing

- Cons
  - Dimensionality is the size of vocabulary
  - Out-of-Vocabulary (OOV) problem
  - All words are independent

Bag-of-Words

- Steps
  - Build vocab i.e., set of all the words in the corpus
  - Count the occurrence of words in each document

The cat and the dog play
The cat is on the mat

corpus

and, the, cat, dog,
play, on, mat, is

vocab.

1	2	1	1	1	0	0
1	2	0	0	1	1	1

Bag-of-Words

- Pros
    - Simple
    - Surprisingly effective
    - Fast

- Cons
    - Order of words does not matter
    - Cannot capture syntactic/semantic information
    - High dimensionality

N-gram model

- Steps
  - Build vocab, which set of all n-gram in the corpus
  - Count the occurrence of n-gram in each document

The cat and the dog play
The cat is on the mat

corpus

The cat, cat and, and the, the dog, dog play, cat is, is on, on the, the mat

vocab.

1	1	1	1	1	0	0	0	0
1	0	0	0	0	1	1	1	1

N-gram model

- Pros
  - Word order is considered

- Cons
  - Vocab size is very huge
  - Can not capture syntactic/semantic information
  - Is able to incorporate limited word order information

Term Frequency-Inverse Document Frequency

- Build vocab i.e., set of all the words in the corpus
- Count the occurrence of words in each document
- Use weighting scheme to determine the value
  - TF(w) = number of times term w appears in a document/Total number of terms in the document
  - IDF(w) = log(total number of documents / number of documents with the term w in it)
  - The final weight is TF(w) * IDF(w)
- Intuitive logic:
  - Capture the importances of a word to document in a corpus
  - Importance of words is proportionally to the number of times a word appears
  - Importance of words is inversely proportionally to the document containing the word

How to Learn More Semantic Representation

Reduce high dimensionality

- Latent Semantic Analysis
- Topic Models

Word Embeddings

- Definition
  - A continuous vector representation of word
  - Syntax and semantics information may be encoded into embeddings space.

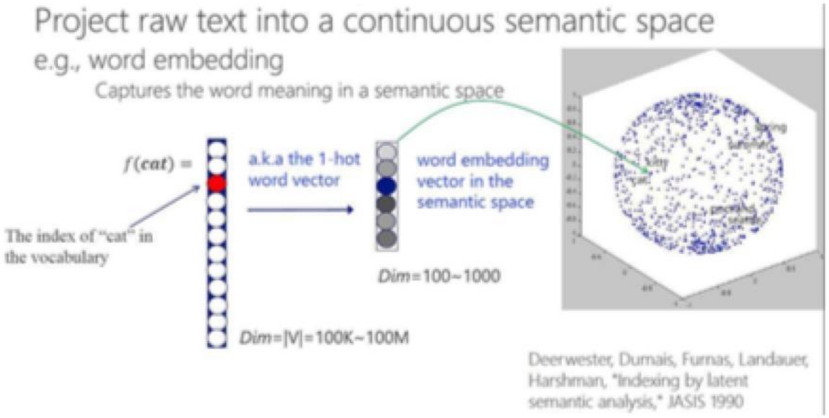
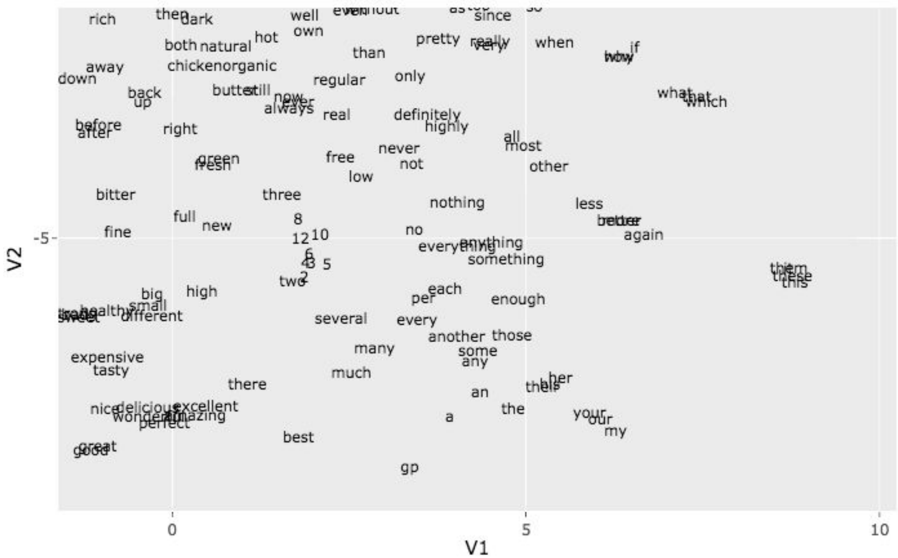


Figure: This is a graphic from (He. et al, 2014)

Word Embeddings

- Advantages
  - Low-dimensional and dense word vectors make the application of neural network on NLP possible
  - Word vectors will be related: similar words may be close to each other in the vector space.



Embeddings vs one-hot vector

Dog	0	1	0	0	0
			⊗		
Puppy	0	0	0	1	0
			=		
	0	0	0	0	0

element-wise product is zero vector.
Thus, Inner product is zero.

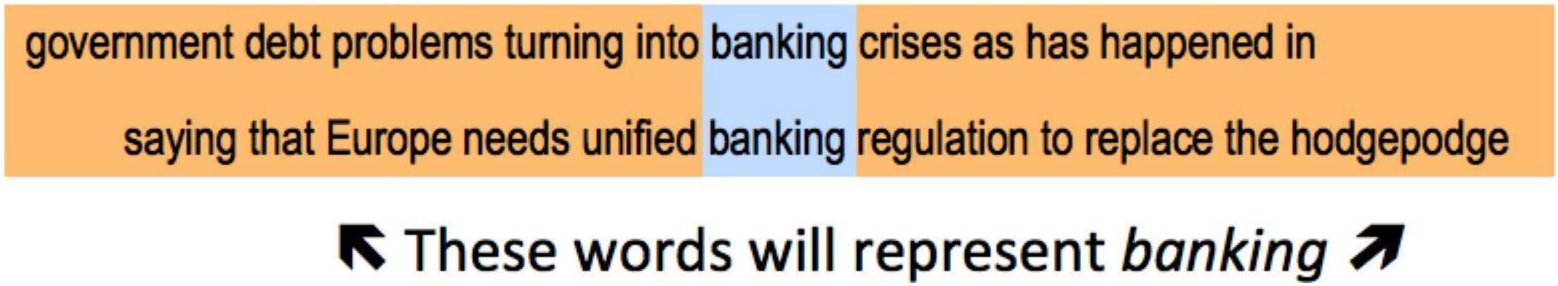
50～300 *dim*

Dog	0.52	0.21	0.37	…
Puppy	0.48	0.21	0.33	…
Computer	0.05	0.23	0.06	…

Distributional Semantics

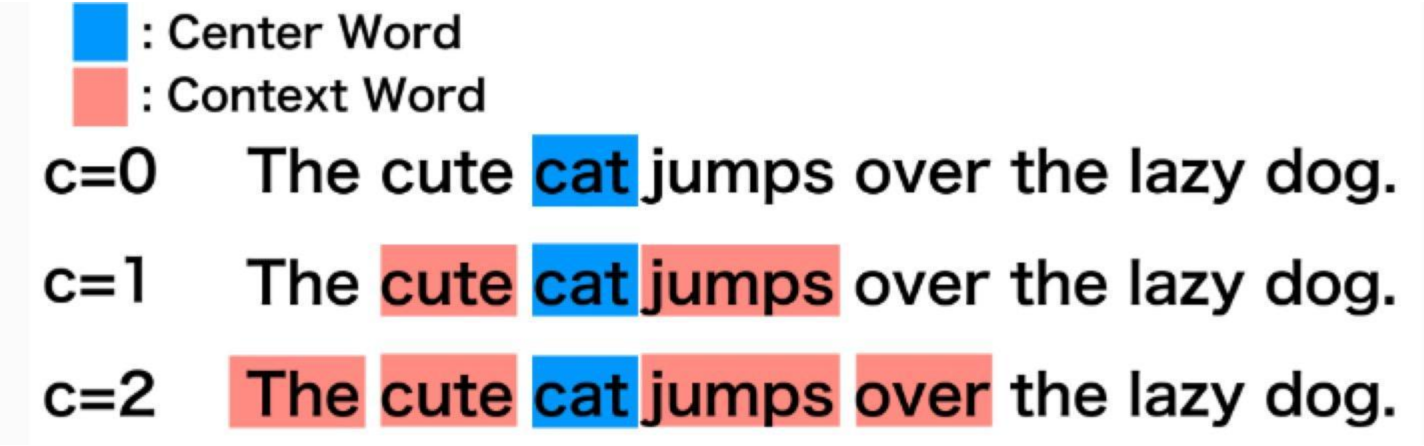
- The Rationale behind word embeddings is that you shall know a word by the company it keeps.

- Words are similar if they appear in similar context



Context vs Center Words

- Create the pair between center and context words.
- Hyper-parameter: window size c



Windows based Vectors

- Win Lens: commonly 5-10
- Symmetric
- Construct the concurrence matrix from the corpus
- Each row can be word embeddings

Example corpus:

- I like deep learning.
- I like NLP.
- I enjoy flying.

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Still not perfect

- High dimensionality: vocab size
- Still sparse
- Need dimensionality reduction