

Biểu diễn ngữ nghĩa của từ

Vector Semantic và Word embedding

Nội dung chính

1. Nghĩa của từ
2. Vector ngữ nghĩa / word embedding
3. TF-IDF
4. Word2Vec
5. Một số đặc điểm khác của Embedding
6. Hiện thực word embedding đơn giản.

Một số thuật ngữ quan trọng

- Word: từ.
- Lemma: Gốc từ.
- Sense: mặt nghĩa.
- Definition: Định nghĩa.

Words, Lemmas, Senses, Definitions

Mặt nghĩa

lemma

sense

definition

pepper, *n.*

Pronunciation: B.Mt. /ˈpepə/, U.S. /ˈpepər/

Forms: OE **peopor** (*rare*), OE **pipecer** (transmission error), OE **pipor**, OE **pipur** (*rare*)
Frequency (in current use):

Etymology: A borrowing from Latin. **Etymon:** Latin *piper*.
< classical Latin *piper*, a loanword < Indo-Aryan (as is ancient Greek *πῖπερι*); compare Sai

I. The spice or the plant.

1.

a. A hot pungent spice derived from the prepared fruits (peppercorns) of the pepper plant, *Piper nigrum* (see sense 2a), used from early times to season food, either whole or ground to powder (often in association with salt). Also (locally, chiefly with distinguishing word): a similar spice derived from the fruits of certain other species of the genus *Piper*; the fruits themselves.

The ground spice from *Piper nigrum* comes in two forms, the more pungent *black pepper*, produced from black peppercorns, and the milder *white pepper*, produced from white peppercorns: see **BLACK adj.** and **n.** Special uses 5a, **PEPPERCORN n.** 1a, and **WHITE adj.** and **n.**¹ Special uses 7b(a).

2.

a. The plant *Piper nigrum* (family Piperaceae), a climbing shrub indigenous to South Asia and also cultivated elsewhere in the tropics, which has alternate stalked entire leaves, with pendulous spikes of small green flowers opposite the leaves, succeeded by small berries turning red when ripe. Also more widely: any plant of the genus *Piper* or the family Piperaceae.

b. Usu. with distinguishing word: any of numerous plants of other families having hot pungent fruits or leaves which resemble pepper (1a) in taste and in some cases are used as a substitute for it.

c. U.S. The California pepper tree, *Schinus molle*. Cf. **PEPPER TREE n.** 3.

3. Any of various forms of capsicum, esp. *Capsicum annuum* var. *annuum*. Originally (chiefly with distinguishing word): any variety of the *C. annuum* Longum group, with elongated fruits having a hot, pungent taste, the source of cayenne, chilli powder, paprika, etc., or of the perennial *C. frutescens*, the source of Tabasco sauce. Now frequently (more fully **sweet pepper**): any variety of the *C. annuum* Grossum group, with large, bell-shaped or apple-shaped, mild-flavoured fruits, usually ripening to red, orange, or yellow and eaten raw in salads or cooked as a vegetable. Also: the fruit of any of these capsicums.

Sweet peppers are often used in their green immature state (more fully **green pepper**), but some new varieties remain green when ripe.

Gốc từ: pepper

- Sense 1: spice from pepper plant
- Sense 2: the pepper plant itself
- Sense 3: another similar plant (Jamaican pepper)
- Sense 4: another plant with peppercorns (California pepper)
- Sense 5: *capsicum* (i.e. chili, paprika, bell pepper, etc)

A **sense** or “**concept**” is the meaning component of a word

Gốc từ

- Một từ có thể mang **nhiều mặt nghĩa** khác nhau:

VD: từ “mouse” (con chuột)

mouse (N)

1. any of numerous small rodents... (động vật)
2. a hand-operated device that controls a cursor... (thiết bị máy tính).

lemma (gốc từ) là một “hình thái” nhằm biểu đạt một khái niệm hay sự vật trong thực tế. “mouse” được gọi là lemma – nhằm thể hiện cho ý nghĩa về con chuột. Ngoài “mouse”, “mice” cũng có ý nghĩa tương tự

➔ Để xác định được chính xác nghĩa của từ “mouse”, cần dựa vào ngữ cảnh (context) cụ thể.

Bài toán khử nhập nhằng nghĩa của từ (word sense disambiguation – WSD).

Các loại quan hệ ngữ nghĩa giữa các từ

- Synonyms (tạm dịch: đồng nghĩa)
- Word similarity (tạm dịch: tương đồng)
- Word relatedness (tạm dịch: liên quan)
- Connotation (tạm dịch: nghĩa mở rộng)

Synonyms

- Các từ khác nhau, nhưng mang chung một ý nghĩa (cùng đề cập đến một khái niệm hay ý nghĩa) được gọi là synonym. Hay nói cách khác, 2 từ được gọi là synonyms nếu như chúng có thể hoán đổi cho nhau mà không thay đổi ý nghĩa của câu.

VD:

Couch / sofa

Car / automobile

Word similarity

- Hai từ khác nhau, nếu cùng đề cập đến một khía cạnh ý nghĩa cụ thể, thì được gọi là similarity.

VD: cats và dogs (cùng đề cập đến khía cạnh về động vật)

- Để đánh giá được similarity, vẫn phải có sự can thiệp của con người. Con người sẽ đánh một trọng số xác định cho thấy tính “similarity” giữa 2 từ với nhau.

VD: Bộ SimLex-99 (Hill et al.)

vanish	disappear	9.8
behave	obey	7.3
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

Word relatedness

- Hai từ được gọi là “liên quan (relatedness)” với nhau, nếu như nó cùng thuộc một lĩnh vực ngữ nghĩa (semantic fields) nhất định:
 - Semantic fields: là tập hợp các từ cùng đề cập đến một vùng ý nghĩa cụ thể, và có cấu trúc liên quan với nhau

VD: nếu cùng đề cập về lĩnh vực “machine learning”, thì các từ sau được gọi là *relatedness* với nhau: SVM, deep learning, transformers, AdaGrad.

Semantic Frames and Roles

- **Semantic frames (khung ngữ nghĩa)**: đề cập đến một đối tượng chủ thể trong câu nói.
- **Roles (vai trò)**: Chức năng của chủ thể trong câu nói

VD: → đang đề cập đến ngữ nghĩa về “mua bán” (buy)

*Sam bought the book from **Ling***

Ling** sold the book to **Sam

Sam và Ling là 2 chủ thể

Chức năng của Sam và Ling là khác nhau:

- Sam: người bán
- Ling: người mua

Connotations

- Là ngữ nghĩa của một từ liên quan tới cảm xúc, quan điểm, ý kiến cá nhân của người đọc, người viết.

VD: cảm xúc tích cực (happy), cảm xúc tiêu cực (sad). Đánh giá tích cực (great, love), đánh giá tiêu cực (terrible, hate).

- Các khía cạnh liên quan về mặt cảm xúc (tính theo mức độ cảm xúc tích cực) (Osgood et al., 1957):
 - **valence** (tạm dịch: mức độ)
 - **arousal** (tạm dịch: cường độ)
 - **dominance** (tạm dịch: uy lực).

VD:

happy sẽ có valence cao, annoyed sẽ có mức độ thấp.

excited sẽ có arousal cao, calm sẽ có arousal thấp.

important sẽ có dominance cao, *influenced* sẽ có dominance thấp.

	Valence	Arousal	Dominance
courageous	8.05	5.5	7.38
music	7.67	5.57	6.5
heartbreak	2.45	5.65	3.58
cub	6.71	3.95	4.24
life	6.68	5.59	5.89

Vector ngữ nghĩa

- Mục đích: biểu diễn **ngữ nghĩa** của từ.

Định nghĩa về *ngữ nghĩa* của từ:

The meaning of a word **is its use** in the language (Ludwig Wittgenstein)

➔ Như vậy, ý nghĩa của một từ được thể hiện qua **ngữ cảnh xung quanh nó**.

➔ Ngữ cảnh xung quanh ở đây chính là các **từ xung quanh**.

Ví dụ về ý nghĩa của từ

- **Ong Choi** is delicious **sautéed with garlic**.
- **Ong Choi** is superb **over rice**
- **Ong Choi** **leaves** with **salty sauces**
- ...spinach **sautéed with garlic over rice**
- Chard stems and **leaves** are **delicious**
- Collard greens and other **salty** leafy greens

Kết luận ý nghĩa của từ Ong Choi dựa vào tần suất xuất hiện của các từ xung quanh nó như: *spinach* , *chard* , *collard greens*,

Kết luận: Ongchoi is a **leafy green like spinach, chard, or collard greens**

Thế Ong Choi thực ra là gì ?

空心菜
kangkong
rau muống
...



Spinach, Chard, Collard



Spinach (rau chân vịt)



Collard (cải búp)



Chard (Rau lá xanh Thụy Sĩ)

Word embedding

- Biểu diễn ngữ nghĩa 1 từ (**word**) bằng một **vector**.
 - Khái niệm embedding có nghĩa là “nhúng” từ vào không gian vector.
- Đây là kỹ thuật rất cơ bản để biểu diễn ngữ nghĩa của 1 từ thông qua ngữ cảnh xung quanh nó.
- Khái niệm **embedding** ở đây dùng để chỉ **vector** biểu diễn cho các từ.

Two-dimensional (t-SNE) projection of embeddings

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



Word và vector

- Co-occurrence matrix (tạm dịch: Ma trận đồng hiện) là công cụ dùng để biểu diễn vector ngữ nghĩa của một từ thông qua tần suất xuất hiện của từ đó trong một **ngữ cảnh cụ thể**.
- Có 2 dạng Co-occurent matrix thường gặp:
 - *Ma trận term-document.*
 - *Ma trận word-word.*

Biểu diễn bằng ma trận term-documents

- Mỗi dòng biểu diễn cho 1 từ trong tập từ vựng.
- Mỗi cột biểu diễn cho một document (tạm dịch là: văn bản).
- Mỗi ô thể hiện số lần xuất hiện của một từ đang xét trong một văn bản cụ thể.

Ví dụ

- Có 4 văn bản:
 - D1: As you like it (W. Shakespeare).
 - D2: Twelfth Night (W. Shakespeare).
 - D3: Julius Caesar.
 - D4: Henry V.
- Ghi chú: D1 và D2 là 2 tác phẩm hài kịch, còn D3 và D4 là 2 tác phẩm lịch sử.

Term-document matrix

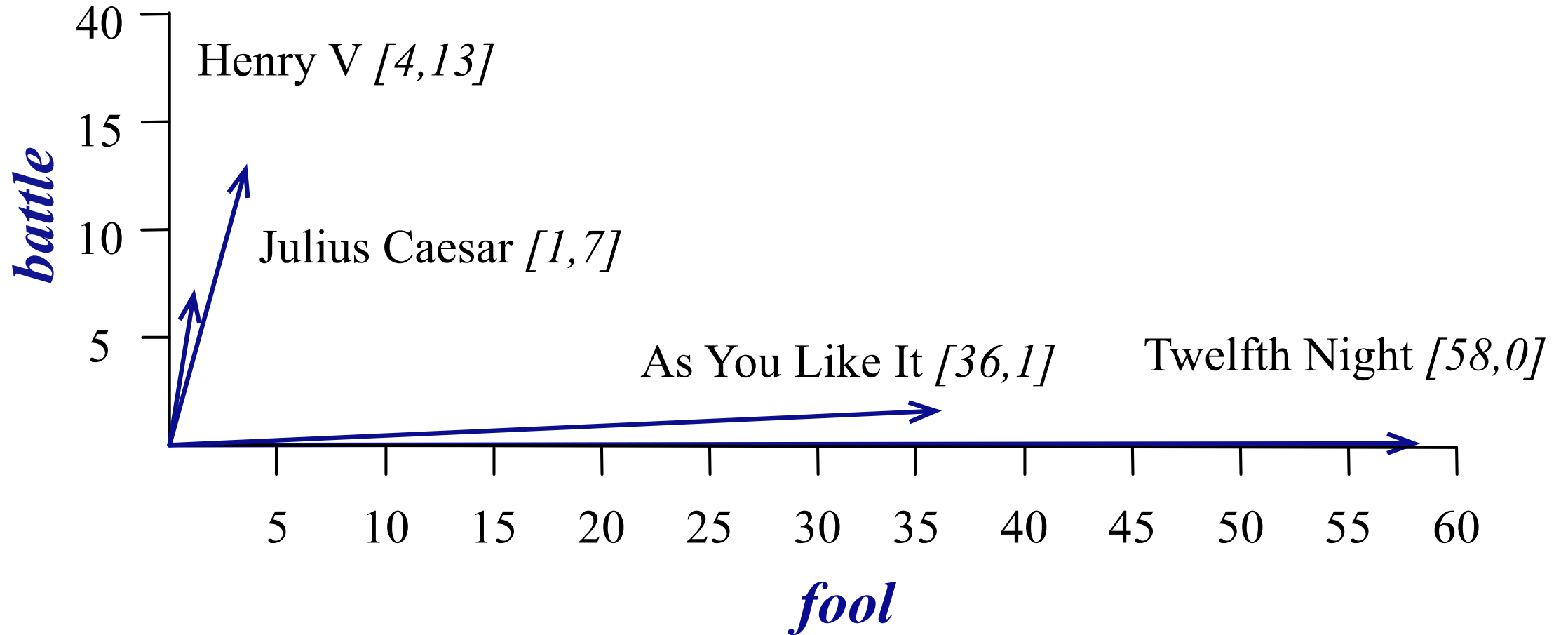
	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Mỗi văn bản được biểu diễn thành dạng vector từ.

Nhận xét

- Trong các tác phẩm hài kịch của Shakespeare, chữ “fool” xuất hiện rất nhiều:
 - As you like it xuất hiện 36 lần.
 - Twelfth night xuất hiện 58 lần.
 - Ở các tác phẩm về lịch sử, chữ “good” xuất hiện nhiều lần:
 - Julius Caesar: 62 lần.
 - Henry V.: 89 lần.
- ➔ hai văn bản được xem là “tương tự” nhau nếu vector biểu diễn từ của 2 văn bản tương tự nhau.

Visualizing document vectors



Word-word co-occurrence matrix

- Mỗi dòng biểu diễn cho 1 từ đích (target) trong tập từ vựng.
- Mỗi cột sẽ biểu diễn cho cho một từ trong tập từ vựng có cùng ngữ cảnh (context) với từ đích.
- Mỗi ô thể hiện giá trị xuất hiện của 2 từ trên trong cùng 1 ngữ cảnh nhất định.

Ví dụ

- Các từ: cherry, strawberry, digital và information là các từ đích.
- Ngữ cảnh xung quanh của các từ đích là 4 từ lần lượt bên trái, và bên phải của từ đích.

is traditionally followed by **cherry** pie, a traditional dessert
often mixed, such as **strawberry** rhubarb pie. Apple pie
computer peripherals and personal **digital** assistants. These devices usually
a computer. This includes **information** available on the internet

Word-word co-occurrence matrix

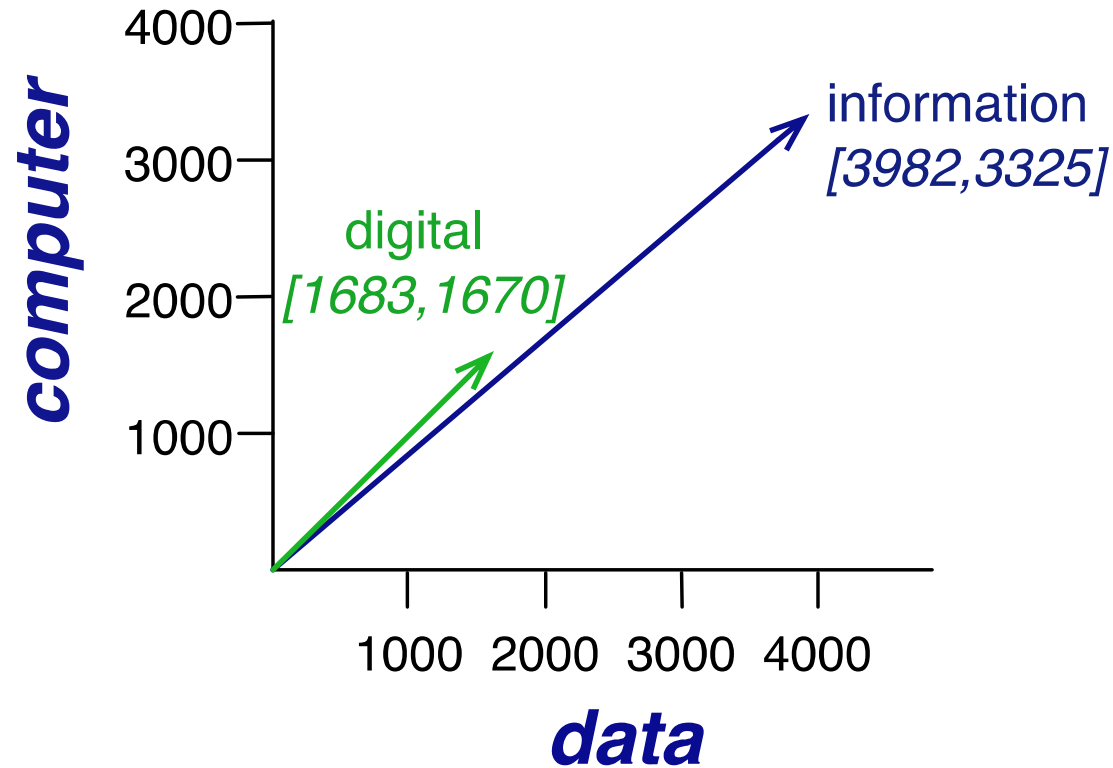
Ma trận Co-occurent của 4 từ: cherry, strawberry, digital và information trong bộ Wikipedia

	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	...

Ví dụ về sự xuất hiện của từ pie (context) xung quanh từ digital. Giả định ngữ cảnh là 4 từ xung quanh từ digital

- I love a **digital** *pie*.....
- What's **digital** is often a *pie*.....
- May I have some **digital** *pie*....
- **Digital** world necessitates *pie* eating
- There's something **digital** about this *pie*

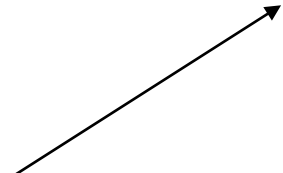

Visualizing words vectors



***Vấn đề: Định lượng sự tương đồng giữa 2 vector bằng cách nào ?
→ Độ đo tương đồng.***

Cosine similarity

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

chiều dài của vector  *dot product !!
(Tích vô hướng)* 

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

Một số lưu ý

- Giá trị cosine = -1 \rightarrow 2 vector ngược nhau.
- Giá trị cosine = 1 \rightarrow 2 vector giống nhau.
- Giá trị cosine = 0 \rightarrow 2 vector trực giao.

Cosine examples

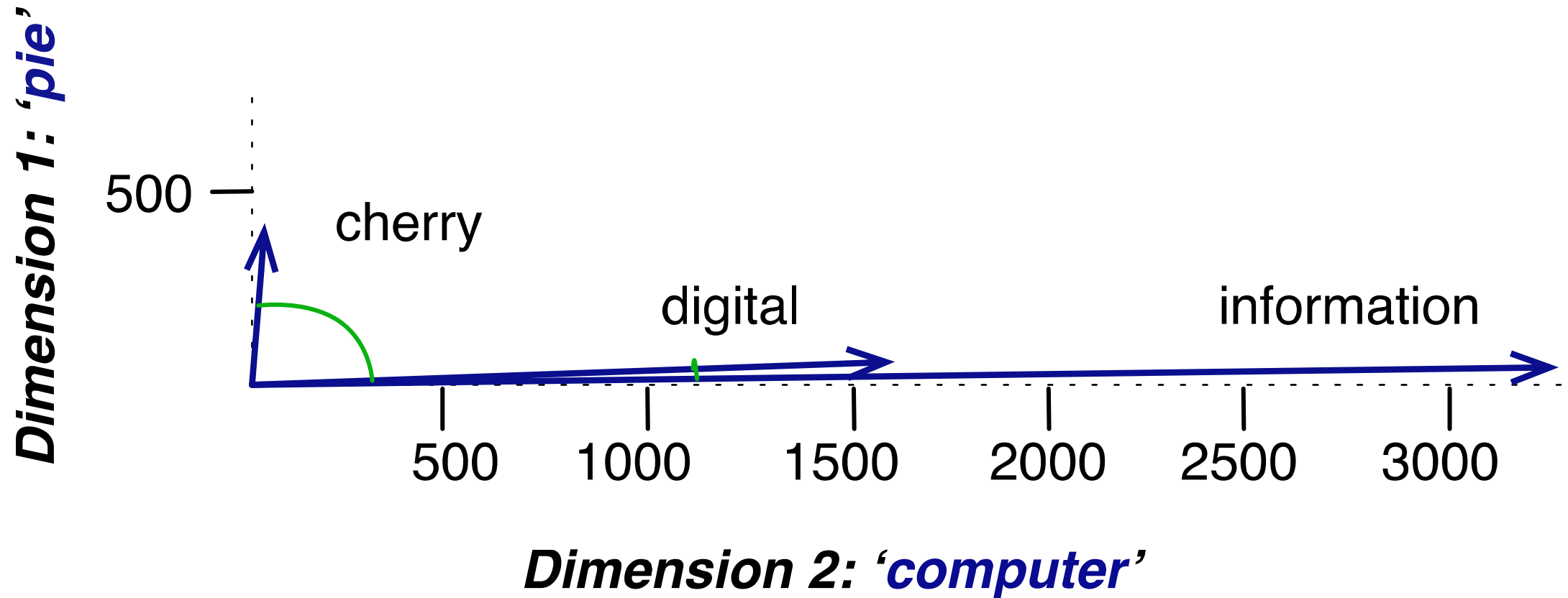
$$\cos(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}| |\vec{w}|} = \frac{\vec{v}}{|\vec{v}|} \cdot \frac{\vec{w}}{|\vec{w}|} = \frac{\sum_{i=1}^N v_i w_i}{\sqrt{\sum_{i=1}^N v_i^2} \sqrt{\sum_{i=1}^N w_i^2}}$$

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

$$\cos(\text{cherry}, \text{information}) = \frac{442 * 5 + 8 * 3982 + 2 * 3325}{\sqrt{442^2 + 8^2 + 2^2} \sqrt{5^2 + 3982^2 + 3325^2}} = .017$$

$$\cos(\text{digital}, \text{information}) = \frac{5 * 5 + 1683 * 3982 + 1670 * 3325}{\sqrt{5^2 + 1683^2 + 1670^2} \sqrt{5^2 + 3982^2 + 3325^2}} = .996$$

Visualizing cosines (well, angles)



Vấn đề với cosine

- Cosine dựa trên phép tính dot product.
 - Các đặc điểm của dot product:
 - *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).*
- ➔ Như vậy, sự xuất hiện của các từ như: *of, the* sẽ cho kết quả dot product cao ➔ giá trị cosine cao.
- Các từ như *of, the* có thực sự mang nhiều ý nghĩa ??

Các kỹ thuật nhúng từ

tf-idf

- Kỹ thuật xác định trọng số của một từ dựa trên việc đếm tần suất xuất hiện của nó.
- Dễ sử dụng.
- **Sparse vector.**

word2vec

- Dự đoán khả năng xuất hiện của 1 từ dựa vào các từ trước nó.
- Cần phải huấn luyện mô hình.
- **Dense vector.**

Các phương pháp đánh trọng số cho từ

Thay vì sử dụng phương pháp đếm tần suất xuất hiện, ta sử dụng phương pháp đánh trọng số. Có 2 phương pháp như sau:

1. **TF-IDF (Salton, 1971)**
2. Pointwise mutual information (PMI).

*(Đọc thêm Chương **Vector Semantics and Embeddings** trong sách **Speech and Language Processing**)*

TF-IDF

- Gồm 2 thành phần:
 - **Term frequency**: Đo lường độ phổ biến của 1 từ t bằng cách đếm tần suất xuất hiện của từ t .

$$tf_{t,d} = \log_{10}(\text{count}(t,d)+1)$$

- **Inverse document frequency**: Đo lường mức **độ hiếm** của một từ t trong tập dữ liệu.

$$idf_t = \log_{10} \left(\frac{N}{df_t} \right)$$

N: Tổng số văn bản.
 df_t : số văn bản chứa từ t .

- Nhân lại với nhau:

$$w_{t,d} = tf_{t,d} \times idf_t$$

Ví dụ:

- Trong một vở kịch của Shakespeare, chữ “**Romeo**” và “**action**” được đếm tần suất như sau:

	Collection Frequency	Document Frequency
Romeo	113	1
action	113	31

Ví dụ (tt)

- Bảng sau trình bày tần suất xuất hiện của một số từ trong toàn bộ 37 vở kịch của Shakespeare.
 - Chữ “sweet” và “good” xuất hiện trong toàn bộ 37 vở kịch. Đây là một trong các từ được nhà văn Shakespeare thích dùng (Jurafsky, 2014, p. 175).

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

Đánh trọng số cho “Romeo” và “action”

- TF-IDF (“**Romeo**”) = $\text{count}(t,d) \times \text{idf}(t,d) = \log_{10}(113) \times \log_{10}(37/1) = 3.21$
- TF-IDF (“action”) = $\text{count}(t,d) \times \text{idf}(t,d) = \log_{10}(113) \times \log_{10}(37/31) = 0.15$

	Collection Frequency	Document Frequency
Romeo	113	1
action	113	31

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

Tương tự - tính TF-IDF cho các từ khác

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

TF

	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

TF-IDF

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	0.074	0	0.22	0.28
good	0	0	0	0
fool	0.019	0.021	0.0036	0.0083
wit	0.049	0.044	0.018	0.022

IDF

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

Tập văn bản D và tập từ vựng V

- D1: Data Base System Concepts
- D2: Introduction to Algorithms
- D3: Computational Geometry: Algorithms and Applications
- D4: Data Structures and Algorithm Analysis on Massive Data Sets
- D5: Computer Organization

Xét tập từ vựng (8 từ):

$V = \{$

w1= data,

w2= system,

w3= algorithm,

w4=computer,

w5= geometry,

w6= structures,

w7= analysis,

w8= organization

$\}$

Yêu cầu

- Sử dụng công thức TF-IDF để tính trọng số của các từ trên trong tập văn bản D.
- Thể hiện quan hệ giữa các văn bản thông qua 2 từ: **computer** và **algorithm**

Tính TF – ma trận term-document

	D1	D2	D3	D4	D5
data	2	1	1	3	1
system	2	1	1	1	1
algorithm	1	2	2	2	1
computer	1	1	2	1	2
geometry	1	1	2	1	1
structure	1	1	1	2	1
analysis	1	1	1	2	1
organization	1	1	1	1	2

Tính IDF

data: xuất hiện trong văn bản *D1*, *D4*

$$\Rightarrow df = 2, idf = \log_{10}(5/2) = 0.39$$

system: xuất hiện trong văn bản *D1*

$$\Rightarrow df = 1, idf = \log_{10}(5/1) = 0.69$$

algorithm: xuất hiện trong văn bản *D2*, *D3*, *D4*

$$\Rightarrow df = 3, idf = \log_{10}(5/3) = 0.22$$

computer: xuất hiện trong văn bản *D5*

$$\Rightarrow df = 1, idf = \log_{10}(5/1) = 0.69$$

geometry: Xuất hiện trong văn bản *D3*

$$\Rightarrow df = 1, idf = \log_{10}(5/1) = 0.69$$

structure: Xuất hiện trong văn bản *D1*

$$\Rightarrow df = 1, idf = \log_{10}(5/1) = 0.69$$

analysis: Xuất hiện trong văn bản *D4*

$$\Rightarrow df = 1, idf = \log_{10}(5/1) = 0.69$$

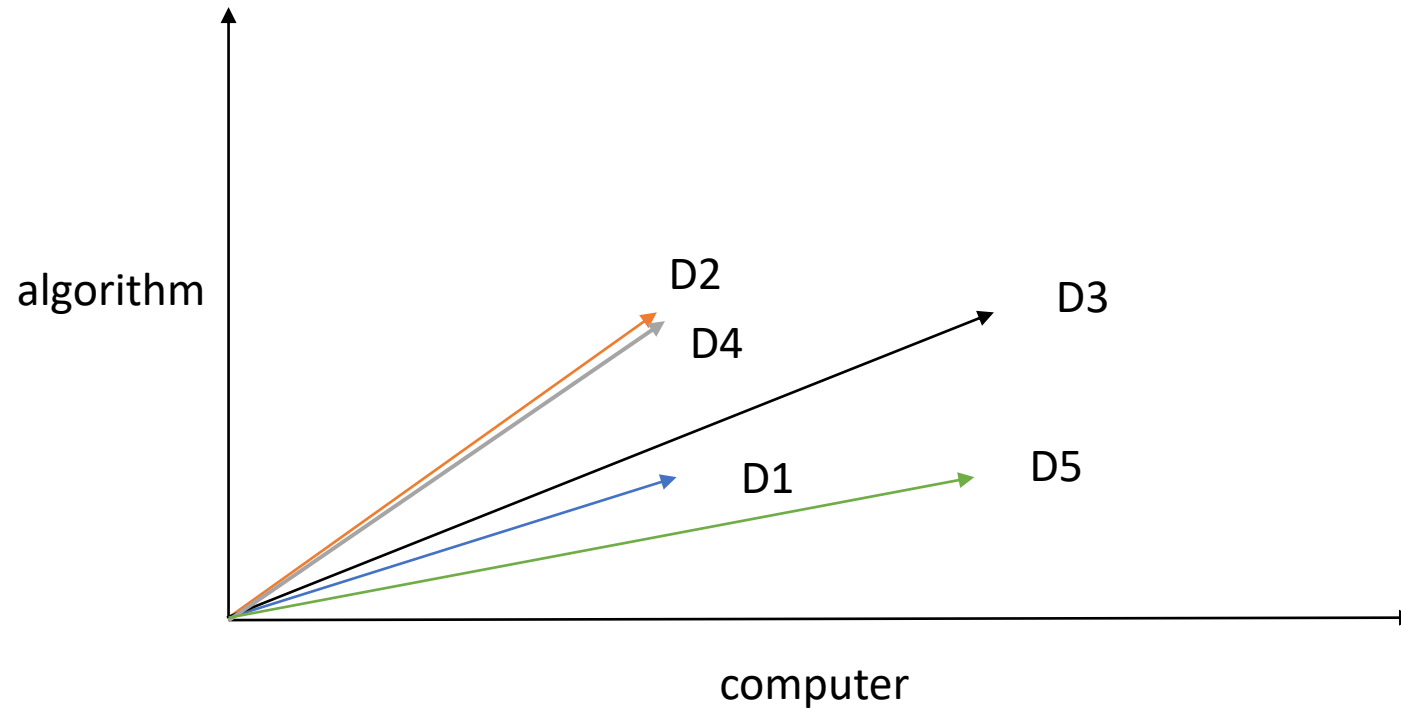
organization: xuất hiện trong văn bản *D5*

$$\Rightarrow df = 1, idf = \log_{10}(5/1) = 0.69$$

Tính TF-IDF – ma trận term-document

	D1	D2	D3	D4	D5
data	0.78	0.39	0.39	<i>1.17</i>	0.39
system	0.78	0.69	0.69	0.69	0.69
algorithm	0.22	0.44	0.44	0.44	0.22
computer	0.69	0.69	1.38	0.69	<i>1.38</i>
geometry	0.69	0.69	1.38	0.69	0.69
structure	0.69	0.69	0.69	1.38	0.69
analysis	0.69	0.69	0.69	1.38	0.69
organization	0.69	0.69	0.69	0.69	<i>1.38</i>

Thể hiện quan hệ giữa các văn bản thông qua 2 từ: **computer** và **algorithm**



D1 = [0.69, 0.22]

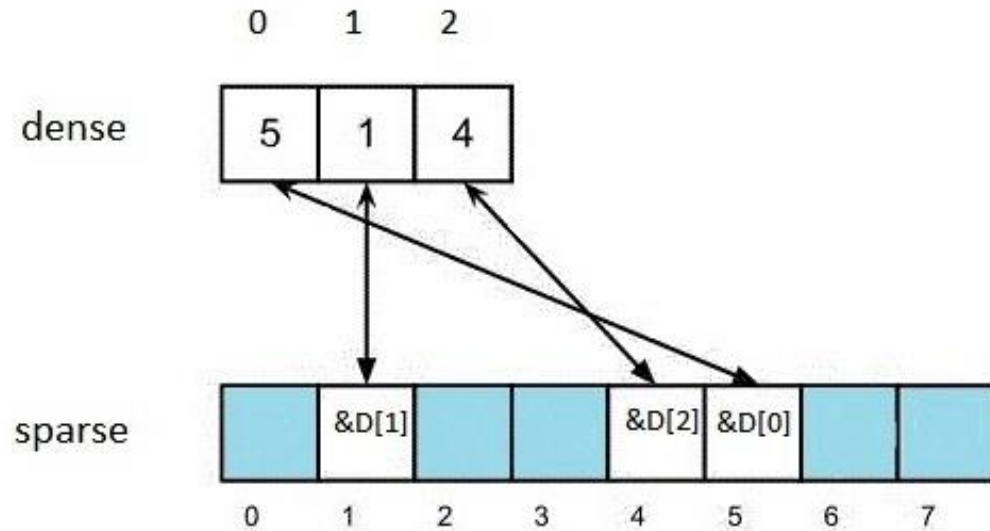
D2 = [0.69, 0.44]

D3 = [1.38, 0.44]

D4 = [0.69, 0.44]

D5 = [1.38, 0.22]

Sparse vector vs Dense vector.



- tf-idf vectors are
 - **long** (length $|V| = 20,000$ to $50,000$)
 - **sparse** (most elements are zero)
- Alternative: **learn vectors** which are
 - **short** (length 50-1000)
 - **dense** (most elements are non-zero)

Vì sao dense vector tốt hơn

- Ít trọng số học hơn => dễ tinh chỉnh tham số khi huấn luyện.
- Có tính khái quát nhiều hơn.
- Dễ bắt được tính synonymy của từ:
 - *car* và *automobile* là 2 từ đồng nghĩa (synonyms); nhưng sẽ có chiều (dimension) khác nhau.

Các kỹ thuật biến đổi từ sparse thành dense vector

- “Neural Language Model”-inspired models
 - **Word2vec** (skipgram, CBOW), Glove.
- Singular Value Decomposition (SVD)
- Contextual Embeddings (ELMo, BERT)

Word2VEC

- Ý tưởng chính: **predict** rather than **count** .
- Thay vì đếm số lần xuất hiện của từ w bất kỳ xung quanh từ “*apricot*”, Word2VEC sẽ **dự đoán** xem **xác suất** từ w xuất hiện khi biết từ *apricot* là bao nhiêu?



Tomáš Mikolov

Ý tưởng

... lemon, a [tablespoon of **apricot** jam, a] pinch ...

Input: Cho bộ (w, c)

với w là từ **đích**, c là từ xuất hiện khi biết từ w (ngữ cảnh).

Output: Xác suất c là **ngữ cảnh thực sự** của w . Ký hiệu là:

$$P(+|w, c).$$

Xác suất để c không là ngữ cảnh thực sự của w :

$$P(-|w, c) = 1 - P(+|w, c).$$

VD: Cho 2 bộ: (apricot, jam) và (apricot, lemon)

➔ Tính xác suất để *jam* là ngữ cảnh thực sự của w thay vì *lemon*.

➔ $P(+| \text{apricot, jam}) = ?$

Tính xác suất P

- Dựa vào **embedding similarity**: “a word is likely to occur near the target if its embedding is similar to the target embedding”.
→ tính similarity giữa 2 vector của 2 từ: w (từ đích) và c (từ ngữ cảnh).

$$\text{Sim}(\mathbf{c}, \mathbf{w}) \approx \mathbf{c} \cdot \mathbf{w}$$

Sử dụng hàm **sigmoid**: $\sigma(x) = \frac{1}{1+e^{-x}}$

Thế vào P, ta được: $P(+|\mathbf{w}, \mathbf{c}) = \frac{1}{1+e^{(-\mathbf{c} \cdot \mathbf{w})}}$

Suy ra: $P(-|\mathbf{w}, \mathbf{c}) = \frac{1}{1+e^{(\mathbf{c} \cdot \mathbf{w})}}$

Skip-gram

- Dự đoán xác suất các từ c_1, c_2, \dots, c_L là ngữ cảnh thực sự của w .
- Giả định rằng các từ ngữ cảnh c_1, c_2, \dots, c_L **độc lập** nhau.

$$P(+|w, c_{1:L}) = \prod_{i=1}^L \sigma(c_i \cdot w)$$

$$\log P(+|w, c_{1:L}) = \sum_{i=1}^L \log \sigma(c_i \cdot w)$$

Skip-gram with negative example

- Nhắc lại ý tưởng về Word2VEC: **dự đoán** xem **xác suất** từ w xuất hiện khi biết từ *apricot* là bao nhiêu?
 - Huấn luyện một mô hình trả lời cho câu hỏi: từ c có phải là từ ngữ cảnh của từ *apricot* hay không → mô hình nhị phân.
- Để huấn luyện mô hình nhị phân, cần cả **positive** và **negative** example.
- Để tạo ra negative example → Lấy ngẫu nhiên k từ trong tập từ vựng không phải là ngữ cảnh của từ w .

Skip-gram with negative sampling

...lemon, a [tablespoon of apricot jam, a] pinch...

c1 c2 [target] c3 c4

positive examples +

t	c
apricot	tablespoon
apricot	of
apricot	jam
apricot	a

negative examples -

t	c	t	c
apricot	aardvark	apricot	seven
apricot	my	apricot	forever
apricot	where	apricot	dear
apricot	coaxial	apricot	if

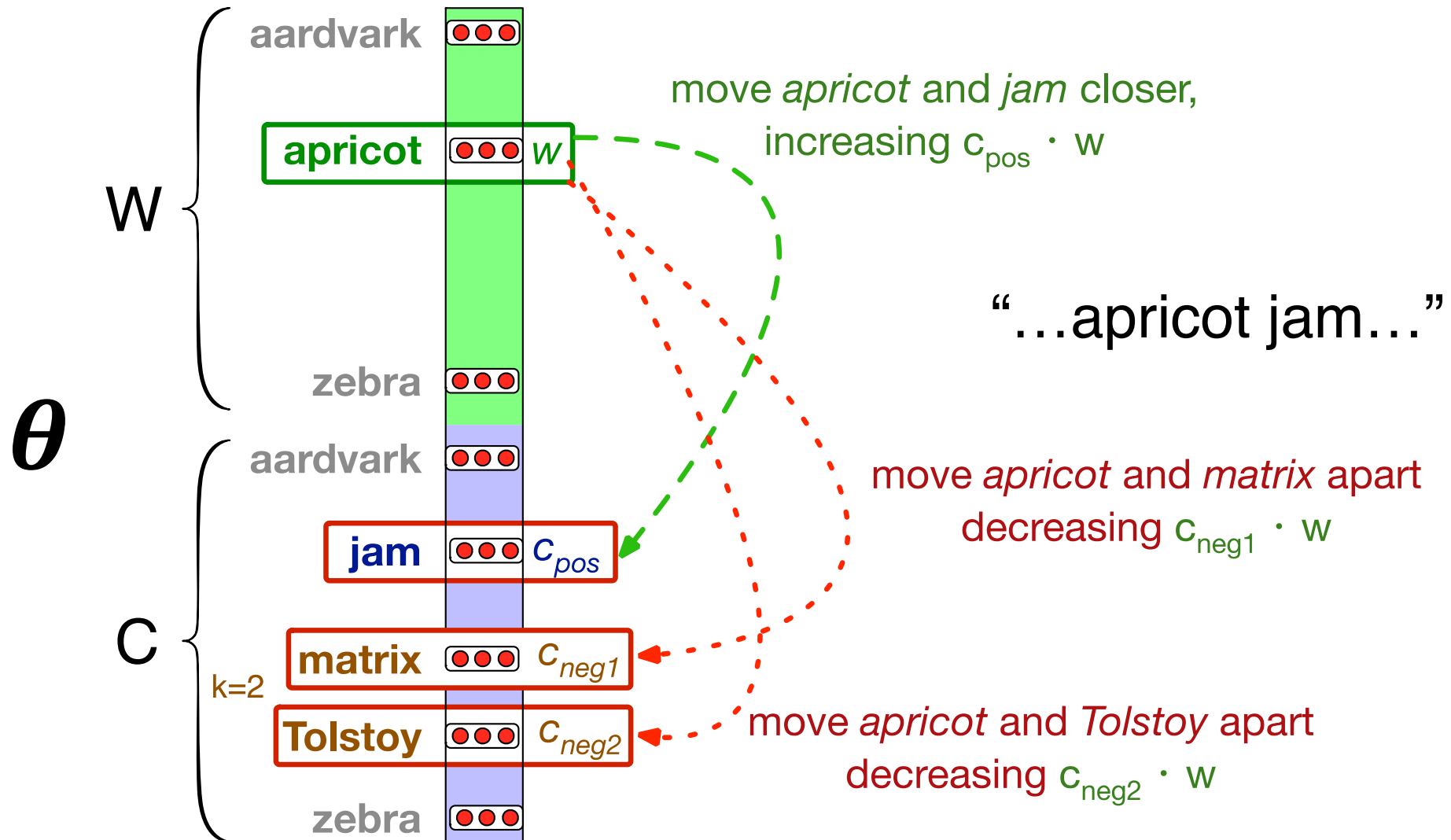
Huấn luyện skip-gram with negative sampling

- Cho bộ dữ liệu huấn luyện ban đầu gồm: positive và negative training.
- Khởi tạo tập các embedding ban đầu.
- Mục tiêu huấn luyện:
 - **Maximize** the *similarity* of the *target word*, *context word* pairs (w, c_{pos}) drawn from the positive data.
 - **Minimize** the similarity of the (w, c_{neg}) pairs drawn from the negative data.
- Hàm mất mát (loss function):

k là số lượng negative example.

$$L_{CE} = - \left[\log \sigma(c_{pos} \cdot w) + \sum_{i=1}^k \log \sigma(-c_{neg_i} \cdot w) \right]$$

Intuition of one step of gradient descent



Mục tiêu huấn luyện

- Tối ưu hàm mất mát (loss function):

$$L_{CE} = - \left[\log \sigma(c_{pos} \cdot w) + \sum_{i=1}^k \log \sigma(-c_{neg_i} \cdot w) \right]$$

- Công cụ sử dụng: gradient descent!!

Tính đạo hàm của hàm mất mát

$$L_{CE} = - \left[\log \sigma(c_{pos} \cdot w) + \sum_{i=1}^k \log \sigma(-c_{neg_i} \cdot w) \right]$$

$$\frac{\partial L_{CE}}{\partial c_{pos}} = [\sigma(c_{pos} \cdot w) - 1]w$$

$$\frac{\partial L_{CE}}{\partial c_{neg}} = [\sigma(c_{neg} \cdot w)]w$$

$$\frac{\partial L_{CE}}{\partial w} = [\sigma(c_{pos} \cdot w) - 1]c_{pos} + \sum_{i=1}^k [\sigma(c_{neg_i} \cdot w)]c_{neg_i}$$

Cập nhật trọng số cho w theo GD

Start with randomly initialized C and W matrices, then incrementally do updates

$$c_{pos}^{t+1} = c_{pos}^t - \eta [\sigma(c_{pos}^t \cdot w) - 1] w$$

$$c_{neg}^{t+1} = c_{neg}^t - \eta [\sigma(c_{neg}^t \cdot w)] w$$

$$w^{t+1} = w^t - \eta [\sigma(c_{pos} \cdot w^t) - 1] c_{pos} + \sum_{i=1}^k [\sigma(c_{neg_i} \cdot w^t)] c_{neg_i}$$

Summary: How to **learn word2vec** (skip-gram) embeddings

- Start with V random d -dimensional vectors as initial embeddings
- Train a classifier based on embedding similarity
 - Take a corpus and take pairs of words that co-occur as positive examples
 - Take pairs of words that don't co-occur as negative examples
 - Train the classifier to distinguish these by slowly adjusting all the embeddings to improve the classifier performance
 - Throw away the classifier code and keep the embeddings.

Các đặc điểm của embedding

- Đặc điểm 1: Different types of similarity or association.
- Đặc điểm 2: Analogy/Relational Similarity.

Đặc điểm 1: The kinds of neighbors depend on window size

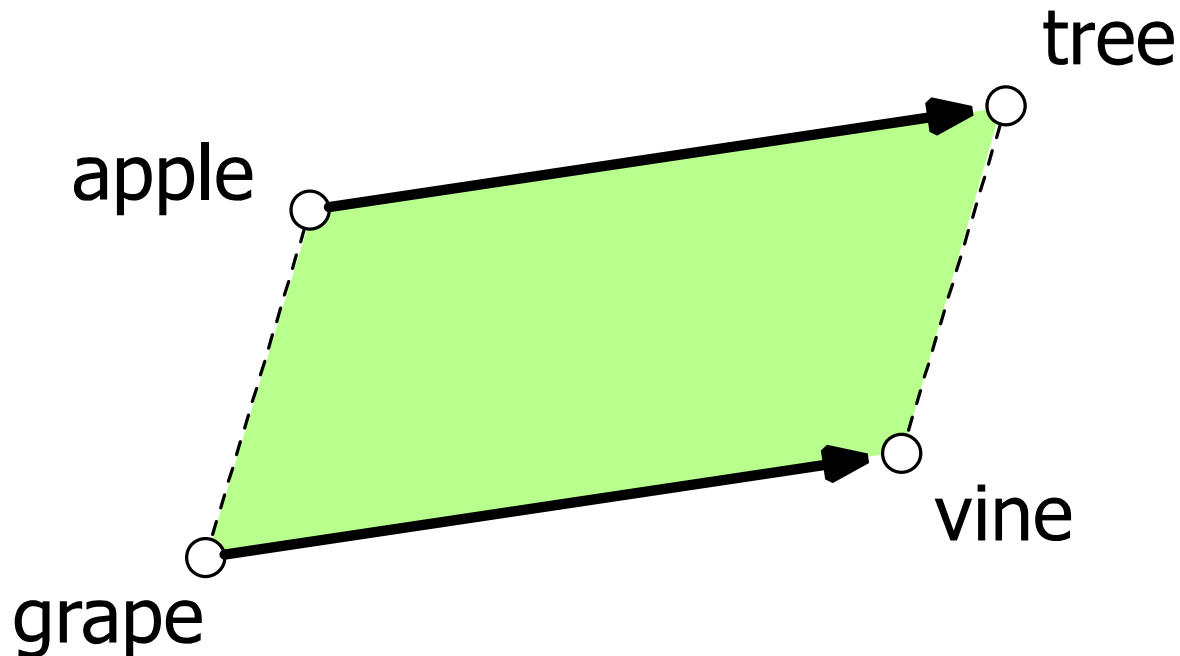
- **Large windows** ($C = +/- 5$) : nearest words are **related words** in same semantic field
 - *Hogwarts* nearest neighbors are Harry Potter world:
 - *Dumbledore, Half-blood, Malfoy*
- **Small windows** ($C = +/- 2$) : nearest words are **similar nouns, words** in same taxonomy
 - *Hogwarts* nearest neighbors are other fictional schools
 - *Sunnydale, Evernight, Blandings*

Đặc điểm 2: Analogical relations

The classic parallelogram model of analogical reasoning (Rumelhart and Abrahamson 1973)

To solve: "*apple is to tree as grape is to _____*"

Add $\overrightarrow{\text{apple} - \text{tree}}$ to $\overrightarrow{\text{grape}}$ to get ***vine***



Analogical relations via parallelogram

The parallelogram method can solve analogies with both sparse and dense embeddings (Turney and Littman 2005, Mikolov et al. 2013b)

$\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}}$ is close to $\overrightarrow{\text{queen}}$

$\overrightarrow{\text{Paris}} - \overrightarrow{\text{France}} + \overrightarrow{\text{Italy}}$ is close to $\overrightarrow{\text{Rome}}$

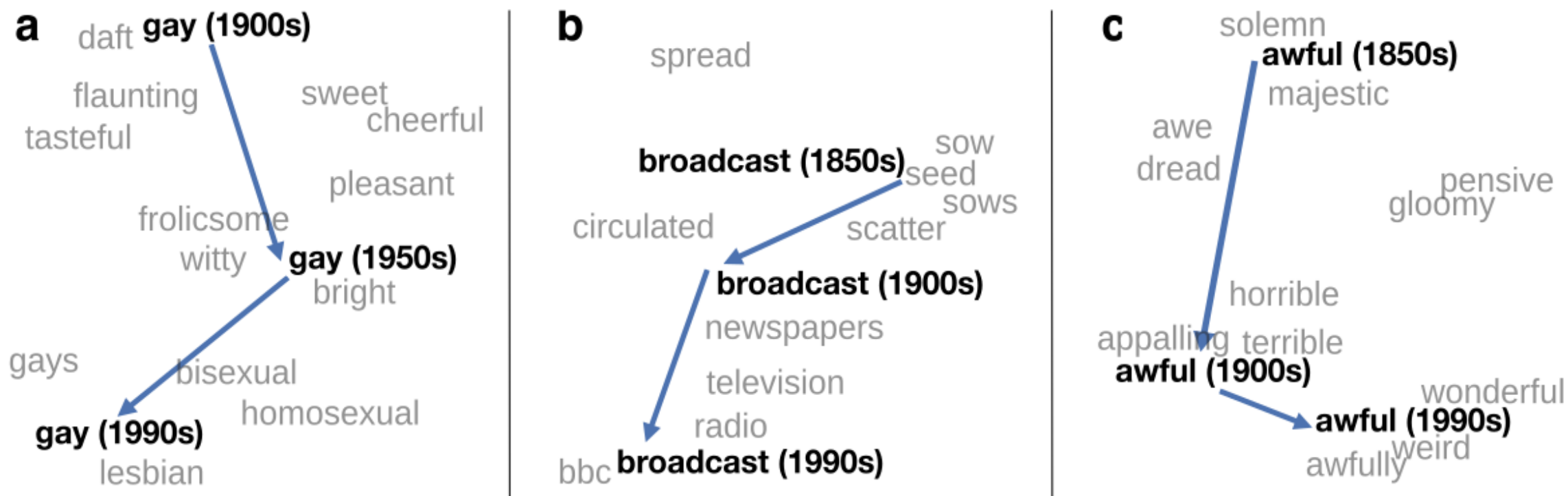
For a problem $a:a^*:b:b^*$, the parallelogram method is:

$$\hat{b}^* = \operatorname{argmax}_x \operatorname{distance}(x, a^* - a + b)$$

Hiện tượng Bias trong embedding

Train embeddings on different decades of historical text to see meanings shift

~30 million books, 1850-1990, Google Books data



William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. Proceedings of ACL.

Tổng kết

- Trong word embedding, một từ sẽ được biểu diễn dưới dạng một vector – gọi là vector nhúng từ (word embedding). Mỗi vector sẽ có số chiều cố định.
- Có 2 dạng biểu diễn bằng vector: vector thưa (sparse) và vector dày (dense).
 - Biểu diễn sparse vector: TF-IDF.
 - Biểu diễn dense vector: Skip-grams.
- Huấn luyện classifier trong embedding: dự đoán từ có khả năng xuất hiện nhất với từ đích (target) dựa vào “ngữ cảnh”.
- Tính toán độ tương đồng của các vector: cosine và dot product.

Xây dựng word embedding đơn giản

Bộ dữ liệu huấn luyện

- Bộ dữ liệu **UIT-VSFC**.
- Công bố khoa học: K. V. Nguyen, V. D. Nguyen, P. X. V. Nguyen, T. T. H. Truong and N. L. Nguyen, "*UIT-VSFC: Vietnamese Students' Feedback Corpus for Sentiment Analysis*", KSE 2018.
- Số lượng dữ liệu: khoảng hơn 16,000 câu.
- Thư viện sử dụng: *gensim*

<https://pypi.org/project/gensim/>

Đọc dữ liệu

```
import pandas as pd

X_train = pd.read_csv('uit-vsfc/train/sents.txt',
                      sep='\n', header=None, index_col=None)
X_train = X_train.iloc[:, 0]
```

Tách từ

```
from pyvi import ViTokenizer

sentences = X_train.values
tokenized_sentences = []

for s in sentences:
    tokenized_sentences.append(
        ViTokenizer.tokenize(s).split()
    )
```

Huấn luyện mô hình Word2Vec

```
import multiprocessing
from gensim.models import Word2Vec

w2v_model = Word2Vec(min_count=20, window=2, size=300)
w2v_model.build_vocab(tokenized_sentences)

w2v_model.train(tokenized_sentences,
                total_examples=w2v_model.corpus_count,
                epochs=30, report_delay=1
)
```


Lưu lại bộ word embedding

- Lưu dạng txt:

```
w2v_model.wv.save_word2vec_format('w2v_vsfc.txt', binary=False)
```

- Lưu dạng binary:

```
w2v_model.wv.save_word2vec_format('w2v_vsfc.bin', binary=True)
```

Sử dụng

- Lấy ra vector của từ “giảng_dạy”:

```
print(w2v_model["giảng_dạy"])
```

- Analogical reasoning: “**giảng_dạy**” + “**đồ_án**” – “**kiểm_tra**” is to ____

```
w2v_model.wv.most_similar(  
    positive=["giảng_dạy", "đồ_án"],  
    negative=["kiểm_tra"],  
    topn=1  
)
```

Hãy thực hiện thử và cho biết kết quả nhé ☺