



ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

CENG 463: Introduction to Natural Language Processing

Word Vectors: Bag-of-Words

Asst. Prof. Cagri Toraman
Computer Engineering Department
ctoraman@ceng.metu.edu.tr

28.10.2025

** The Course Slides are subject to [CC BY-NC](#). Either the original work or a derivative work can be shared with appropriate attribution, but only for noncommercial purposes.*

Text Semantics

Ambiguity

Every fifteen minutes a woman in this country gives birth. Our job is to find this woman, and stop her!

Text Semantics

Expressivity

She gave the book to Tom

She gave Tom the book

Text Semantics

Sparsity

Power Law (Zipf’s Law)

any word		nouns	
Frequency	Token	Frequency	Token
1,698,599	the	124,598	European
849,256	of	104,325	Mr
793,731	to	92,195	Commission
640,257	and	66,781	President
508,560	in	62,867	Parliament
407,638	that	57,804	Union
400,467	is	53,683	report
394,778	a	53,547	Council
263,040	I	45,842	States

Text Semantics

How to represent the knowledge a human has/needs?

What is the “meaning” of a word or sentence?

Text Semantics

Concepts or word senses

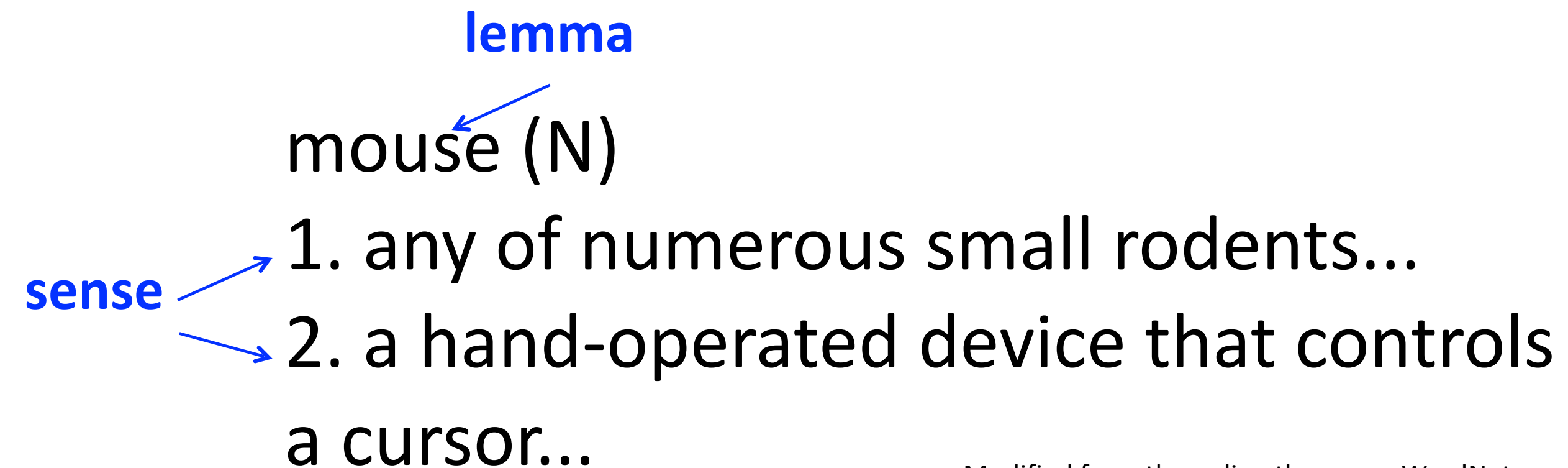
- Have a complex many-to-many association with **words** (homonymy, multiple senses)

Have relations with each other

- Synonymy
- Antonymy
- Similarity
- Relatedness
- Connotation

Text Semantics

Lemmas and senses



Modified from the online thesaurus WordNet

A **sense** or “**concept**” is the meaning component of a word
Lemmas can be **polysemous** (have multiple senses)

Text Semantics

Synonyms have the same meaning in some or all contexts.

- filbert / hazelnut
- couch / sofa
- big / large
- automobile / car
- vomit / throw up
- water / H₂O

Text Semantics

Words with **similar** meanings. Not synonyms, but sharing some element of meaning

car, bicycle

cow, horse

Text Semantics

Relatedness

Words can be related in any way, perhaps via a semantic frame or field

- coffee, tea: **similar**
- coffee, cup: **related**, not similar

Text Semantics

Antonymy: Senses that are opposites with respect to only one feature of meaning

dark/light	short/long	fast/slow	rise/fall
hot/cold	up/down		in/out

Text Semantics

Positive **connotations** (*happy, great, love*)

Negative connotations (*sad, terrible, hate*)

Can be subtle:

Positive connotation: *copy, replica, reproduction*

Negative connotation: *fake, knockoff, forgery*

Text Semantics

Can we build a theory of how to represent word meaning that accounts for those semantic concepts?

Vector semantics

Basic model for language processing

Handles many of our goals

Text Semantics

Idea 1: Defining meaning by linguistic distribution

Idea 2: Meaning as a point in multidimensional space

Text Semantics

The meaning of a word is its use in the language.

Words are defined by their environments (the words around them).

*I'm going to the **bank** to deposit my paycheck.*

*The **bank** of the river was lined with trees.*

Text Semantics

3 affective dimensions for a word

- valence:** pleasantness
- arousal:** intensity of emotion
- dominance:** the degree of control exerted

A word is a vector in 3-space.

	Word	Score		Word	Score
Valence	love	1.000		toxic	0.008
	happy	1.000		nightmare	0.005
Arousal	elated	0.960		mellow	0.069
	frenzy	0.965		napping	0.046
Dominance	powerful	0.991		weak	0.045
	leadership	0.983		empty	0.081

Text Semantics

Defining meaning as a point in space based on distribution

Each word = a vector (not just "good" or " w_{45} ")

Similar words are "**nearby in semantic space**"

We build this space automatically by seeing which words are **nearby in text**



Text Semantics

We define meaning of a word as a vector.

Called an "embedding" because it's embedded into a space.

The standard way to represent meaning in NLP:

Every modern NLP algorithm uses embeddings as the representation of word meaning.

Text Semantics

Why do we need embeddings compared to word features?

With **words**, a feature is a word identity

Feature number 729: "terrible"

Requires **exact same word** to be in training and test

With **embeddings**:

Feature is a word vector

The previous word was vector [35,22,17]

Now in the test set we might see a similar vector [34,21,14]

We can generalize to **similar but unseen** words!

Text Semantics

Bag-of-Words (e.g. tf-idf (embedding) vector)

A common baseline model from Information Retrieval

Sparse vectors

Words are represented by (a simple function of) the **counts** of nearby words

Word embeddings (e.g. Word2vec embedding vector)

Dense vectors

Representation is created by training a classifier to **predict** whether a word is likely to appear nearby

Later we'll discuss extensions called **contextual embeddings**

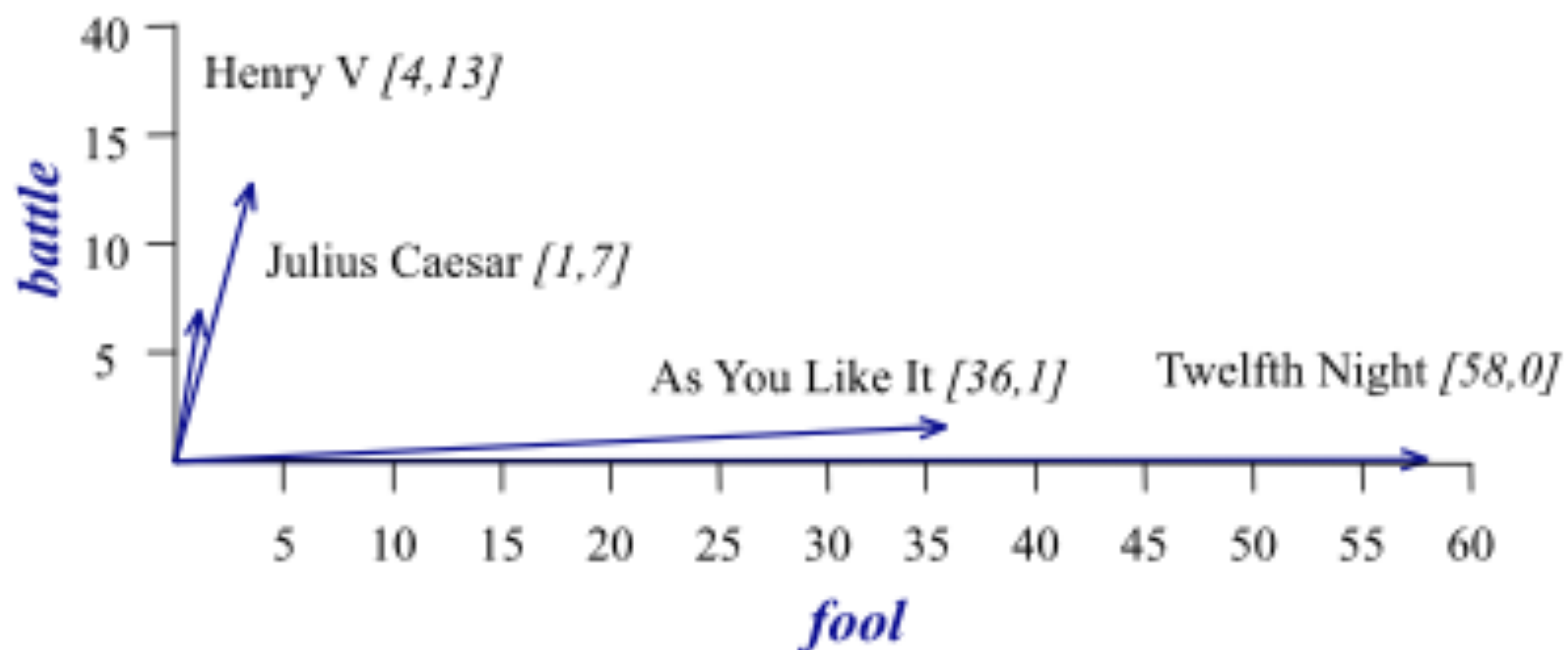
Text Semantics

Term-document matrix

Each document is represented by a vector of words

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Text Semantics



Text Semantics

Vectors are similar for the two comedies.

But comedies are different than the other two:
Comedies have more *fools* and *wit* and fewer *battles*.

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	14	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Text Semantics

Idea for word meaning: Words can be vectors too

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

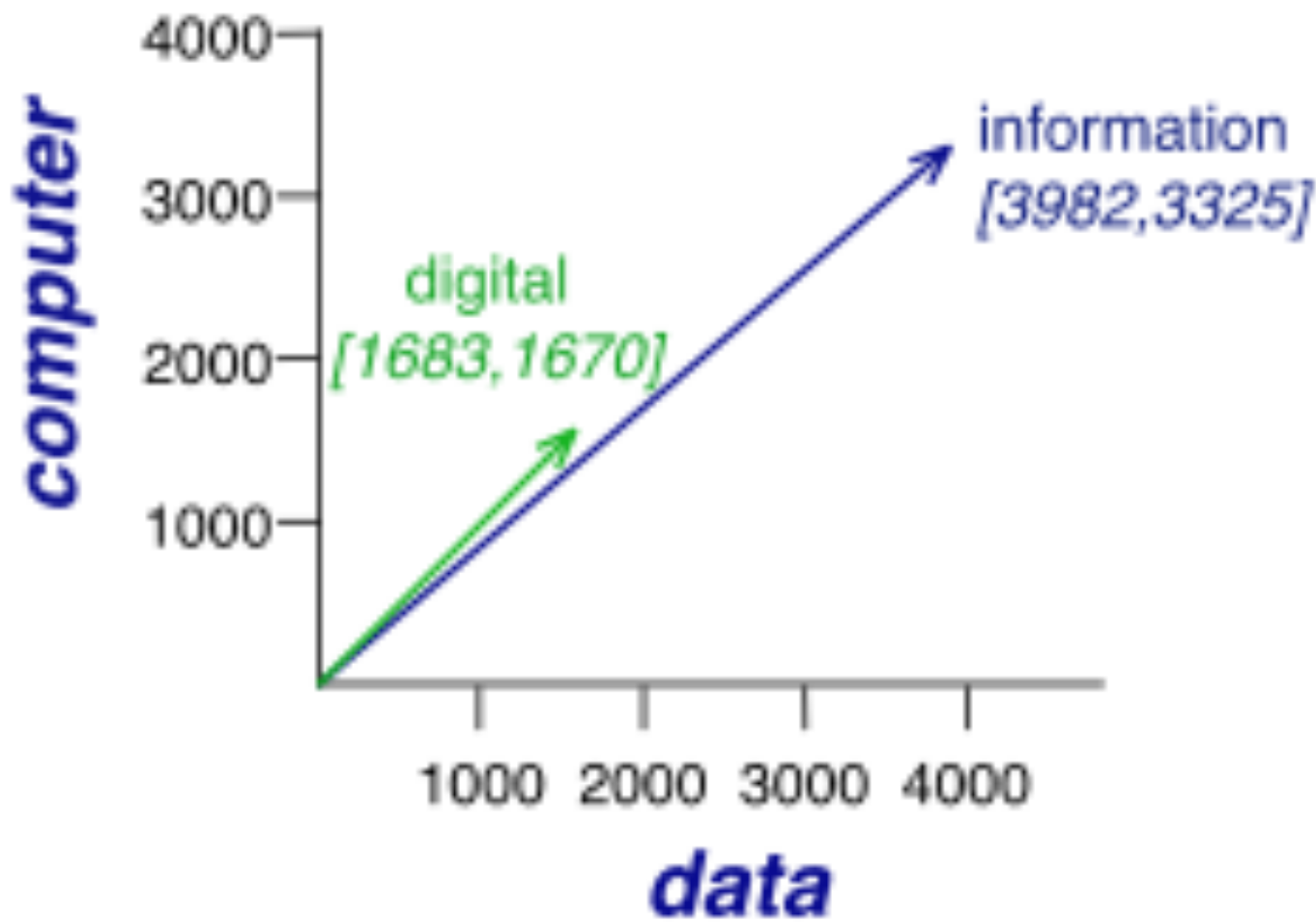
battle is "the kind of word that occurs in Julius Caesar and Henry V"

fool is "the kind of word that occurs in comedies, especially Twelfth Night"

Text Semantics

Two **words** are similar in meaning if their context vectors are similar.

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...



Text Semantics

Similarity calculation between vectors:

$$\text{dot product}(\mathbf{v}, \mathbf{w}) = \mathbf{v} \cdot \mathbf{w} = \sum_{i=1}^N v_i w_i = v_1 w_1 + v_2 w_2 + \dots + v_N w_N$$

The dot product tends to be high when the two vectors have large values in the same dimensions.

But Dot product favors long vectors: Dot product is higher if a vector is longer (has higher values in many dimension)

Frequent words (of, the, you) have long vectors (since they occur many times with other words).

So dot product overly favors frequent words

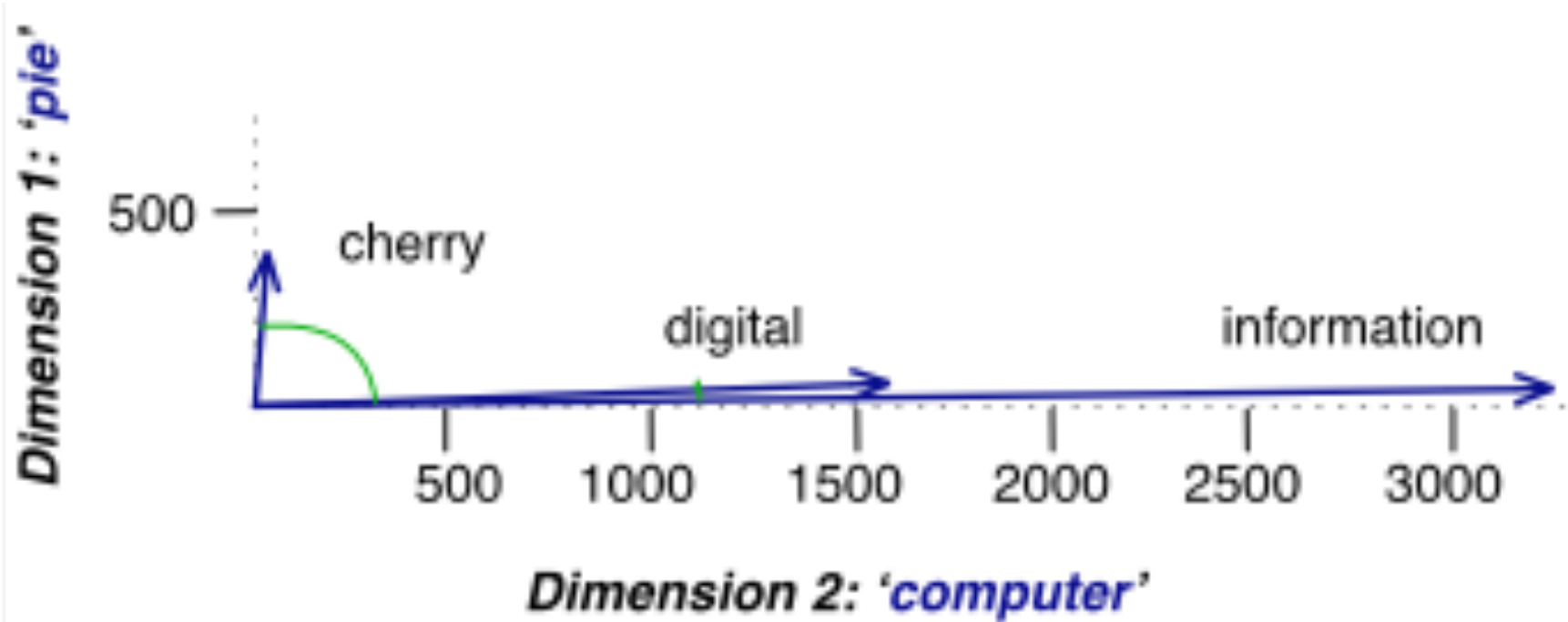
Text Semantics

Similarity calculation between vectors:

$$\text{cosine}(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}| |\vec{w}|} = \frac{\sum_{i=1}^N v_i w_i}{\sqrt{\sum_{i=1}^N v_i^2} \sqrt{\sum_{i=1}^N w_i^2}}$$

- 1: vectors point in opposite directions
- +1: vectors point in same directions
- 0: vectors are orthogonal

	pie	data	computer
cherry	442	8	2
digital	5	1683	1670
information	5	3982	3325



Text Semantics

Frequency is useful: If *sugar* appears a lot near *apricot*, that's useful information.

But most frequent words like *the*, *it*, or *they* are not very informative.

How can we balance these two conflicting constraints?

Text Semantics

Two common solutions for word weighting

tf-idf: tf-idf value for word t in document d :

$$w_{t,d} = \text{tf}_{t,d} \times \text{idf}_t$$

Words like "the" or "it" have very low idf

PMI: (Pointwise mutual information)

$$\text{PMI}(w_1, w_2) = \log \frac{p(w_1, w_2)}{p(w_1)p(w_2)}$$

If words like "good" appear more often with "great" than we would expect by chance

Text Semantics

$$w_{t,d} = \text{tf}_{t,d} \times \text{idf}_t$$

$$\text{tf}_{t,d} = \begin{cases} 1 + \log_{10} \text{count}(t,d) & \text{if } \text{count}(t,d) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{idf}_t = \log_{10} \left(\frac{N}{\text{df}_t} \right)$$

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

Text Semantics

Example of Cosine Similarity:

$$d_1 = 3 \ 2 \ 0 \ 5 \ 0 \ 0 \ 0 \ 2 \ 0 \ 0$$

$$d_2 = 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 2$$

$$d_1 \bullet d_2 = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5$$

$$\|d_1\| = (3*3 + 2*2 + 0*0 + 5*5 + 0*0 + 0*0 + 0*0 + 2*2 + 0*0 + 0*0)^{0.5} = (42)^{0.5} = 6.481$$

$$\|d_2\| = (1*1 + 0*0 + 0*0 + 0*0 + 0*0 + 0*0 + 0*0 + 1*1 + 0*0 + 2*2)^{0.5} = (6)^{0.5} = 2.245$$

$$\cos(d_1, d_2) = 0.3150$$

Text Semantics

Other similarity metrics (Set Similarity)

Jaccard Coefficient:

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}$$

1: existence of a word

0: absence of a word

Example:

sim(dx, dy)?

d1 = “Saturn is the gas planet with rings.”

d2 = “Jupiter is the largest gas planet.”

d3 = “Saturn is the Roman god of sowing.”

	Saturn	is	the	gas	planet	with	rings	Jupiter	largest	Roman	god	of	sowing
d1	1	1	1	1	1	1	1	0	0	0	0	0	0
d2	0	1	1	1	1	0	0	1	1	0	0	0	0
d3	1	1	1	0	0	0	0	0	0	1	1	1	1

Text Semantics

d1 = “Saturn is the gas planet with rings.”

d2 = “Jupiter is the largest gas planet.”

d3 = “Saturn is the Roman god of sowing.”

	Saturn	is	the	gas	planet	with	rings	Jupiter	largest	Roman	god	of	sowing
d1	1	1	1	1	1	1	1	0	0	0	0	0	0
d2	0	1	1	1	1	0	0	1	1	0	0	0	0
d3	1	1	1	0	0	0	0	0	0	1	1	1	1

Jaccard vs. Cosine?

	Saturn	is	the	gas	planet	with	rings	Jupiter	largest	Roman	god	of	sowing
d1	0.03	0	0	0.03	0.03	0.07	0.07	0	0	0	0	0	0
d2	0	0	0	0.03	0.03	0	0	0.08	0.08	0	0	0	0
d3	0.03	0	0	0	0	0	0	0	0	0.07	0.07	0.07	0.07

Text Classification

Given a dataset (labeled text documents according to classes),
Train a model to detect the class of unseen text documents,
And evaluate the model performance!

Supervised or Unsupervised?

Examples:

Sentiment Analysis in Movie Reviews

Spam Detection in E-mails

Hate Speech Detection on Social Media

Disinformation Detection on Social Media

Text Classification

Common Approach:

Represent text semantics with word/sentence/paragraph/document vectors

Apply a Machine Learning algorithm such as Naive Bayesian and SVM
or Deep Learning algorithm such as RNN and BERT

Bottlenecks?

Text Semantics

Sparse vs. dense vectors

tf-idf (or PMI) vectors

long (length $|V| = 10,000$ to $50,000$)

sparse (most elements are zero)

Alternative: learn vectors

short (length 50-1000)

dense (most elements are non-zero)

Text Semantics

Why dense vectors?

Short vectors may be easier to use as **features** in deep learning (fewer weights to tune)

Dense vectors may **generalize** better than explicit counts

Dense vectors may do better at capturing synonymy:

car and *automobile* are synonyms; but are distinct dimensions

In practice, they work better

Text Semantics

Drawbacks of Bag-of-Words Model

Loss of Context and Meaning

“The car chased the mouse.”

“The mouse chased the cat.”

“He is happy.”

“He is not happy.”

Text Semantics

Drawbacks of Bag-of-Words Model

High Dimensionality and Sparsity

A dataset of 10,000 unique words would require a 10,000-dimensional vector.

Text Semantics

Drawbacks of Bag-of-Words Model

Out-of-Vocabulary Words

Out-of-Vocabulary words are typically either ignored or mapped to a special "unknown" token, which can lead to a loss of information.

Text Semantics

Drawbacks of Bag-of-Words Model

Lack of Semantic Similarity

"excellent" and "great" are treated as two entirely different features, even though they are synonyms and convey a similar positive sentiment.

Text Semantics

Drawbacks of Bag-of-Words Model

Stopwords

Common words like "the," "a," and "is" (known as stop words) often appear with high frequency in texts.



ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

Thanks for your participation!

Çağrı Toraman
28.10.2025