Applied Natural Language Processing

Words to Vectors

Ramaseshan Ramachandran

RECAP

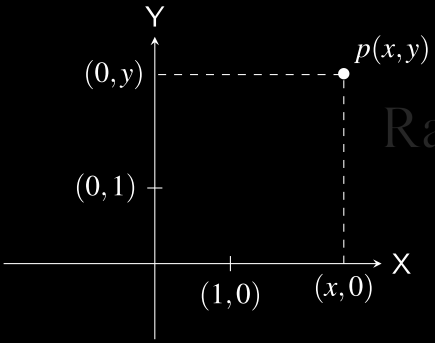
- Why NLP is hard
- Data driven approach to find information from Corpus
  - Term frequency, IFT and document ranking
  - Prediction of vocabulary (Heap's Law), frequency distribution (Zipf's Law)
  - Measure of Lexical density using Type-Token Ratio
- Binary Incidence Matrix representation of a corpus and unranked boolean retrieval
- Ranked document retrieval using TF-IDF

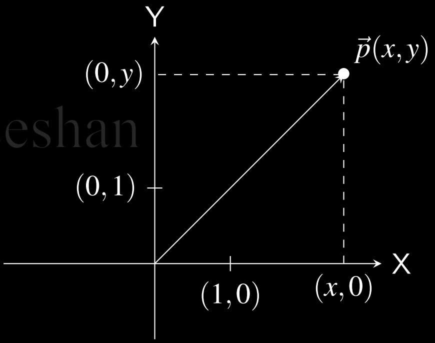
TOPICS TO BE COVERED IN THIS CLASS

1. Word2Vector
   - Recap
   - Topics to be covered
   - 2-D Vector Space
   - 3-D Vector Space
2. Vector Space Model for Words and Documents
   - VSM for Words
   - Document Vector Space Model
   - Document-Term Matrix
   - Query Modeling
   - Document Similarity
   - Demo - Cosine Similarity
   - Word Vector
   - One-Hot Vector
   - One-Hot- Vector - example
   - Relationship among terms
   - Is-A Vector
   - Information Extraction
3. Co-occurrences
   - Contextual Understanding of Text
   - Co-occurrence Matrix
   - Unigram, Bigrams and Trigrams
   - N-grams
   - Collocations
4. Semantically connected Word Vectors
   - Dense Vectors
   - Firth
   - Singular Value Decomposition
5. Latent Semantic Indexing
   - Why SVD for LSI?
   - LSI
   - Queries in the LSI
   - Applications of LSI
   - LSI/LSA Topic Modeling-2
   - LSI/LSA
   - Word Embedding

2-D VECTOR SPACE

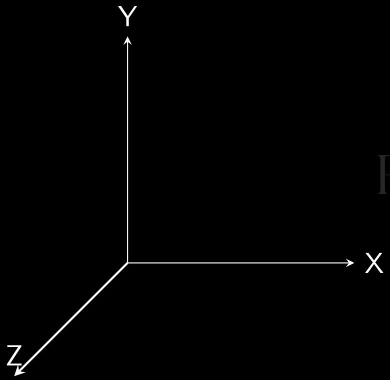
A 2-D vector-space is defined as a set of linearly independent basis vectors with 2 axes. Each axis corresponds to a dimension in the vector-space

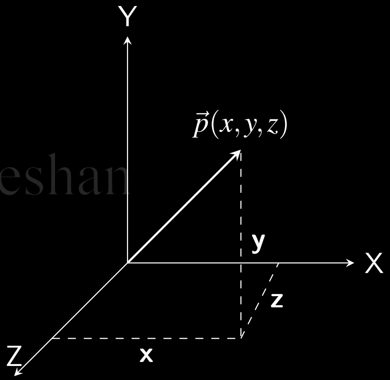




3-D VECTOR SPACE

A 3-D vector-space is defined as a set of linearly independent basis vectors with 3 axes. Each axis corresponds to a dimension in the vector-space





Linearly independent vectors of size $\mathcal{N}$ will result in $\mathcal{N}$-dimensional axes which are mutually orthogonal to each other

VECTOR SPACE MODEL FOR WORDS

Let us assume that the words in a corpus are considered as linearly independent basis vectors.

VECTOR SPACE MODEL FOR WORDS

Let us assume that the words in a corpus are considered as linearly independent basis vectors. If a corpus contains $|\mathscr{V}|$ words which are linearly independent, then every word represents an axis in the continuous vector space $\mathscr{R}$.

VECTOR SPACE MODEL FOR WORDS

Let us assume that the words in a corpus are considered as linearly independent basis vectors. If a corpus contains $|\mathscr{V}|$ words which are linearly independent, then every word represents an axis in the continuous vector space $\mathscr{R}$. Each word takes an independent axis which is orthogonal to other words/axes.

Let us assume that the words in a corpus are considered as linearly independent basis vectors. If a corpus contains $|\mathscr{V}|$ words which are linearly independent, then every word represents an axis in the continuous vector space $\mathscr{R}$. Each word takes an independent axis which is orthogonal to other words/axes. Then $\mathscr{R}$ will contain $|\mathscr{V}|$ axes.

Let us assume that the words in a corpus are considered as linearly independent basis vectors. If a corpus contains $|\mathscr{V}|$ words which are linearly independent, then every word represents an axis in the continuous vector space $\mathscr{R}$. Each word takes an independent axis which is orthogonal to other words/axes. Then $\mathscr{R}$ will contain $|\mathscr{V}|$ axes.

Examples

1. The vocabulary size of *emma corpus* is 7079. If we plot all the words in the real space $\mathscr{R}$, we get 7079 axes
2. The vocabulary size of *Google News Corpus corpus* is 3 million. If we plot all the words in the real space $\mathscr{R}$, we get 3 million axes

DOCUMENT VECTOR SPACE MODEL

- Vector space models are used to represent words in a continuous vector space $\mathscr{R}$

DOCUMENT VECTOR SPACE MODEL

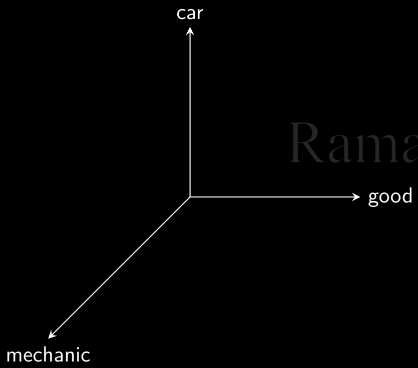
- Vector space models are used to represent words in a continuous vector space $\mathscr{R}$
- Combination of Terms represent a document vector in the word vector space

DOCUMENT VECTOR SPACE MODEL

- Vector space models are used to represent words in a continuous vector space $\mathscr{R}$
- Combination of Terms represent a document vector in the word vector space
- Very high dimensional space - several million axes, representing terms and several million documents containing several terms

EXAMPLE - BINARY INCIDENCE MATRIX

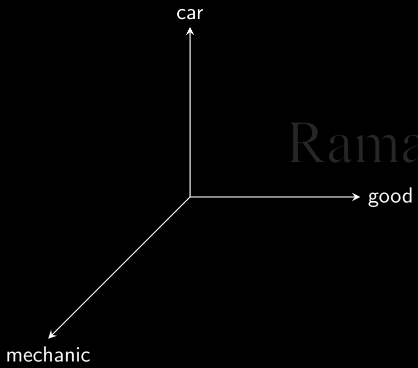
Let us consider three words - *good*, *car*, *mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	1	1	1
D2	1	0	1
D3	0	1	1

EXAMPLE - BINARY INCIDENCE MATRIX

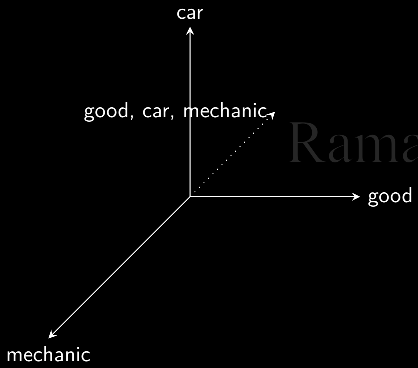
Let us consider three words - *good*, *car*, *mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	1	1	1
D2	1	0	1
D3	0	1	1

EXAMPLE - BINARY INCIDENCE MATRIX

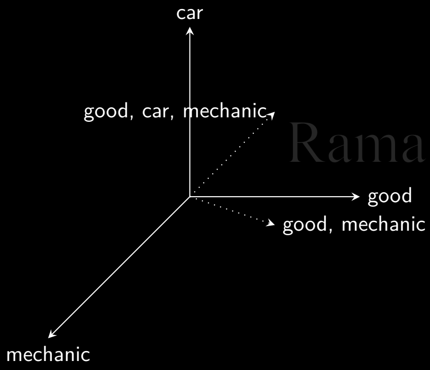
Let us consider three words - *good, car, mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	1	1	1
D2	1	0	1
D3	0	1	1

EXAMPLE - BINARY INCIDENCE MATRIX

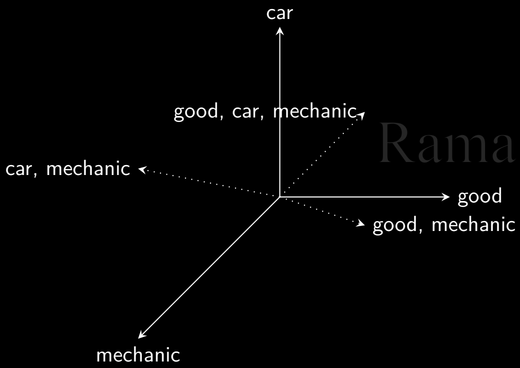
Let us consider three words - *good, car, mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	1	1	1
D2	1	0	1
D3	0	1	1

EXAMPLE - BINARY INCIDENCE MATRIX

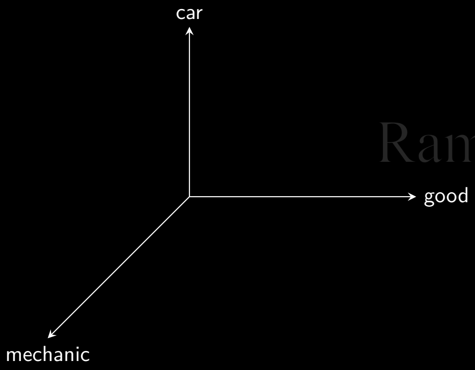
Let us consider three words - *good, car, mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	1	1	1
D2	1	0	1
D3	0	1	1

EXAMPLE - TF-IDF INCIDENCE MATRIX

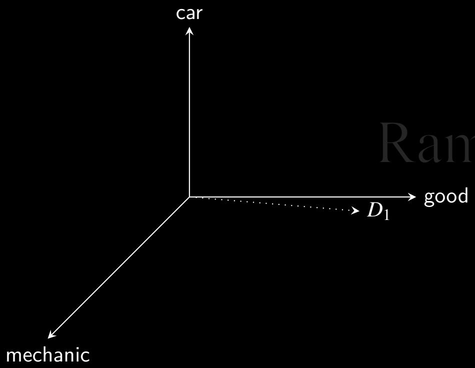
Let us consider three words - *good*, *car*, *mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

EXAMPLE - TF-IDF INCIDENCE MATRIX

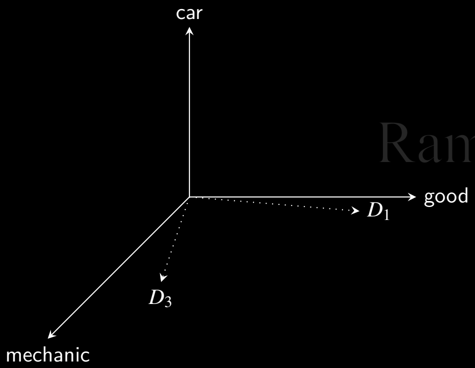
Let us consider three words - *good, car, mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

EXAMPLE - TF-IDF INCIDENCE MATRIX

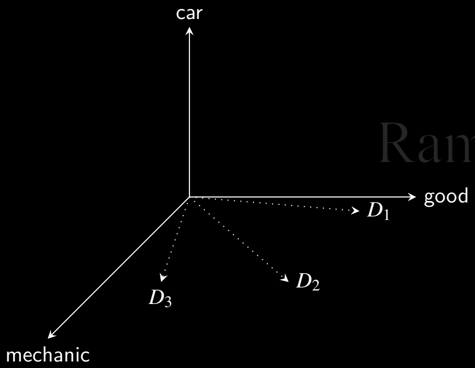
Let us consider three words - *good*, *car*, *mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

EXAMPLE - TF-IDF INCIDENCE MATRIX

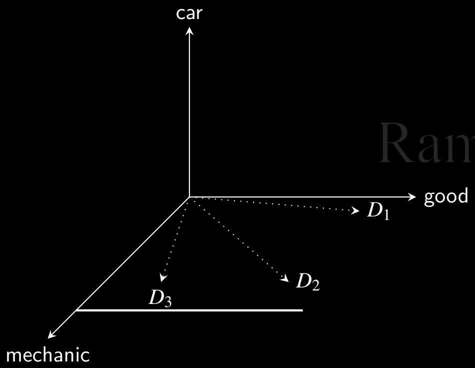
Let us consider three words - *good*, *car*, *mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

EXAMPLE - TF-IDF INCIDENCE MATRIX

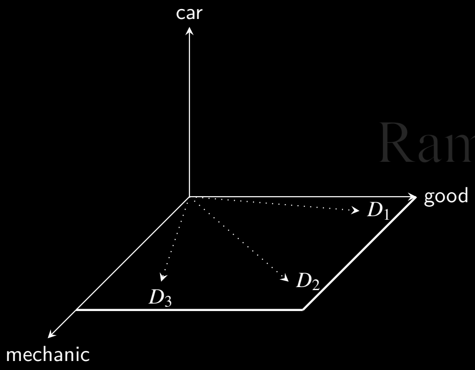
Let us consider three words - *good, car, mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

EXAMPLE - TF-IDF INCIDENCE MATRIX

Let us consider three words - *good*, *car*, *mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

DOCUMENT-TERM MATRIX

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12
t1	0.1	0.0	0.4	0.1	0.2	0.0	0.1	0.9	0.9	0.3	0.0	0.8
t2	0.1	0.0	0.4	0.1	0.2	0.0	0.1	0.9	0.9	0.3	0.0	0.8
t3	0.0	0.9	0.0	0.2	0.3	0.1	0.7	0.0	0.2	0.7	0.5	0.5
t4	0.0	0.9	0.3	0.9	0.5	0.1	0.9	0.3	0.8	0.4	0.1	0.4
t5	0.4	0.0	0.3	0.2	0.5	0.9	0.3	0.7	0.4	0.6	0.0	0.3
t6	0.6	0.0	0.4	0.7	0.3	0.3	0.9	0.1	0.9	0.0	0.0	0.3
t7	0.0	0.8	0.5	0.6	0.6	0.6	0.0	0.1	0.4	0.9	0.3	0.1
t8	0.4	0.0	0.6	0.5	0.5	0.1	0.7	0.1	0.5	0.3	0.8	0.1
t9	0.3	0.0	0.7	0.9	0.8	0.7	0.7	0.8	0.6	0.6	0.8	0.0
t10	0.0	0.5	0.5	0.0	0.2	0.0	0.0	0.1	0.3	0.4	0.5	0.3

The columns of the matrix represent the document as vectors. A document vector is represented by the terms present in the document

WEIGHTED-TF-IDF

Every element in the matrix represent tf-idf either in the plain form or in some of the weighted forms as given below:

$$tf.idf = tf \times log_{10}\left(\frac{N}{df_t}\right) \textbf{ or} \tag{1}$$

$$= w_{t,d} \times \left(\frac{N}{df_t}\right) \tag{2}$$

$$\text{where } w_{t,d} = \begin{cases} (1 + log_{10} tf_t), & \text{if } tf_{t,d} > 0 \\ 0 & otherwise \end{cases} \tag{3}$$

QUERY MODELING

Each query is modeled as a vector using the same attribute space of the documents.

$$q = \begin{bmatrix} q_{t_1} & q_{t_2} & q_{t_3} & \dots & q_{t_n} \end{bmatrix} \tag{4}$$

The relevancy ranking of a document depends on the distance of the document with respect to the query. The proximity of the query with every document is computed using distance measures.

DOCUMENT SIMILARITY

Earlier, using the binary incidence matrix, a query returned a set of documents whether the query keywords were found in documents or absent. It did not give any ranking for the retrieved documents. A similarity measure is a real-valued function that quantifies the similarity between two objects [1]. Some of the methods are given below.

$$\textbf{Euclidean Distance} - \mathscr{E}(\vec{d}_1, \vec{d}_2) = \sqrt{d_1^2 - d_2^2} \tag{5}$$

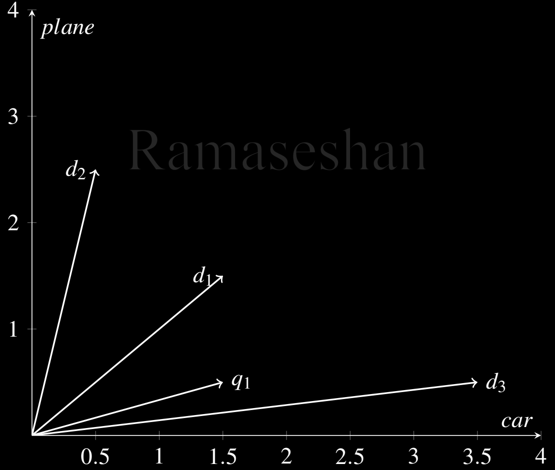
$$\textbf{Cosine Similarity} = \frac{\vec{d}_1.\vec{d}_2}{\left\|\vec{d}_1\right\|\left\|\vec{d}_2\right\|} = \frac{\vec{d}_1}{\left\|\vec{d}_1\right\|}.\frac{\vec{d}_2}{\left\|\vec{d}_2\right\|} \tag{6}$$

$$\text{Cosine distance} = 1 - \frac{\vec{d}_1.\vec{d}_2}{\left\|\vec{d}_1\right\|\left\|\vec{d}_2\right\|} = \frac{\vec{d}_1}{\left\|\vec{d}_1\right\|}.\frac{\vec{d}_2}{\left\|\vec{d}_2\right\|} \tag{7}$$

$$\textbf{Cluster similarity-}\mathscr{L}(\vec{d}_1, \vec{d}_2) = \frac{\vec{d}_1.\vec{d}_2}{\left\|\vec{d}_1\right\|_1} \tag{8}$$

WHICH MEASURE?

Euclidean measure does not work well for unequal sized vectors as the vectors are not normalized. We often use normalized correlation coefficient or cosine distance for similarity measure



PROXIMITY SCORE

A query is considered as a document vector[2]. The proximity of the query with every document is computed using a distance measure.

Cosine distance is preferred and it is easy to compute if the document vector distances are normalized. Proximity score is listed in the descending order and the documents within a predefined proximity score (angle) will be considered as relevant and retrieved.

GITHUB - ANLP

Demo code related to this course is available at
https://github.com/Ramaseshanr/ANLP[1]
You may use this repository as a playground to learn NLP. If you wish, you also contribute by adding new demos, fixing bugs, adding relevant comments to the code, etc. to help future learners

[1] https://github.com/Ramaseshanr/ANLP

DEMO - COSINE SIMILARITY FOR DOCUMENTS

The demo is available at - Cosine Distance Demo[2]

[2] https://github.com/Ramaseshanr/ANLP/blob/master/CosDistance.ipynb

INVERTED INDEX

- D1 - A ball is thrown from a bridge horizontally at a speed of 8m/s. Just before it hits the ground it is moving 50m/s vertically. How long was the ball in the air?
- D3- A ball is kicked at an angle of 35deg with the ground. (a) What should be the initial velocity of the ball so that it hits a target that is 30 meters away at a height of 1.8 meters?
- ...

Term	TF	Postings
ball 4	1	2,1,8,9
bridge	2	1
air	1	1
...	...	...

EXERCISE

Construct a tf.idf matrix using log weighting for the corpus Shakespeare play. Construct a query vector consisting of terms from the vocabulary and find the ranks of the plays with respect to the query

Ramaseshan

VECTOR REPRESENTATION OF WORDS

Let V be the unique terms and $|V|$ be the size of the vocabulary. Then every vector representing the word $\mathscr{R}^{|V|x1}$ would point to a vector in the V-dimensional space

ONE-HOT VECTOR - 1

Consider all the ≈39000 words (estimated tokens in English is ≈ 13M) in the Oxford Learner's pocket dictionary. We can represent each word as an independent vector quantity as follows in the real space $\mathscr{R}^{|V|X1}$

$$t^{a} = \begin{pmatrix} 1 \\ 0 \\ \dots \\ 0 \\ \dots \\ 0 \\ 0 \end{pmatrix} \quad t^{aback} = \begin{pmatrix} 0 \\ 1 \\ \dots \\ 0 \\ \dots \\ 0 \\ 0 \end{pmatrix} \quad \dots \; t^{zoom} = \begin{pmatrix} 0 \\ 0 \\ \dots \\ 0 \\ \dots \\ 1 \\ 0 \end{pmatrix} \quad t^{zucchini} = \begin{pmatrix} 0 \\ 0 \\ \dots \\ 0 \\ \dots \\ 0 \\ 1 \end{pmatrix}$$

This is a very simple codification scheme to represent words independently in the vector space. This is known as ***one-hot vector***.

ONE-HOT VECTOR - 2

In one-hot vector, every word is represented independently. The terms, *home*, *house*, *apartments*, *flats* are independently coded. With one-hot vector based model, the dot product

$$\left(t^{House}\right)^T \cdot t^{Apartment} = 0 \tag{9}$$

$$\left(t^{Home}\right)^T \cdot t^{House} = 0 \tag{10}$$

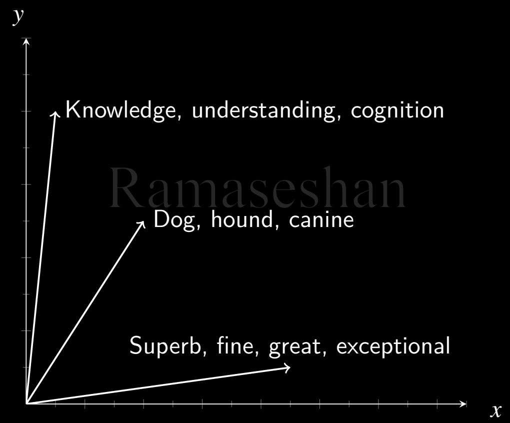
With one-Hot vector, there is no notion of similarity or synonyms.

The Goal of Word to Vector

- Reduce word-vector space into a smaller sub-space
- Encode the relationship among words

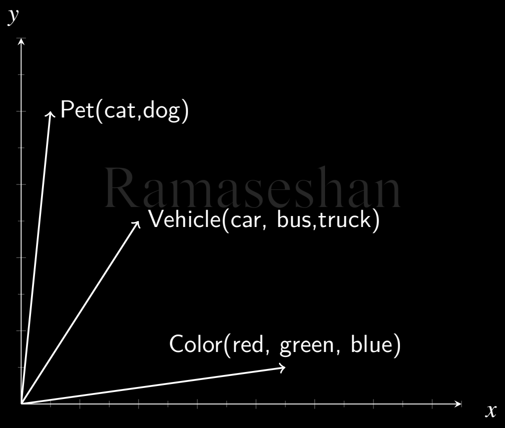
RELATIONSHIP AMONG TERMS - SYNONYMS

We could represent all the synonyms of a word in one axis

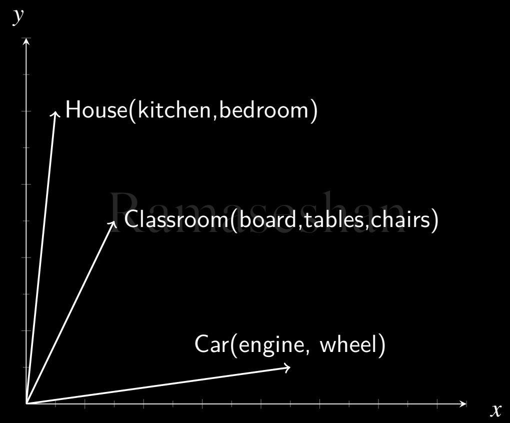


RELATIONSHIP AMONG TERMS - IS-A VECTOR

We could represent inheritance relationships of words as vectors.



RELATIONSHIP AMONG TERMS - HAS-A VECTOR - COMPOSITIONS



IS-A VECTOR

	Color	Animal	Fruit	Company Name
Apple	0	0	10	1850
Banana	0	0	165	0
Blackberry	0	0	156	190
Elephant	0	87	0	0
Fox	0	76	0	1
Goat	0	57	0	0
Green	145	0	0	0
Orange	454	0	213	134
Raspberry	0	0	197	74
Red	650	0	0	0
Sheep	0	132	0	0
Yellow	345	0	0	0

A simple example of Named Entity Extraction

The Apple Watch has a completely new user interface, different from the iPhone, and the 'crown' on the Apple Watch is a dial called the 'digital crown.' A key quality attribute of apple is its peel or skin color, which affects consumer preferences. Immature fruits are green, and as the fruit ripens the green may fade partially or completely, resulting in very pale cream to green background colors.

The ⟨**org**⟩**Apple** Watch has a completely new user interface, different from the iPhone, and the 'crown' on the ⟨**org**⟩**Apple** Watch is a dial called the 'digital crown.' A key quality attribute of ⟨**org**⟩**apple** is its peel or skin color, which affects consumer preferences. Immature fruits are green, and as the fruit ripens the green may fade partially or completely, resulting in very pale cream to green background colors.

REFERENCES

- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. *An Introduction to Information Retrieval*. Cambridge UP, 2009. Chap. 6, pp. 109–133.
- Manning et al. *Foundations of statistical natural language processing*. Mit Press. MIT Press, 1999. ISBN: 9780262133609. URL: https://books.google.co.in/books?id=YiFDxbEX3SUC.
- Scott Deerwester et al. “Indexing by latent semantic analysis”. In: *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE* 41.6 (1990), pp. 391–407.

You shall know a word by the company it keeps[3]

[3] Firth, J. R. 1957

CONTEXTUAL UNDERSTANDING OF WORDS

- The study of *meaning* and *context* should be central to linguistics
- Exploiting the context-dependent nature of words
- Language patterns cannot be accounted for in terms of a single system
- The *collocation*, gives enough clue to understand a word and its meaning
- *No study of meaning apart from context can be taken seriously* [4]

[4] Firth, J. R. 1957

DISAMBIGUATION OF BANK

Synset('bank.n.01')	sloping land (especially the slope beside a body of water)
Synset('depository-financial-institution.n.01')	a financial institution that accepts deposits and channels the money into lending activities
Synset('bank.n.03')	a long ridge or pile
Synset('bank.n.10')	a flight maneuver; aircraft tips laterally about its longitudinal axis (especially in turning)
Synset('bank.v.02')	enclose with a bank
Synset('bank.v.03')	do business with a bank or keep an account at a bank
Synset('bank.v.04')	act as the banker in a game or in gambling
Synset('bank.v.05')	be in the banking business
Synset('deposit.v.02')	put into a bank account
Synset('trust.v.01')	have confidence or faith in

DIFFERENT MEANINGS FOR THE WORD PROGRAM

Synset('plan.n.01')	a series of steps to be carried out or goals to be accomplished
Synset('program.n.02')	a system of projects or services intended to meet a public need
Synset('broadcast.n.02')	a radio or television show
Synset('platform.n.02')	a document stating the aims and principles of a political party
Synset('program.n.05')	an announcement of the events that will occur as part of a theatrical or sporting event
Synset('course_of_study.n.01')	an integrated course of academic studies
Synset('program.n.07')	(computer science) a sequence of instructions that a computer can interpret and execute
Synset('program.n.08')	a performance (or series of performances) at a public presentation
Synset('program.v.01')	arrange a program of or for
Synset('program.v.02')	write a computer program

SYNONYMS

small.a.01	['small', 'little']
minor.s.10	['minor', 'modest', 'small', 'small-scale', 'pocket-size', 'pocket-sized']
humble.s.01	['humble', 'low', 'lowly', 'modest', 'small']
little.s.07	['little', 'minuscule', 'small']
belittled.s.01	['belittled', 'diminished', 'small']
potent.a.03	['potent', 'strong', 'stiff']
impregnable.s.01	['impregnable', 'inviolable', 'secure', 'strong', 'unassailable', 'hard'] He has such an impregnable defense (Cricket-Very hard to find the gap between the bat and the pad)
solid.s.07	['solid', 'strong', 'substantial']
strong.s.09	['strong', 'warm']
firm.s.03	['firm', 'strong'] - firm grasp of fundamentals

CONTEXTUAL UNDERSTANDING OF TEXT

You shall know a word by the company it keeps - (Firth, J. R. 1957)

- In order to understand the word and its meaning, it not enough if we consider only the individual word
- The *meaning* and *context* should be central in understanding word/text
- Exploit the context-dependent nature of words
- Language patterns cannot be accounted for in terms of a single system
- The *collocation*, a particular word consistently co-occurs with the other words, gives enough clue to understand a word and its meaning

UNDERSTANDING A WORD FROM ITS CONTEXT

The view from the top of the mountain was

- awesome
- breathtaking
- amazing
- stunning
- astounding
- astonishing
- awe-inspiring
- extraordinary
- incredible
- unbelievable
- magnificent
- wonderful
- spectacular
- remarkable

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0
Physics	0	1	0	0	0	0	0	0	0

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0
Physics	0	1	0	0	0	0	0	0	0
He	0	0	0	0	1	0	0	0	0

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0
Physics	0	1	0	0	0	0	0	0	0
He	0	0	0	0	1	0	0	0	0
hates	0	0	0	1	0	1	0	0	0

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.
Example:Consider the following short documents:
1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0
Physics	0	1	0	0	0	0	0	0	0
He	0	0	0	0	1	0	0	0	0
hates	0	0	0	1	0	1	0	0	0
Maths	0	0	0	0	1	0	0	0	0

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0
Physics	0	1	0	0	0	0	0	0	0
He	0	0	0	0	1	0	0	0	0
hates	0	0	0	1	0	1	0	0	0
Maths	0	0	0	0	1	0	0	0	0
She	0	0	0	0	0	0	0	1	0

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0
Physics	0	1	0	0	0	0	0	0	0
He	0	0	0	0	1	0	0	0	0
hates	0	0	0	1	0	1	0	0	0
Maths	0	0	0	0	1	0	0	0	0
She	0	0	0	0	0	0	0	1	0
loves	0	0	0	0	0	0	1	0	1

CO-OCCURRENCE MATRIX

A co-occurrence is a combination of terms that are likely to be used in the same context. A co-occurrence matrix stores co-occurrences of words. The count of a pair of words that appears in a context window is represented as an element of a matrix.

Example:Consider the following short documents:

1. I love Physics 2. He hates Maths 3. She loves Biology

	I	love	Physics	He	hates	Maths	She	loves	Biology
I	0	1	0	0	0	0	0	0	0
love	1	0	1	0	0	0	0	0	0
Physics	0	1	0	0	0	0	0	0	0
He	0	0	0	0	1	0	0	0	0
hates	0	0	0	1	0	1	0	0	0
Maths	0	0	0	0	1	0	0	0	0
She	0	0	0	0	0	0	0	1	0
loves	0	0	0	0	0	0	1	0	1
Biology	0	0	0	0	0	0	0	1	0

- A sequence of two words is called a bigram
- A three-word sequence is called a trigram
- n-gram means a sequence of words of length n

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$<s>$	$<s>$Peter	$<s1><s2>$Peter

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper
Piper	Piper picked	Peter Piper picked

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper
Piper	Piper picked	Peter Piper picked
picked	picked a	Piper picked a

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper
Piper	Piper picked	Peter Piper picked
picked	picked a	Piper picked a
a	a peck	picked a peck

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper
Piper	Piper picked	Peter Piper picked
picked	picked a	Piper picked a
a	a peck	picked a peck
peck	peck of	a peck of

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper
Piper	Piper picked	Peter Piper picked
picked	picked a	Piper picked a
a	a peck	picked a peck
peck	peck of	a peck of
of	of pickled	peck of pickled

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper
Piper	Piper picked	Peter Piper picked
picked	picked a	Piper picked a
a	a peck	picked a peck
peck	peck of	a peck of
of	of pickled	peck of pickled
pickled	pickled peppers	of pickled peppers

N-GRAMS

Consider the tongue twister as four documents:
1. Peter Piper picked a peck of pickled peppers 2. A peck of pickled peppers Peter Piper picked. 3. If Peter Piper picked a peck of pickled peppers. 4. Where's the peck of pickled peppers Peter Piper picked?

Unigrams	Bigrams	Trigrams
$< s >$	$< s >$Peter	$< s1 >< s2 >$Peter
Peter	Peter Piper	$< s2 >$Peter Piper
Piper	Piper picked	Peter Piper picked
picked	picked a	Piper picked a
a	a peck	picked a peck
peck	peck of	a peck of
of	of pickled	peck of pickled
pickled	pickled peppers	of pickled peppers
peppers	peppers	–

COLLOCATIONS

Collocations is a juxtaposition of two or more words that more often occur together than ny chance.

- Poverty is a ***major problem*** for many countries
- Ram has a ***powerful computer***
- I had a ***brief chat*** with Raj
- I could not see anything in the room, it was ***pitch dark*** inside
- The crime was committed in ***broad daylight*** - We don't use wide, large, big daylight
- I wish I had a ***strong tea*** - we don't use powerful, tough
- The ***heavy rain*** prevented us from playing outside - We don't use strong rain
- Someone ***knocked*** on the front ***door***

CREATION OF SEMANTICALLY CONNECTED VECTORS

- Identify a model that enumerates the relationships between terms and documents
- Identify a model that tries to put similar items closer to each other in some space or structure
- A model that discovers/uncovers the semantic similarity between words and documents in the latent semantic domain
- Develop a distributed word vectors or dense vectors that captures the linear combination of word vectors in the transformed domain

METHODS TO CREATE DENSE VECTORS

- Latent Semantic Analysis or Latent Semantic Indexing
- Neural networks using skip grams and CBOW
  - CBOW - uses surrounding words to predict the center of words
  - Skip grams use center of words to predict the surrounding words
- Brown clustering - statistical algorithms for assigning words to classes based on the frequency of their co-occurrence with other words
- Hyperspace Analogue to Language - HAL
- Correlated Occurrence Analogue to Lexical Semantic - COALS
- Global Vectors - GloVe

- Sparse vectors are too long and not very convenient as features machine learning
- Abstracts more than just frequency counts
- It captures neighborhood words that are connected by synonyms
  - Consider these two documents (1) Automobile association (2) car driver
  - Connects the neighbor of Automobile and the neighbor of car
  - "Automobile association" with "car driver" - driver and association could be connected using the similar words ***Automobile and car***
- What is Xalapa?

WORD SIMILARITY

- Sparse vectors are too long and not very convenient as features machine learning
- Abstracts more than just frequency counts
- It captures neighborhood words that are connected by synonyms
  - Consider these two documents (1) Automobile association (2) car driver
  - Connects the neighbor of Automobile and the neighbor of car
  - "Automobile association" with "car driver" - driver and association could be connected using the similar words ***Automobile and car***
- What is Xalapa?
- Every one likes Xalapa

WORD SIMILARITY

- Sparse vectors are too long and not very convenient as features machine learning
- Abstracts more than just frequency counts
- It captures neighborhood words that are connected by synonyms
  - Consider these two documents (1) Automobile association (2) car driver
  - Connects the neighbor of Automobile and the neighbor of car
  - "Automobile association" with "car driver" - driver and association could be connected using the similar words ***Automobile and car***
- What is Xalapa?
- Every one likes Xalapa
- Xalapa is served in the morning

WORD SIMILARITY

- Sparse vectors are too long and not very convenient as features machine learning
- Abstracts more than just frequency counts
- It captures neighborhood words that are connected by synonyms
  - Consider these two documents (1) Automobile association (2) car driver
  - Connects the neighbor of Automobile and the neighbor of car
  - "Automobile association" with "car driver" - driver and association could be connected using the similar words ***Automobile and car***
- What is Xalapa?
- Every one likes Xalapa
- Xalapa is served in the morning
- Main Ingredients - black beans, avocado, tortilla, cumins, tomato puree

WORD SIMILARITY

- Sparse vectors are too long and not very convenient as features machine learning
- Abstracts more than just frequency counts
- It captures neighborhood words that are connected by synonyms
  - Consider these two documents (1) Automobile association (2) car driver
  - Connects the neighbor of Automobile and the neighbor of car
  - "Automobile association" with "car driver" - driver and association could be connected using the similar words ***Automobile and car***
- What is Xalapa?
- Every one likes Xalapa
- Xalapa is served in the morning
- Main Ingredients - black beans, avocado, tortilla, cumins, tomato puree

Intuition

Xalapa is food
Xalapa is served as breakfast
Xalapa is a breakfast item like chapathi roll

Xalapa and chapathi roll are related as the context is breakfast
Xalapa and chapathi roll are related as they both are vegetarian

You shall know a word by the company it keeps

- Firth, 1957

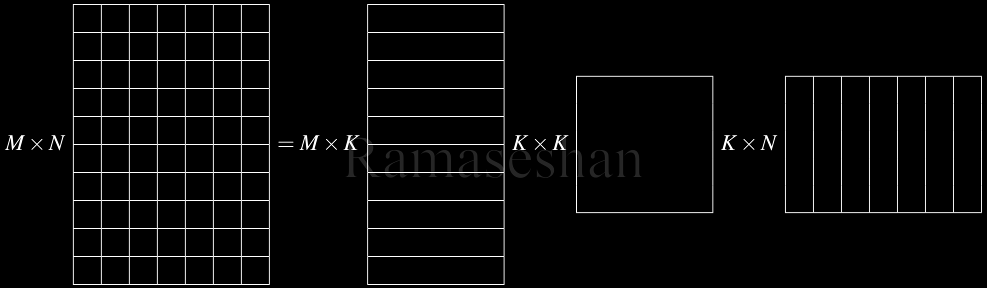
Singular value decomposition is a method to factorize a rectangular/square matrix into three matrices.

$$A = U\Sigma V^T \tag{11}$$

where A is an MXN matrix

- U is the $M \times K$ matrix
- Σ is a diagonal matrix of size $K \times K$
- V^T is the $K \times N$ matrix
- The row vectors of U are called as the left-singular vectors
- Row vectors of U form an orthogonal set
- The columns of V^T are called as the right singular matrix
- The rows of V^T form an orthonormal set
- The Σ is the singular matrix. It is a diagonal matrix and i1ts values are arranged in the descending order.

SVD



SINGULAR VALUES

- It is a diagonal matrix
- Singular values are arranged in the descending order
- Highest order dimension captures the most variance in the original dataset or most of the information related to term-document matrix
- The next higher dimension captures the next higher variance in the original data set
- Singular values reflect the major associative patterns in the data, and ignore the smaller, less important influences

DEMO

1. SVD Projection on a 2D image
2. Query Processing
3. Topic Modeling

DIMENSIONALITY REDUCTION

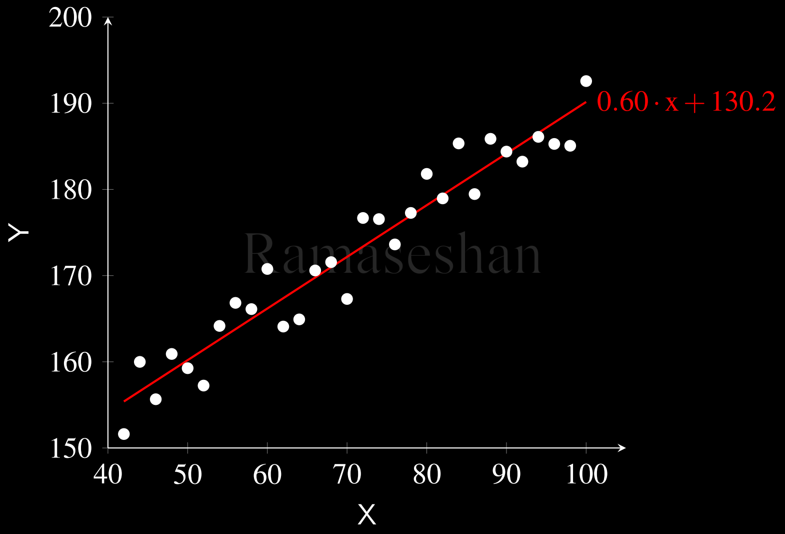
- SVD is better suited for measuring the similarity between documents, by exploiting the similarity patterns that exist in the word co-occurrence[2]
- The co-occurring terms are mapped into the same dimension thereby reducing the dimensions
- Increases the similarity representation of the semantically similar documents

SVD takes the original Matrix A in the m-dimensional space and transforms it as $\hat{A}$ in the reduced dimensional space $k \leq m$

$$\Delta = \left\|A - \hat{A}\right\|_2 \tag{12}$$

where $\|.\|$ is the L_2 norm for the matrices.

LEAST-SQUARE METHODS



IMPORTANT EQUATIONS IN SVD

Since U and V are orthonormal matrices,

$$U^T U = V^T V = I \tag{13}$$

$$\begin{aligned} A^T A &= V\Sigma^T U^T U \Sigma V^T \\ &= V\Sigma^2 V^T \end{aligned} \tag{14}$$

$$\begin{aligned} AA^T &= U\Sigma V^T V \Sigma^T U^T \\ &= U\Sigma^2 V^T \end{aligned} \tag{15}$$

$$U^T A = U^T U \Sigma D^T = \Sigma V^T \tag{16}$$

$$u_q = u_q^T U \Sigma^{-1}, \quad where \tag{17}$$

$$\Sigma_3^{-1} = \begin{bmatrix} \frac{1}{\sigma_{11}} & 0 & 0 \\ 0 & \frac{1}{\sigma_{22}} & 0 \\ 0 & 0 & \frac{1}{\sigma_{33}} \end{bmatrix} \tag{18}$$

SUMMARY OF SVD

- Find a new set of dimensions or attributes that capture the variability of the data
- Identify the strongest pattern ni the data
- Most variability is captured by a small fraction of the total set of dimensions
- Patterns among the terms are captured by the left-singular matrix
- Patterns among the documents are captured by the right-singular matrix
- The eigen vectors associated with the largest eigen value indicates the direction of largest variance[5]
-

[5] Pang-Ning Tan et al, "Introduction to Data Mining"2007

LATENT STRUCTURE IN THE DOCUMENT

- There is some structure in the documents
- The structure is described by the terms in the documents
- Similar terms appear in similar context
- Similar documents are described similar terms
- There is a semantic structure hidden in document though they aren't clearly visible
- The semantic structure is hidden by words that may not contribute to the meaningful retrieval
- Is it possible to retrieve a document using this latent semantic structure?
- Users construct their similar queries using different terms based on their linguistic habits, knowledge and the terms may not present in the corpus

CHOICE OF METHODS TO UNCOVER LATENT SEMANTICS

The goal is to find and fit a model that uncovers the relationships between terms and documents

- A model that places the co-occurring terms closer to each other in the latent space
- A model that places the similar document closer to each other in the latent space
- To build a model, we need to make use of the occurrences of terms to estimate the parameters
- The ideal model should predict a document even if the association between query terms and document was not observed

WHY SVD FOR LSI?

- Need a function that transforms the original matrix using the basis vector
- Eigenvectors are orthogonal
- Eigenvectors represents the basis of the original term-document matrix
- SVD transforms the original matrix into a latent space
  - $A = U\Sigma V^T$
  - U represents the latent space of the term
  - V represents the latent space of the document
  - Σ represents the eigen values
  - Every row U represents p-dimensional vector per word
  - Every row V represents n-dimensional vector per document

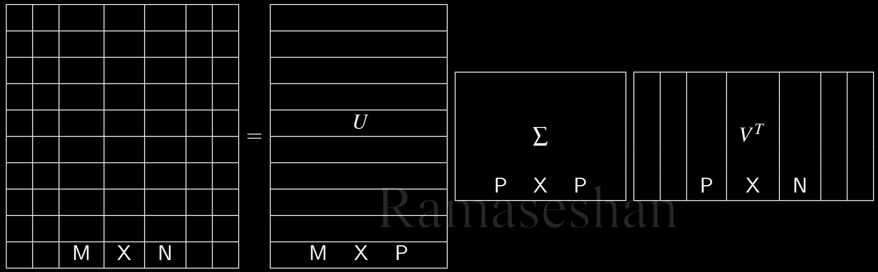
LSI/LSA

- Projects Term-Document relationship into a latent-semantic dimensions[3]
- Similar terms are projected into the same dimension - dimensionality reduction
- Similar documents are projected into the same dimension
- Dimensions of the reduced space correspond to the axes of the greatest variation
- Selection of p is large enough to fit all the patterns in the data and small enough to be able to fit all the data with minimum error.
- Words and documents of the latent space can be represented as points in Euclidean space

QUERIES IN THE LSI

- Queries can be considered as a pseudo document as it contains terms
- Queries and documents are projected into the latent semantic space
- Query and documents may have high cosine similarity even if the query the document share few terms

LSI -2

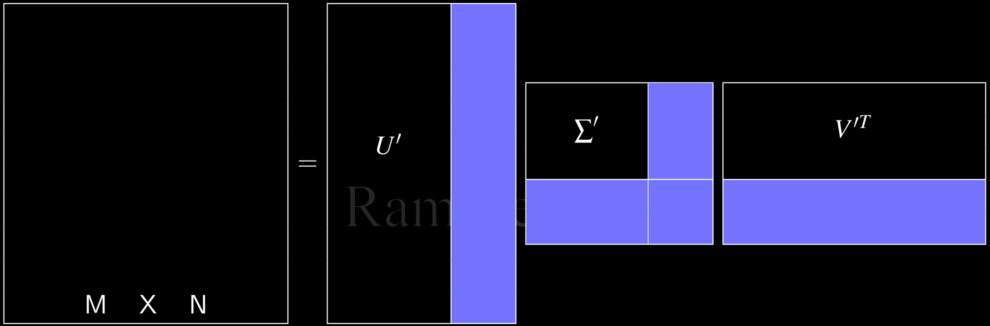


LSI maps the similar terms into the same direction, thereby reducing the dimensions of the space and increases the representations of semantically similar documents and words

APPLICATIONS

- Information discovery
- Clustering of documents
- Text summarization
- Topic modeling

TOPIC MODELING - REDUCED DIMENSIONS

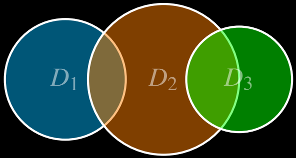


Topic model is a statistical modeling technique that find abstract ”topics” from a collection of documents using the most common words present each document. It minimizes the topic using the LSI

LSI/LSA

- It reduces the dimensionality of the Term-Document matrix.
- It keeps dimensions that capture the most variation in the document
- It causes documents with similar topical content to be close to one another
- Dimensions loosely correspond with topic boundaries
- It can reduce the dimensions from thousands to 100-300

LSA/LSI captures relationship among documents very well. For example, if two documents do not share any words but share words with another document, then all three are projected in the same space



WORD EMBEDDING

- Word embeddings are a family of natural language processing techniques aiming at mapping semantic meaning into a geometric space
- Vector space models (VSMs) represent (embed) words in a continuous vector space where semantically similar words are mapped to nearby points
- Count-based methods compute the statistics of how often some word co-occurs with its neighbor words in a large text corpus, and then map these count-statistics down to a small, dense vector for each word.
- Predictive models directly try to predict a word from its neighbors in terms of learned small, dense embedding vectors (considered parameters of the model)

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. *An Introduction to Information Retrieval*. Cambridge UP, 2009. Chap. 6, pp. 109–133.

Manning et al. *Foundations of statistical natural language processing*. Mit Press. MIT Press, 1999. ISBN: 9780262133609. URL: https://books.google.co.in/books?id=YiFDxbEX3SUC.

Scott Deerwester et al. “Indexing by latent semantic analysis”. In: *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE* 41.6 (1990), pp. 391–407.