# Different ways of representing documents and some similarity measures for document clustering

# Representation 1: Binary term-document incidence matrix

- Each document is represented by a binary vector $\in \{0,1\}^{|V|}$ , Where |v| is size if vocabulary which will be the unique words in the document

Let two documents containing text:

D1: John likes to watch movies. Mary likes movies too.
D2: John likes to watch football games.

| List of words | John | likes | to | watch | movies | Mary | too | football | games |
|---|---|---|---|---|---|---|---|---|---|
| D1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| D2 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

# Representation 2: Term-document count matrices

- Represent each word of document in terms of number of occurrence (denoted as - $tf_{t,d}$ , the number of times term *t* occurs in document '*d*'):
  - Each document is a count vector in |V| dimensional space

Let two documents containing text:

D1: John likes to watch movies. Mary likes movies too.
D2: John likes to watch football games.

| List of words | John | likes | to | watch | movies | Mary | too | football | games |
|---|---|---|---|---|---|---|---|---|---|
| D1 | 1 | 2 | 1 | 1 | 2 | 1 | 1 | 0 | 0 |
| D2 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

# Representation 2 : Disadvantage

- Above representation known as *Bag of words* model (1-gram)
- Disadvantage:
  - Vector representation doesn't consider the ordering of words in a document as documents .
    - Example : '*John is quicker than Mary'* and '*Mary is quicker than John'* have the same vectors
      - Solution : Use n-gram model where n>1

# Representation 2 : Disadvantage

–A document with 10 occurrences of the term is more relevant than a document with 1 occurrence of the term. But, relevance does not increase proportionally with term frequency.

***–Rare terms are more informative than frequent terms (stop words like is, am, are…..)***

–Example:

- Consider a term in the query that is rare in the collection (e.g., *arachnocentric*)
- A document containing this term is very likely to be relevant to the query *arachnocentric*

→We want a higher weight for rare terms like *arachnocentric.*

But lower weights than for rare terms

# Solution: tf-idf scheme

- Statistical measure used to evaluate how important a word is to a document in a collection or corpus
  - **TF: Term Frequency**, which measures how frequently a term(word) occurs in a document.

    TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document)
  - **IDF: Inverse Document Frequency**, which measures how important a term is.

    IDF(t) = $\log_e$(Total number of documents / Number of documents with term t in it).

# Example

- Consider a document containing 100 words
- wherein the word *cat* appears 3 times.
- The term frequency (i.e., tf) for *cat* is then (3 / 100) = 0.03.
- Now, assume we have 10 million documents and the word *cat* appears in one thousand of these.
- Then, the inverse document frequency (i.e., idf) is calculated as log(10,000,000 / 1,000) = 4.

- Thus, the Tf-idf weight is the product of these quantities: 0.03 * 4 = 0.12.

# Representation 3 : Word2vec

•Set of models that are used to produce ***word embeddings*** where words or phrases from the vocabulary are mapped to vectors of real numbers.

•input a large corpus of text and produces a vector space, typically of *several hundred dimensions*, with each unique word in the corpus being assigned a corresponding vector

•Captures the syntax and semantic relations between two documents.

# Word2vec Continued……

•The purpose and usefulness of Word2vec is to group the vectors of similar words together in vector space.

•Given enough data, *usage and contexts, Word2vec can make highly accurate guesses about a word's meaning based on past appearances*

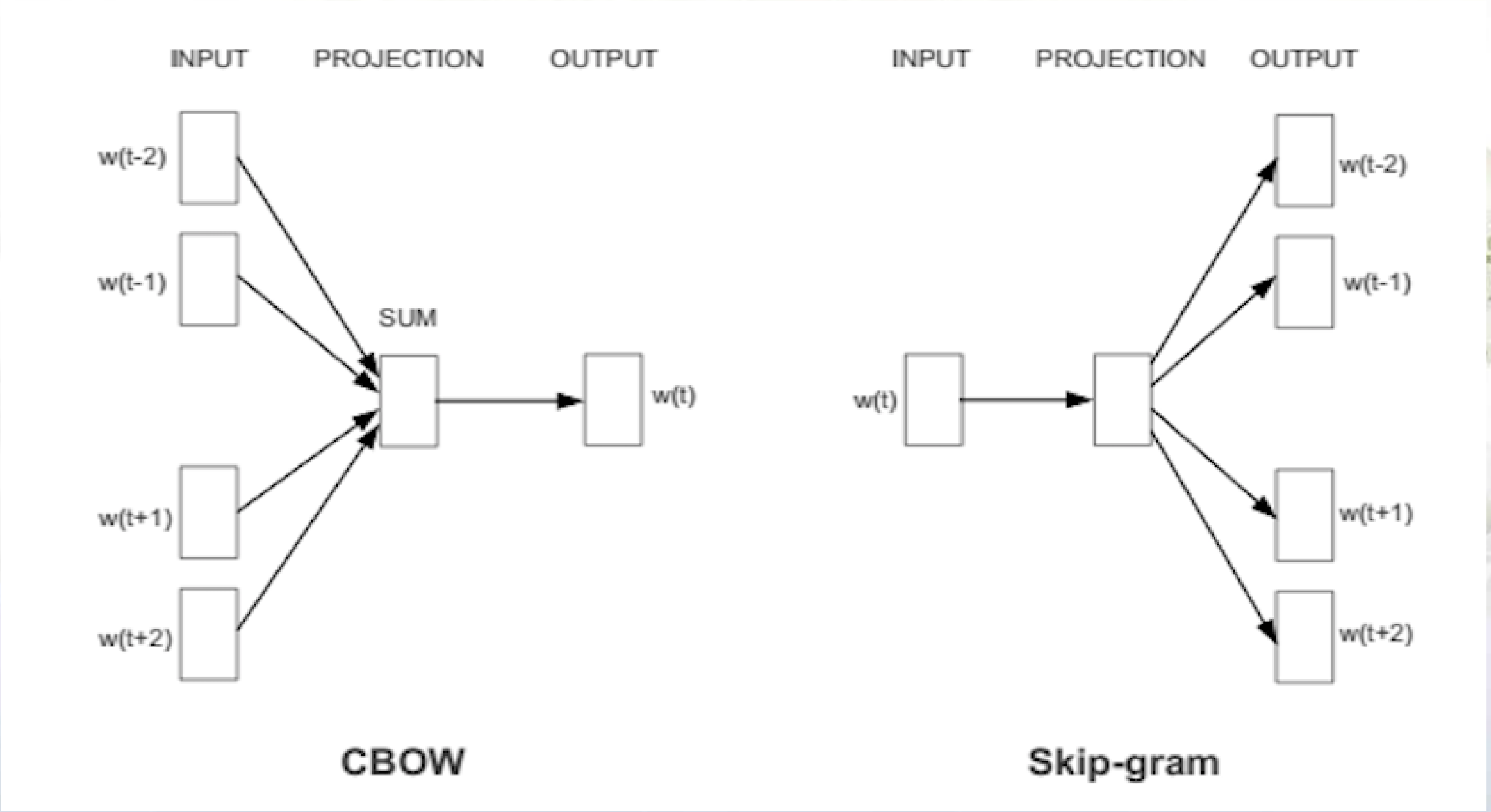
•Here's a list of words associated with "Sweden" using Word2vec, in order of proximity:

•.

| Word | Cosine distance |
|---|---|
| norway | 0.760124 |
| denmark | 0.715460 |
| finland | 0.620022 |
| switzerland | 0.588132 |
| belgium | 0.585835 |
| netherlands | 0.574631 |
| iceland | 0.562368 |
| estonia | 0.547621 |
| slovenia | 0.531408 |

# Word2vec Continued……

It does so in one of two ways, either using context to predict a target word (a method known as continuous bag of words, or CBOW), or using a word to predict a target context, which is called skip-gram



# Word2vec :  A snapshot

Word vector

```
-0.1096,0.36780000000000002,-0.2024,0.52070000000000005,-0.25740000000000002,0.6794999999999999
0.19040000000000001,0.35620000000000002,-0.27250000000000002,0.1145,-0.31009999999999999,0.474
0.6139,1.2233000000000001,-0.63049999999999995,-0.36420000000000002,-1.1585000000000001,1.0278,
-0.24690000000000001,0.57250000000000001,0.87519999999999998,1.8522000000000001,-0.509199999999
-0.1701,-0.1143,-0.064899999999999999,0.1232,-0.40910000000000002,0.15459999999999999,0.022499
-0.074999999999999997,0.1246,-0.25969999999999999,0.55600000000000005,-0.52510000000000001,0.6
-0.18429999999999999,0.27129999999999999,-0.47939999999999999,0.22800000000000001,-0.596700000
0.0356,0.2409,-0.39639999999999997,0.1608,-0.56320000000000003,0.52459999999999996,0.200699999
-0.30130000000000001,0.33939999999999998,-0.26840000000000003,0.34029999999999999,-0.487099999
0.0436,0.40970000000000001,-0.2402,0.55820000000000003,-0.65310000000000001,0.5525999999999999
0.064600000000000005,0.6835,-0.72660000000000002,0.11409999999999999,-0.3866,0.380900000000000
0.2001,0.37369999999999998,-0.30159999999999998,0.091999999999999998,-0.31719999999999998,0.45
-0.111,0.26919999999999999,-0.39290000000000003,0.41880000000000001,-0.59550000000000003,0.664
```

# How to generate document vector using word2vec?

- After generating word vector of all words in the documents, these word vectors are get averaged to obtain the document vector.

- Example:

Let a document have three words w1, w2 w3

their corresponding word vector : v1,v2,v3

document vector= (v1+v2+v3)/3

# Computing similarity/dissimilarity between two document vectors

- Cosine Similarity
- Symmetric conditional probability based similarity
- Correlation
- Euclidean Distance
- Squared Euclidean distance
- Jaccard coefficient

# Cosine similarity

- $d = (x_1, x_2, x_3, ..., x_n)$ →vector in an n-dimensional vector space.

Length of x is given by (extension of Pythagoras's theorem) :

$|d|^2 = x_1^2 + x_2^2 + x_3^2 + ... + x_n^2$

$|d| = (x_1^2 + x_2^2 + x_3^2 + ... + x_n^2)$ ½

- If $d_1$ and $d_2$ are document vectors: Inner product (or dot product) is given by :

$d_1 . d_2 = x_{11}x_{21} + x_{12}x_{22} + x_{13}x_{23} + ... + x_{1n}x_{2n}$

Cosine angle between the docs $d_1$ and $d_2$ determines doc similarity:

$$Cos(\theta) = \frac{d_1 . d_2}{|d_1||d_2|}$$

$\cos(\theta) = 1$; documents exactly the same; $= 0$, totally different

# Example

| | ant | bee | cat | dog | eel | fox | gnu | hog | *length* |
|---|---|---|---|---|---|---|---|---|---|
| $d_1$ | 1 | 1 | | | | | | | $\sqrt{2}$ |
| $d_2$ | 1 | 1 | | 1 | | | | 1 | $\sqrt{4}$ |
| $d_3$ | | | 1 | 1 | 1 | 1 | 1 | | $\sqrt{5}$ |

Ex: *length* $d_1 = (1^2+1^2)^{1/2}$

| | $d_1$ | $d_2$ | $d_3$ |
|---|---|---|---|
| $d_1$ | 1 | 0.71 | 0 |
| $d_2$ | 0.71 | 1 | 0.22 |
| $d_3$ | 0 | 0.22 | 1 |

# SCP based similarity

- It is calculated as

$$SCP(w_1, w_2) = \frac{P(w_1, w_2)^2}{P(w_1) \times P(w_2)}$$

where ,

- *P(.,.) is the* joint probability of two tokens (*$w_1$ and $w_2$) appearing in the same word feature vector*
- *P(.) is the* marginal probability of any token appearing in a word feature vector.

# Example input

- they will call to  office in Chennai
- he needs to call to his mother in Hyderabad
- why dont you call to your office and take leave today

# Corresponding Word vocabulary

1. They
2. Will
3. Call
4. To
5. Office
6. In
7. Chennai
8. He
9. Need
10. His
11. Mother
12. Hyderabad
13. Why
14. Do-nt
15. You
16. Your
17. And
18. Take
19. Leave
20. today

| | They | Will | Call | To | Office | In | Chennai | He | Need | His | Mother | Hydrabad | Why | Dont | You | Your | And | Take | Leave | today |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| They | 1 | | | | | | | | | | | | | | | | | | | |
| Will | | 1 | | | | | | | | | | | | | | | | | | |
| Call | | | 1 | | 0.7 | | | | | | | | | | | | | | | |
| To | | | | 1 | | | | | | | | | | | | | | | | |
| Office | | | 0.7 | | | | | | | | | | | | | | | | | |
| In | | | | | | | | | | | | | | | | | | | | |
| Chennai | | | | | | | | | | | | | | | | | | | | |
| He | | | | | | | | | | | | | | | | | | | | |
| Need | | | | | | | | | | | | | | | | | | | | |
| His | …………………………………… | | | | | | | | | | | | | | | | | | | |
| Mother | | | | | | | | | | | | | | | | | | | | |
| Hydrabad | | | | | | | | | | | | | | | | | | | | |
| Why | | | | | | | | | | | | | | | | | | | | |
| Dont | | | | | | | | | | | | | | | | | | | | |
| You | | | | | | | | | | | | | | | | | | | | |
| Your | | | | | | | | | | | | | | | | | | | | |
| And | | | | | | | | | | | | | | | | | | | | |
| Take | | | | | | | | | | | | | | | | | | | | |
| Leave | | | | | | | | | | | | | | | | | | | | |
| today | | | | | | | | | | | | | | | | | | | | |

- Example: To calculate SCP(call,office)
- P(call,office)$^2$ → $(2/3)^2$
- P(call) → (3/3)\
- P(office) → (2/3)

- SCP(call,office)={(4/9)*(3/3)*(3/2)}=0.7

In order to calculate similarity between two documents this SCP measure can be used in the following way,

$$S(d_i, d_j) = \frac{1}{||d_i||||d_j||} \sum_{r=1}^{||d_i||} \sum_{b=1}^{||d_j||} SCP(w_i^r, w_j^b)$$

Where, $S(d_i,d_j) \rightarrow$ similarity between two documents $d_i$ and $d_j$.

# Correlation

- The dot product of the term vectors of two documents gives an indication to the correlation between the documents.
- This is based on the intuition that documents that describe similar topic are more likely to share words.

The correlation between two documents $d_i$ and $d_j$ due to textual content is given by

$$C_{ij} = d_i \bullet d_j$$

# Example of correlation

| | ant | bee | cat | dog | eel | fox | gnu | hog | *length* |
|---|---|---|---|---|---|---|---|---|---|
| $d_1$ | 1 | 1 | | | | | | | $\sqrt{2}$ |
| $d_2$ | 1 | 1 | | 1 | | | | 1 | $\sqrt{4}$ |
| $d_3$ | | | 1 | 1 | 1 | 1 | 1 | | $\sqrt{5}$ |

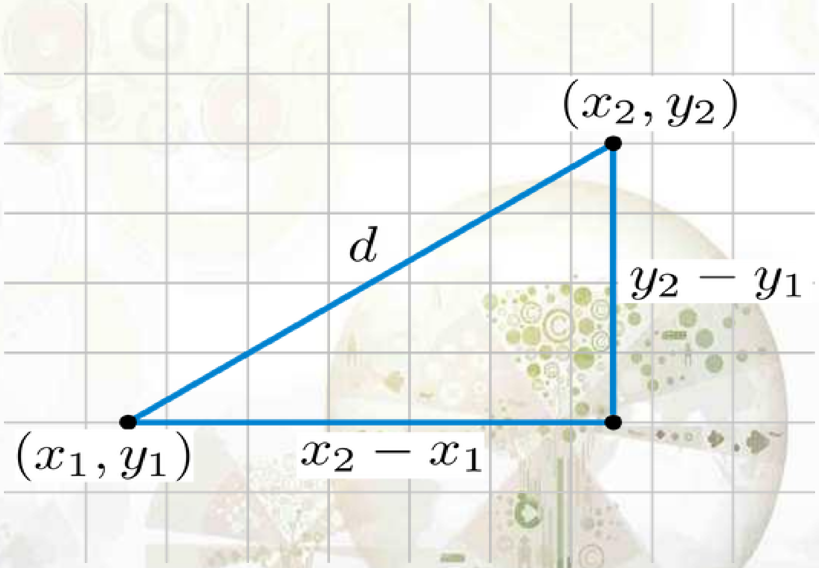
Ex: *length* $d_1 = (1^2+1^2)^{1/2}$

$$C_{d1d2} = d_1 \bullet d_2 = (1*1)+(1*1)+(0*0)+(0*1)+$$
$$(0*0)+(0*0)+(0*0)+(0*1)=2$$

# Euclidean distance

- Adds up all the squared distances between data points x and y, and takes the square root of the result.

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$




- Example : (x1,y1)=(2,4) , (x2,y2)=(2,2)

$$\sqrt{(2-2)^2 + (4-2)^2} = 2$$

# Euclidean Squared Distance Metric

•The Euclidean Squared distance metric uses the same equation as the Euclidean distance metric.

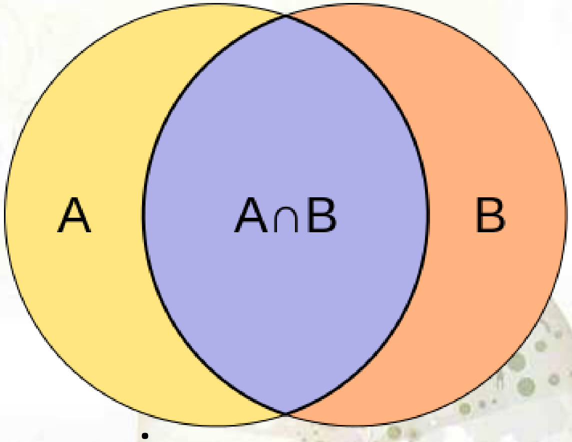
•It does not take the square root. As a result, ***clustering with the Euclidean Squared distance metric* is faster than clustering with the regular Euclidean distance.**

$$\sum_{i=1}^{n}(x_i - y_i)^2$$

# Jaccard Cofficient

- measures similarity between finite sample sets
- also known as **Intersection over Union**

$$J(A,B)=\frac{|A\cap B|}{|A\cup B|}$$




- $J(A,A) = 1$
- $J(A,B) = 0$ if $A \cap B = 0$
- *A* and *B* don't have to be the same size.
- Always assigns a number between 0 and 1.

- Example : A={7,3,2,4,1}, B={ 4,1,9,7,5}

$$J(A,B)=\frac{|\{1,4,7\}|}{|\{1,2,3,4,5,7,9\}|}=\frac{3}{7}=0.429$$

Thank you!!