IA Frameworks
Introduction to Natural Language Processing (NLP)

What is NLP?

What is a language

A **language** is a structured system of communication. The structure of a language is its **grammar** and the free components are its **vocabulary**. Languages are the primary means of communication of humans, and can be conveyed through **speech** (spoken language), **sign**, or **writing**. (Wikipedia)

Natural Language Processing

Objectives:

- Design programs able to understand human language as it is spoken and written.
- Extract insightful information
- Produce controlled text or speech

Examples:

Sentiment analysis, document classification, translation, question answering, summerization, …

Textual data

- A **corpus** is a collection of:
  - **documents** which are sequences of :
    - **tokens**

Token: basic unit of discrete data indexed from a vocabulary

- Word
- Sub-word
- A sequence of words or sub-words
- A character
- A symbol

How to identify tokens?

Tokenization

Tokenization

Consists in segmenting a document into tokens

Not so trivial!

This is an example of tokenization.

Use whitespace?

[“This”, “is”, “an”, “example”, “of”, “tokenisation.”]

Tokenization

Consists in segmenting a document into tokens

Not so trivial!

This is an example of tokenization.

Use whitespace and punctuation?

[“This”, “is”, “an”, “example”, “of”, “tokenisation”, “.”]

That’s a problem…

[“That”, “‘“, “s”, “a”, “problem”, “.”, “.”, “.”]

Tokenization

Many other problems:

- compound words (e.g. pick-pocket, German, …)
- No separators (Chinese, Japanese)
- …

Many partial solutions:

- Character level tokenization
- Regular Expression tokenization
- Dictionary based tokenization
- Rule Based Tokenization (Penn TreeBank, Spacy, Moses, …)

Lemming

Lemmatization is the process of grouping inflected forms together as a single base form

Example:

"builds", "building", or "built" => "build"

Stemming

Stemming is the process of reducing inflected words to their word stem, base or root form

Example:

“programming”, “programs”, “programmed” => “program”

Subwords tokenization

Principle:

- Frequently used words are unit tokens
- Less frequent words should be decomposed into meaningful subwords
  e.g. “annoyingly” => "annoying" and "ly"
- Rely on model training to discover the most frequent occurring pairs of symbols

Advantages:

- reasonable vocabulary size
- meaningful context-independent representations
- process unknown words

Text Cleaning

- Remove noise (HTML, specific symbols like #, nouns, references, …)
- Remove stop words (and, or, the, …)
- Pass a spell checker on your data
- Convert to lowercase
- Look for synonyms?

Vectorization

One hot encoding

(1) I am going to the supermarket
(2) The post office is close to the supermarket

I	am	going	to	the	supermarket	post	office	is	close
1	1	1	1	1	1				
			1	1		1	1	1	1

Count encoding

(1) I am going to the supermarket
(2) The post office is close to the supermarket

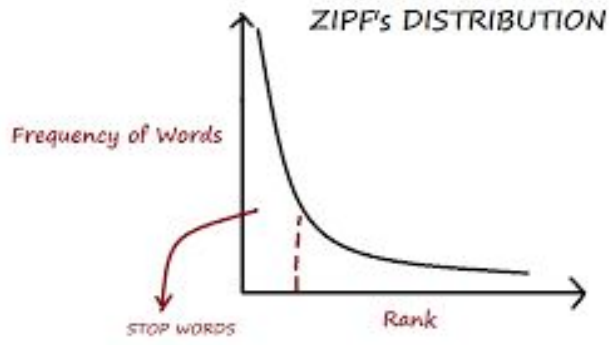
I	am	going	to	the	supermarket	post	office	is	close
1	1	1	1	1	1				
			1	2		1	1	1	1

Term-frequency matrix: $tf_{t,d} = |\{t \in d\}|$

TF-IDF

Count based encoding is sensitive to frequent words

- A word present in all documents is not very informative
- A word present in few documents is very informative



Weight term frequency with the **Inverse Document Frequency**: $idf_{t,C} = \log\left(\frac{|C|}{|\{d \in C, s.t. t \in d\}|}\right)$

TF-IDF

(1) This is a big supermarket
(2) This post office is close to a supermarket

This	is	a	to	supermarket	post	office	big	close
0	0	0		0			0.3	
0	0		0.3		0.3	0.3		0.3

TF-iDF: $$tf_{t,d}.idf_{t,C} = |\{t \in d\}| \log \left(\frac{|C|}{|\{d \in C, s.t. t \in d\}|} \right)$$

Bag of words

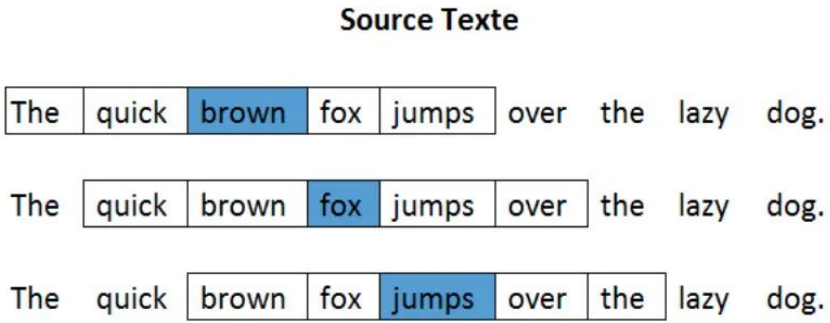
- Does not scale with vocabulary size
- Very sparse representation
- No semantic information (synonyms are treated like different words)

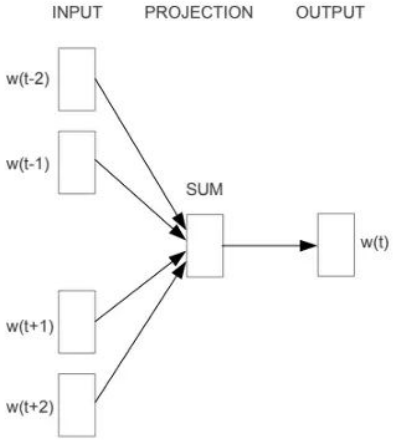
Word2Vec

- Words are mapped to embeddings (e.g. computer -> [0.3, 8, 6.5, …, 4.2])
- Embeddings are built through shallow neural networks trained to reconstruct linguistic contexts of words
- Word with similar semantic are located close to one another in the latent space:

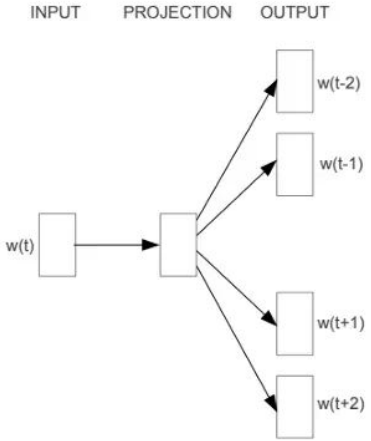
$$sim(motocycle, car) > sim(motocycle, chicken)$$

Word2Vec



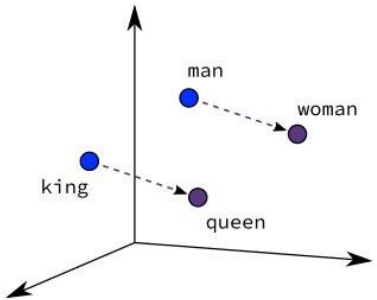


CBOW

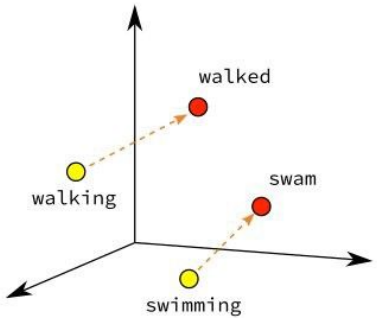


Skip-gram

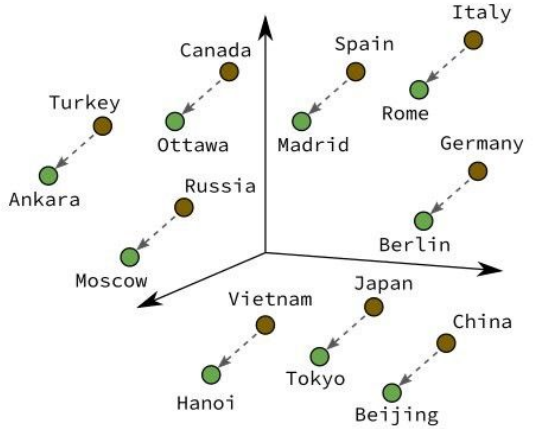
Word2Vec



Male-Female



Verb Tense



Country-Capital

Other features

- Number of words, characters, …
- Grammatical categories
- Number of capital letters