Natural Language Processing

Introduction

Outline

NLP Applications

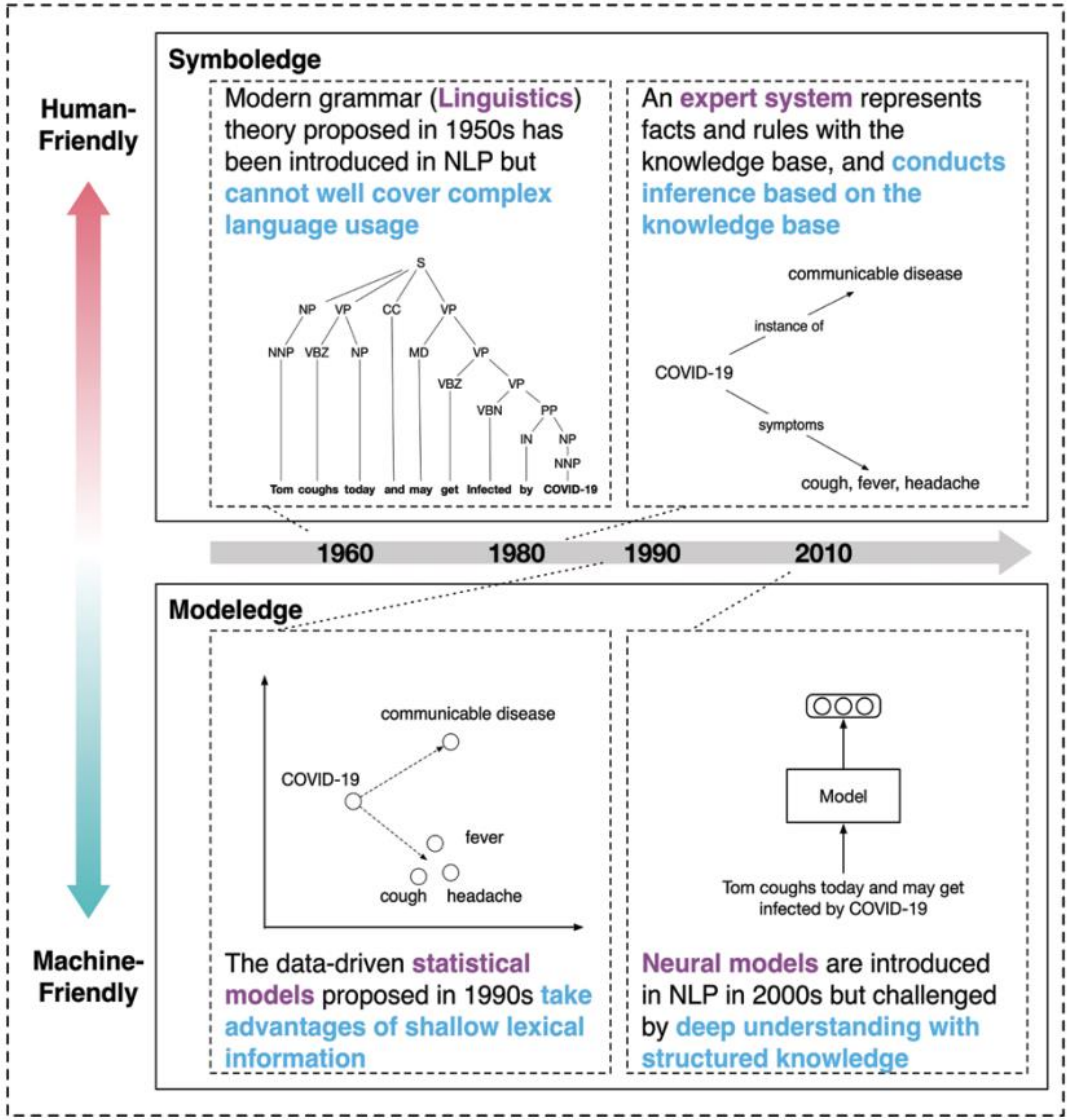
General Text Processing

- Indexing
- Lexical processing
- Document representation

Latent semantic analysis

Word Embedding

Knowledge for Language Understanding



"Knowledgeable Machine Learning for Natural Language Processing", H. Han, Communications of the ACM, 2021.

What is Natural Language Processing (NLP, 自然語言處理)?

Natural language processing (NLP) is the use of human languages by a computer and is viewed as a branch of machine learning.

Translation

Information Retrieval

Chatbot

Information Verification

An example

川普陣營評估，只要在新罕州、科羅拉多州和賓州這3州贏得任何1州，就可跨越270張的選舉人票當選門檻。(圖:美聯社檔案照)

在距離投票日僅剩5天，共和黨總統候選人川普的競選經理凱莉亞妮·康薇指出，在最後幾天，川普陣營將斥資上億美元大打廣告全力搶攻新罕州、科羅拉多州和賓州這3州，川普就差最後一哩路了。

康薇指出，目前，佛州、賓州、維吉尼亞州、俄亥俄州、北卡州、威斯康辛州、愛荷華州、緬因州、新罕州、科羅拉多州和內華達州等州的選情，正處於難分軒輊的膠著狀態。康薇表示，「在最後的一星期中，川普將全力搶攻這些州」。

競選經理康薇透露，在上個月的民調顯示，川普在愛荷華、俄亥俄、緬因、佛州、內華達和北卡州維持領先，如果此一領先局勢能持續下去，則川普的選舉人票將可達266票。她指出，「川普就只差4張選舉人票，即可贏得這場大選。」

康薇認為，川[illegible]來居上之勢。

州，就可跨越

康薇透露，川

美元在重要的

提早投票與投

(中時電子報)

關鍵字：選舉

工商時報財經：13檔低接籌碼守護 有靠山

美股打噴嚏全球股市重感冒，亞股本周無一收漲，台股雖相對抗跌，但短線波動大增，法人建議可留意近2日成交量不減反增、股價守穩短均線之上個股，包括國巨、立敦、智原等13檔，顯示有低接籌碼守護，攻守兼備。

中國時報財經：棄醫念資訊 百萬年薪上門

社會新鮮人面對未來，如何做好萬全準備？棄醫念資訊、今年將從台大資管系畢業的Jackie，目前還沒拿到畢業證書，就被國際知名企業網羅，成為年薪百萬的軟體工程師，Jackie表示，看好大陸惠台政策，未來有更多工作經驗後，也不排除西進發展的可能。

星島日報財經：曾跌見7.8315水位 港匯05年來新低

股市大跌，港美息差不斷擴闊，令港匯弱勢持續，昨日更曾跌穿7.83兌1美元，低見7.8315，創2005年5月金管局引入雙向兌換保證以來的新低，有市場人士估計，不排除有投機者或趁勢重操故技，一方面沽港匯夾息，另方面沽港股期指，從中獲利，港匯料有機會下試7.8350。雖然港匯連日偏弱，但金管局指，暫未有資金流出港元。

Indexing / Search

Classification

Event detection

News influence

Summarization

Article relationship

Sentiment

Unsupervised Training Corpus

How?

Splitting text into sentences.

Find the part-of-speech for words inside a sentence.

Determine different types of subclauses.

Determine the subject and the direct object of a sentence.

Easy for human, hard for computer

Word sense ambiguity

Watch for kids

"Watch for kids" 在繁体中文中的意思是「適合兒童觀看」或「兒童專用」。這個短語通常用來形容專為兒童設計或適合兒童觀看的內容或產品。它常用於描述兒童節目、兒童電影、兒童電視節目或其他針對兒童的娛樂產品。這意味著該內容或產品是專門為兒童群體製作的，通常會考慮到他們的年齡、興趣和認知能力，以確保內容對他們而言是安全、具有教育性或娛樂性的。

?

watch for kids 從Word sense ambiguity 來解釋它的意義

"Watch for kids" 這句話是一個典型的詞義歧義（Word Sense Ambiguity）的例子。詞義歧義是指單詞或詞組有多種可能的解釋，而具體含義取決於上下文。在這句話中，"watch" 和 "kids" 都有多重解釋，導致不同的可能含義。

1. **Watch** 的歧義：
   - **動詞意思：留意、監視**：在這種情況下，"watch" 的意思是密切注意某事或某人。例如："Watch for kids" 可以解釋為 "小心孩子" 或 "注意有小孩" 的意思，通常用於提醒人們留意周圍環境，以防止孩子突然跑出來或造成危險。
   - **名詞意思：手錶**：如果把 "watch" 理解為名詞，那麼這句話可能會被誤解為 "給孩子的手錶"，這顯然不是它的正確含義。
2. **Kids** 的歧義：
   - **孩子（小孩）**：這是最常見的意思，指的是年輕的人或小孩子。
   - **年輕的動物（例如小山羊）**：在某些上下文中，"kids" 也可以指年幼的動物，但這在日常對話中並不常見。

在 "Watch for kids" 這句話中，最合乎常理的解釋是將 "watch" 理解為動詞 "留意" 或 "注意"，而 "kids" 是指小孩子。因此，這句話大多用來提醒他人注意附近是否有小孩在活動，以防可能的危險情況。

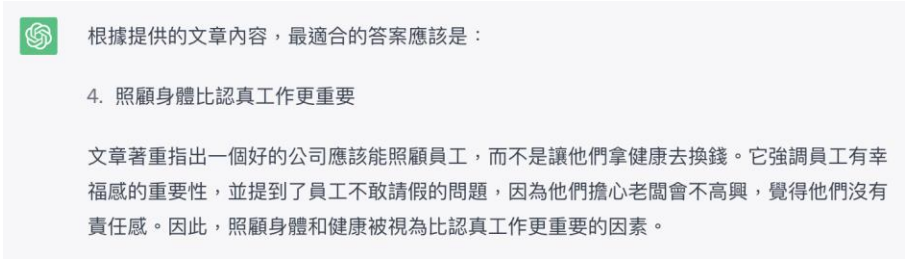
How teach computer to understand this?

Q：曾有一項調查發現，很多員工生病的時候不敢請假，因為他們擔心老闆會不高興，覺得他們沒有責任感。有人認為，員工會這麼想是公司的責任。一個好的公司應該能照顧員工，而不是讓他們拿健康去換錢。因此，讓員工有幸福感，應該是未來企業努力的方向。

這篇文章說了什麼內容？

科技大擂台, 2017

1. 老闆應該給員工多一點兒假
2. 常關心別人的人更有責任感
3. 對公司有意見要勇敢說出來
4. 照顧身體比認真工作更重要



English is difficult

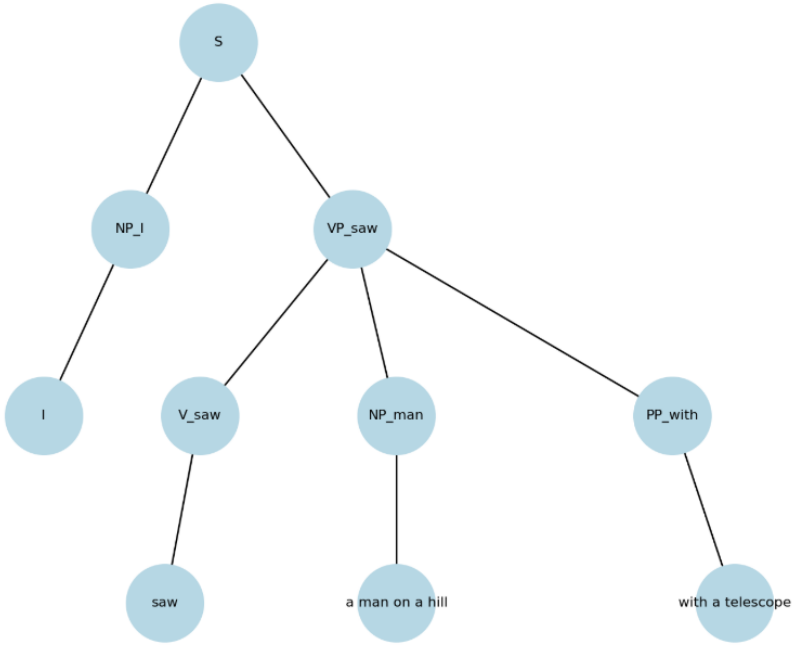
I saw a man on a hill with a telescope.　我用望遠鏡在山上看到一個男人

- There's a man on a hill, and I'm watching him with my telescope.
  (山上有一個人，我正用望遠鏡觀察他。)
- There's a man on a hill, who I'm seeing, and he has a telescope.
  (我看到山上有一個人，他有一架望遠鏡。)
- There's a man, and he's on a hill that also has a telescope on it.
  (有一個男人，他在一座山上，山上也有一架望遠鏡。)
- I'm on a hill, and I saw a man using a telescope.
  (我在一座山上，看到一個男人正在使用望遠鏡。)
- There's a man on a hill, and I'm sawing him with a telescope.
  (山上有一個男人，我用望遠鏡觀察他。)

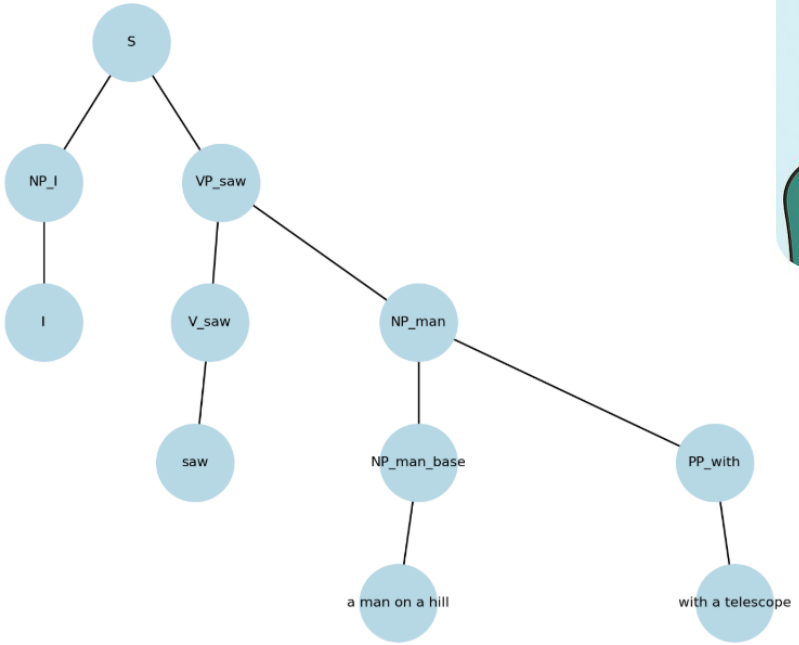
```
(ROOT
 (S
  (NP (PRP I))
  (VP (VBD saw)
   (NP (DT a) (NN man))
   (PP (IN on)
    (NP (DT a) (NN hill)))
   (PP (IN with)
    (NP (DT a) (NN telescope))))
  (. .)))
```

syntactic ambiguity

Interpretation 1: I used a telescope



Interpretation 2: The man has a telescope



I saw a man on a hill with a telescope.

Interpretation 1:
I used a telescope

Interpretattion 2:
The man has a telescope

What is NLP?

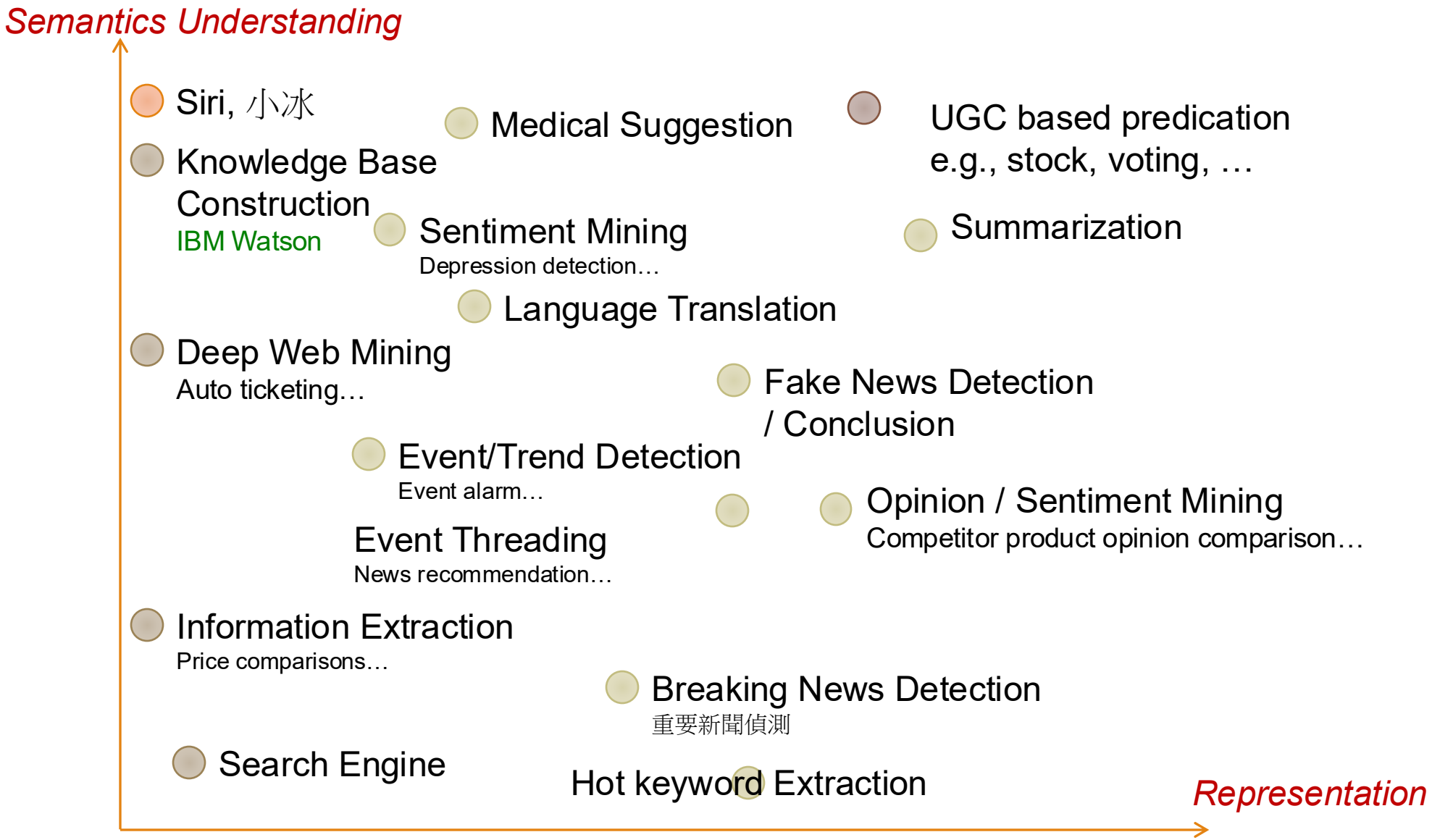
Natural language processing is a field at the intersection of

- computer science
- artificial intelligence
- and linguistics.

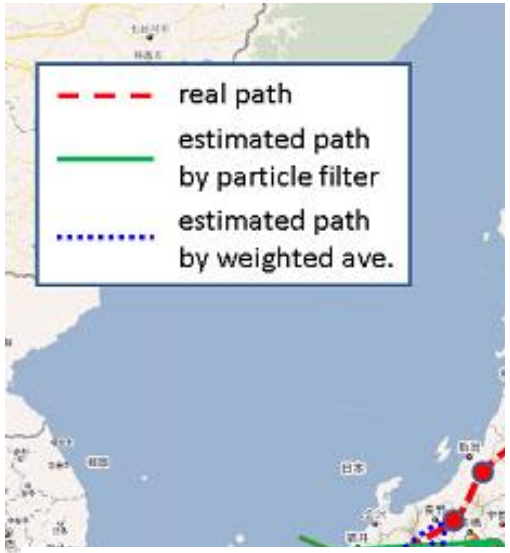
Fully understanding and representing the meaning of language (or even defining it) is a difficult goal.

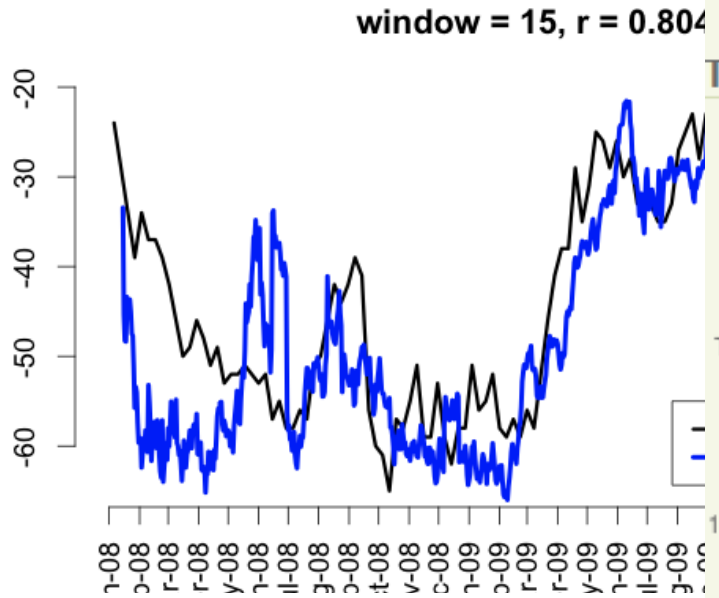
Perfect language understanding is AI-complete. (Christopher Manning)

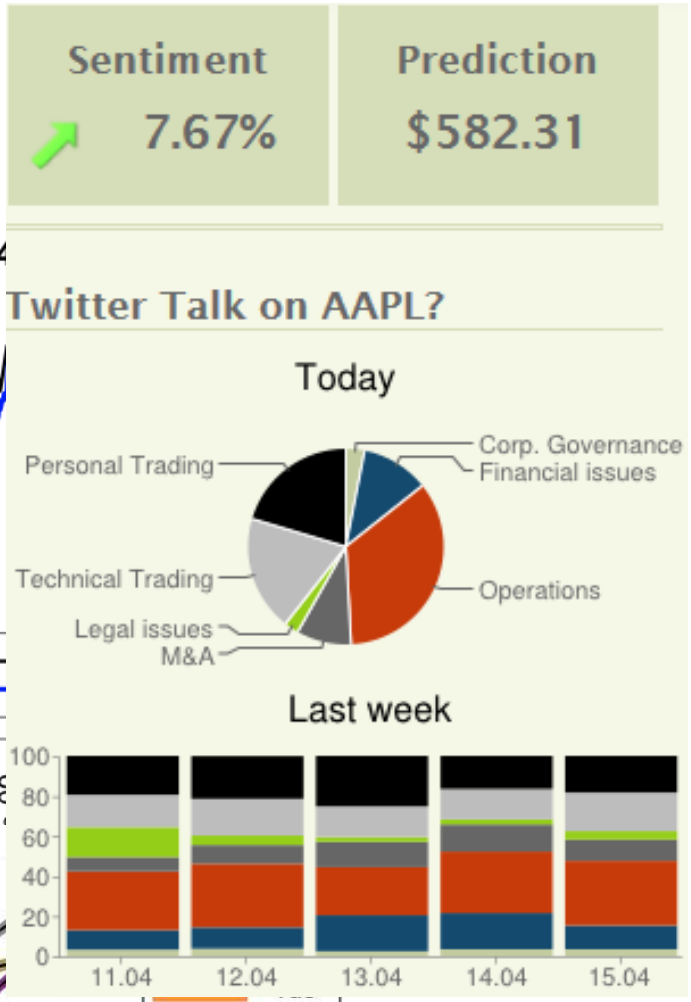
NLP Based Applications (before LLM)

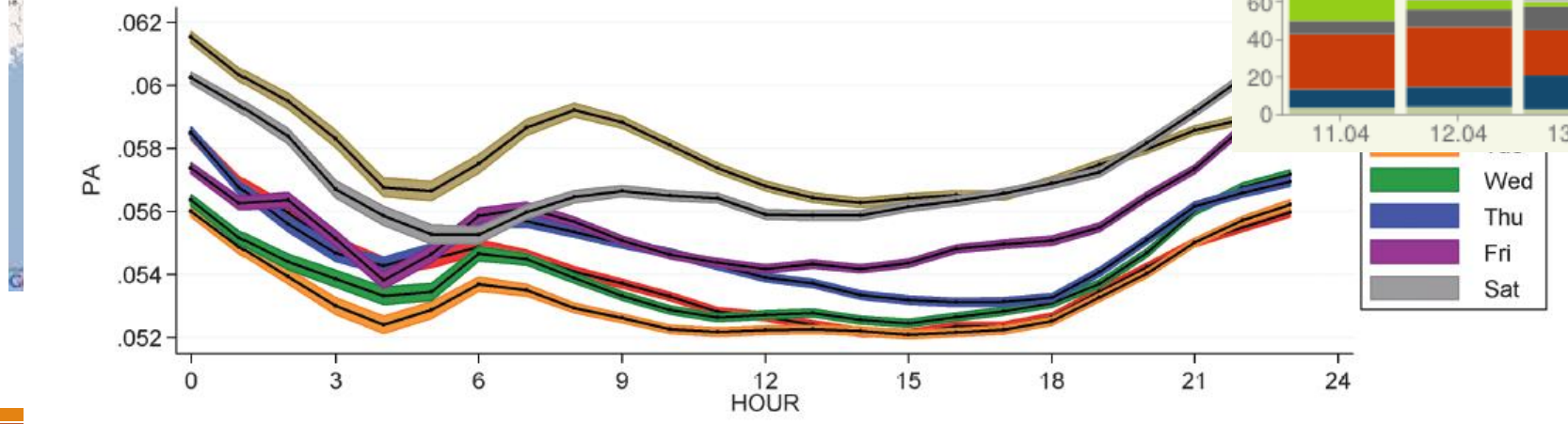


Text mining in Twitter

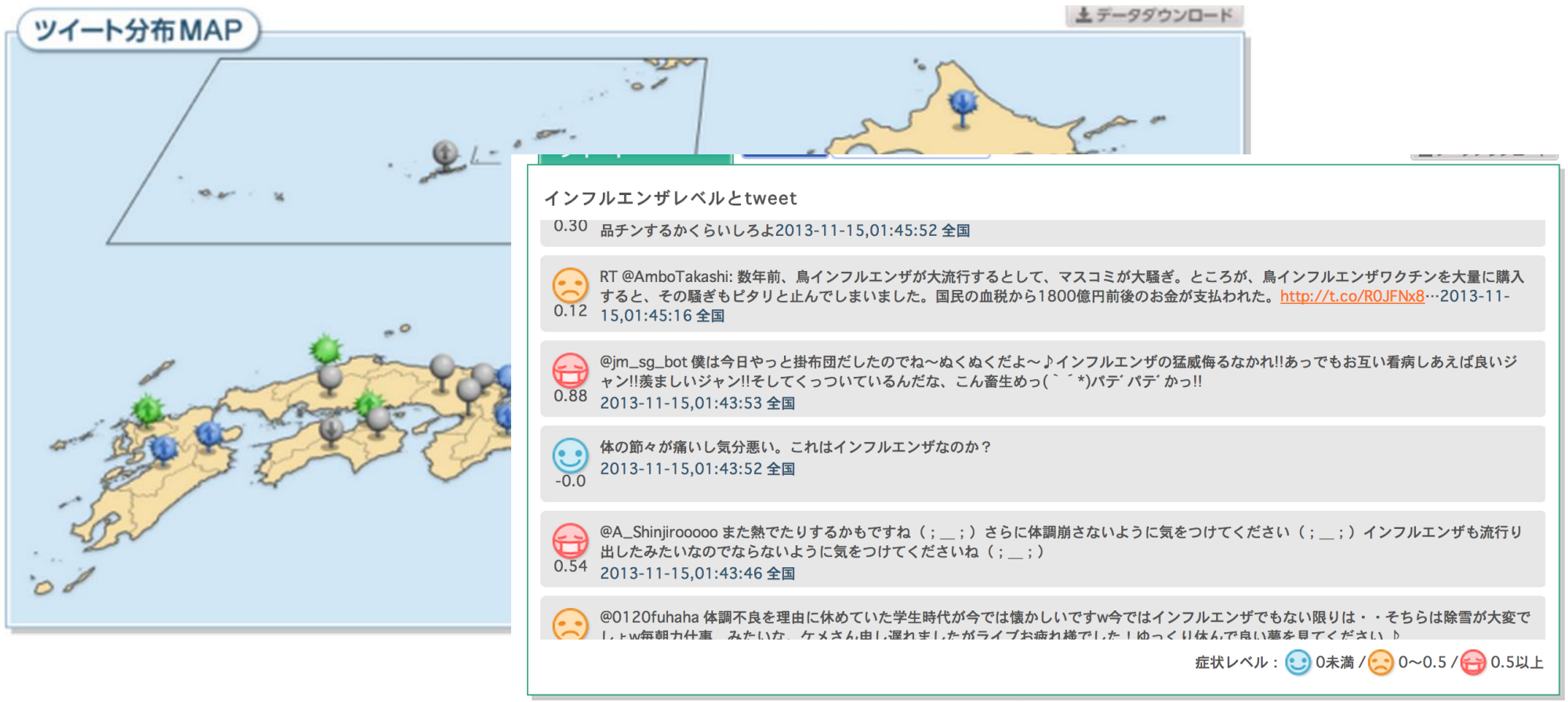






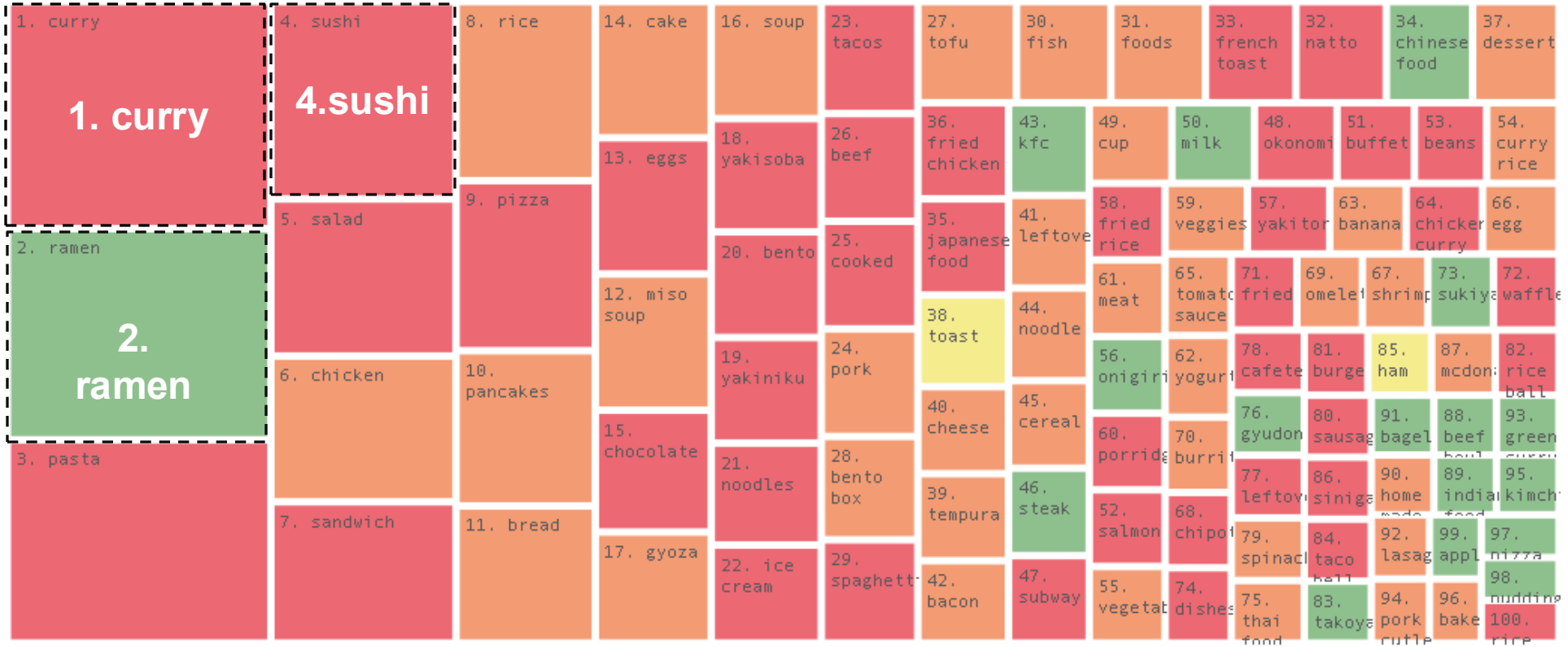


Flu Prediction from Twitter



More applications

Japan FOODMOOD



Powerful application

Facbook *PNAS paper*

- *Private traits and attributes are predictable from digital rec*
  - http://www.pnas.org/content/early/2013/03/06/1218772110

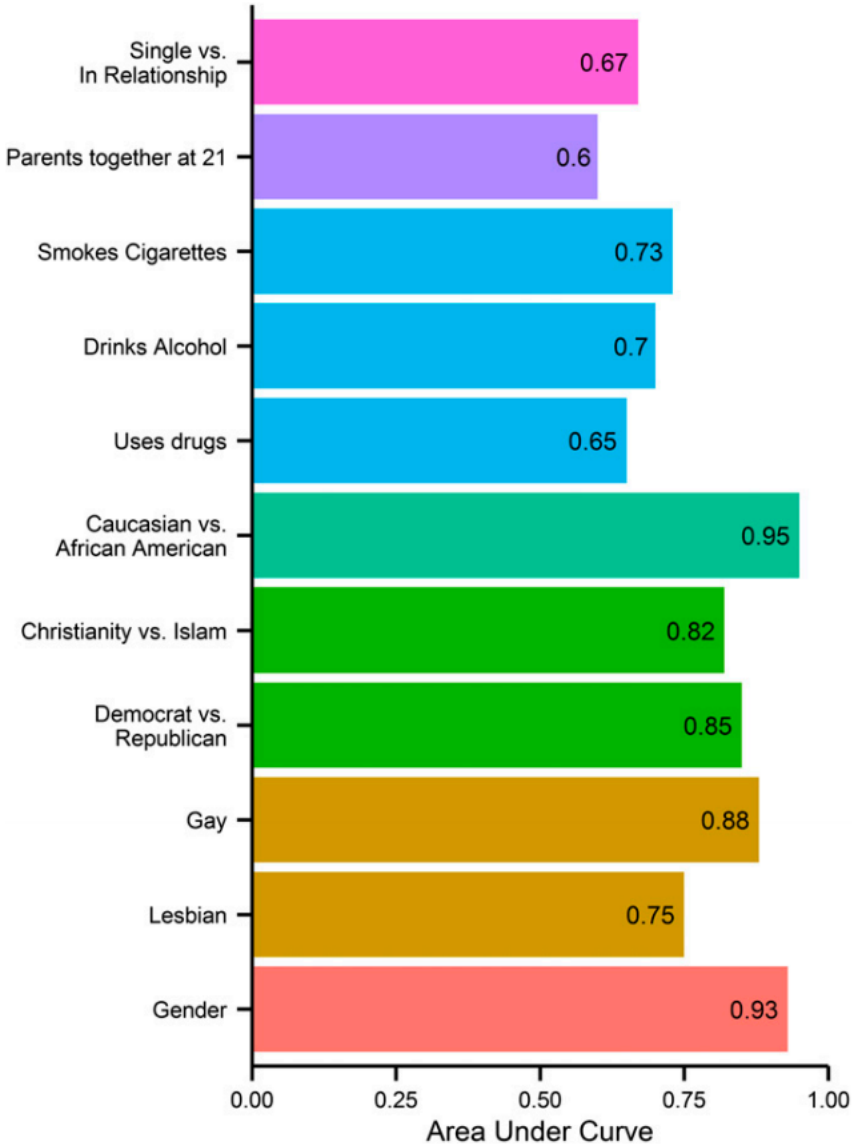
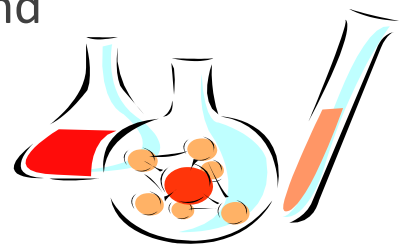


Fig. 2. Prediction accuracy of classification for dichotomous/dichotomized attributes expressed by the AUC.

A medical text mining example

Research objective:

- Follow chains of causal implication to discover a relationship between migraines and biochemical levels.

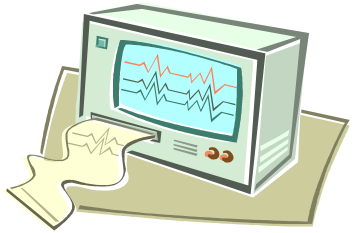


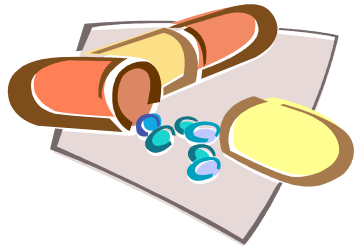
Data:

- medical research papers, medical news (**unstructured text information)**

Key concept types:

- symptoms, drugs, diseases, chemicals…





Example Application: Medical Research

stress is associated with **migraines**

stress can lead to loss of **magnesium**

calcium channel blockers prevent some **migraines**

magnesium is a natural calcium channel blocker

spreading cortical depression (SCD) is implicated in some **migraines**

high levels of **magnesium** inhibit SCD

migraine patients have high platelet aggregability

magnesium can suppress platelet aggregability

(source: Swanson and Smalheiser, 1994)

Unstructured Data Management

Natural language processing is a subset of Unstructured Data Management.

UDM can be broken down into:

- Content and Document Management
- Search and Retrieval
- XML database and tools
- Categorization, Classification, and Visualization

80%-90% of Data is Unstructured

Data Warehouse vs. Document Warehouse

Data warehouse

- Who, what, when, where, how much
- Internally focused
- Operational information
- Rarely include external information

Document warehouse

- Why
- May not be internally focused
- May contain a range of information
- Often integrate external information

The Building Blocks of Language

Morphology
- the study of the structure and form of words

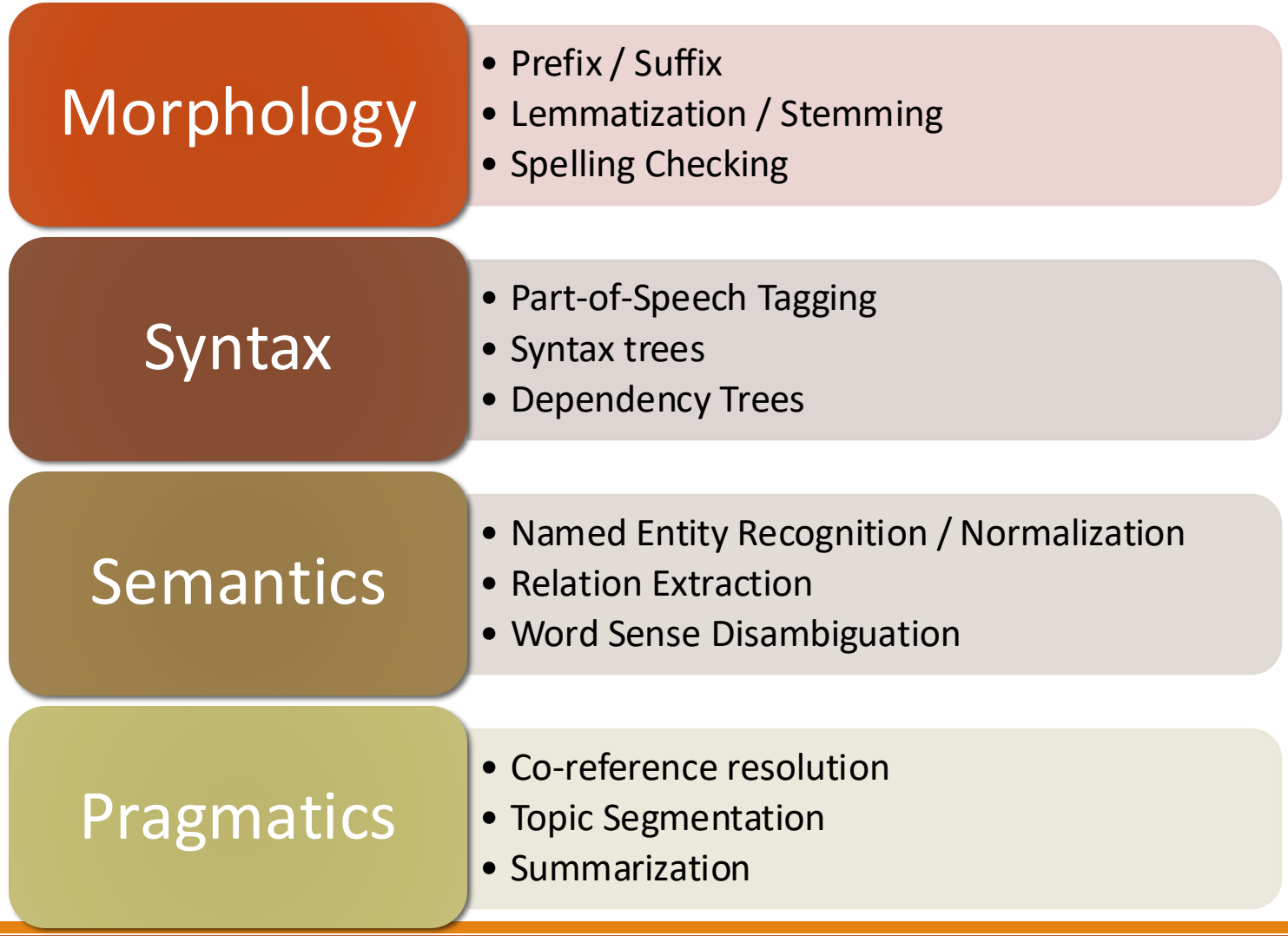
Syntax
- the study of how words and phrases form sentences

Semantics
- relates to the meaning of words and statements

Phonology
- the study of sounds in the language

Pragmatics
- the study of idiomatic phrases that cannot be analyzed with strict semantic analysis

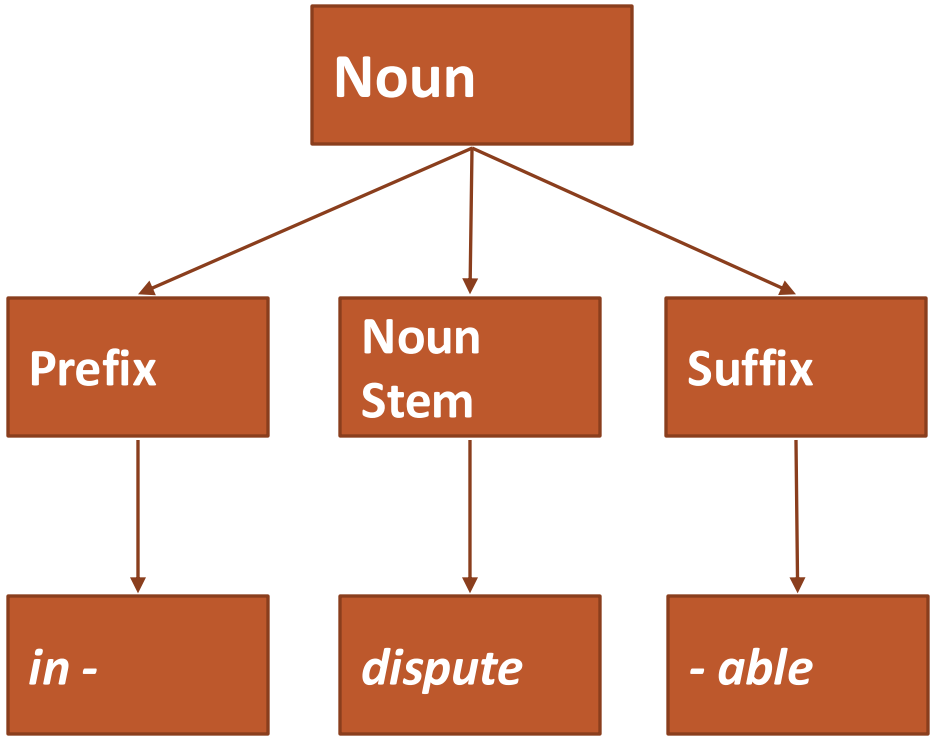
NLP Levels


Morphology
• Prefix / Suffix
• Lemmatization / Stemming
• Spelling Checking
Syntax
• Part-of-Speech Tagging
• Syntax trees
• Dependency Trees
Semantics
• Named Entity Recognition / Normalization
• Relation Extraction
• Word Sense Disambiguation
Pragmatics
• Co-reference resolution
• Topic Segmentation
• Summarization

Morphology

Understanding words

- Stems
- Affixes
  - Prefix
  - Suffix
- Inflectional elements



- Reducing complexity of analysis
- Reduces complexity of representation
- Supports text mining

Morphology level representation

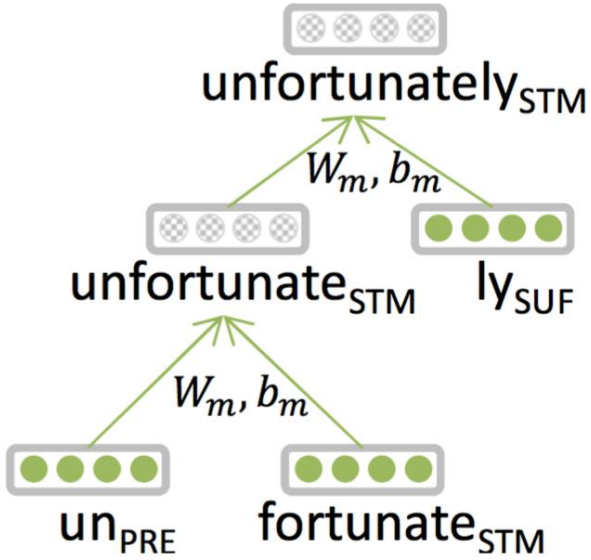
Traditional: Words are made of morphemes

- “unfortunately” = un + fortunate + ly (prefix + stem + suffix)

In DL (Luong, 2013)

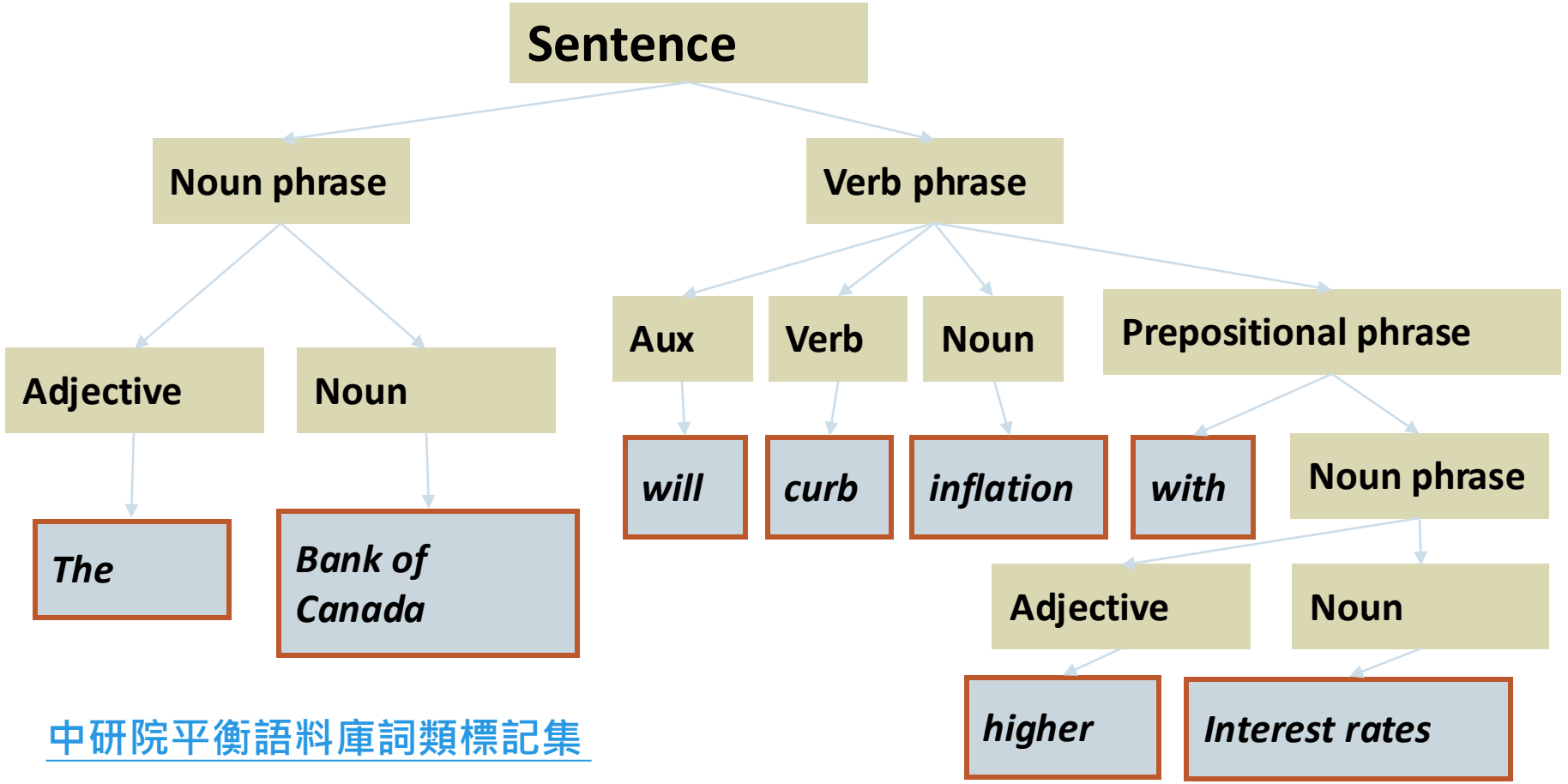
- every morpheme is a vector
- A neural network combines two vectors into one vector

Extend to word, parsing, semantics level representations



Syntax

Ex: The Bank of Canada will curb inflation with higher interest rates.



中研院平衡語料庫詞類標記集

Problems with NLP

Limitations of Natural Language Processing

- Correctly identifying the role of noun phrases
- Representing abstract concepts
- Classifying synonyms
- Representing the number of concepts

Underlying Technology is Based on Linguistics

Text is unstructured, ambiguous, and language dependent.

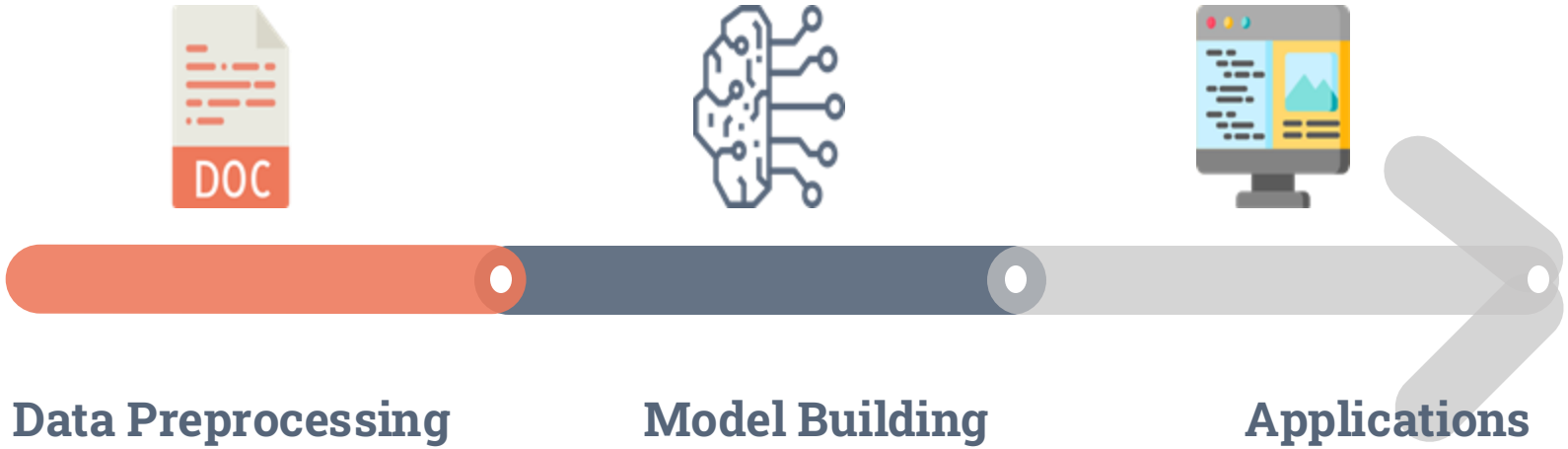
The Linguistic Approach:

- Does not treat a document as a bag of words
- Removes ambiguity by extracting structured concepts

Concepts are the **DNA** of text.

Context Ambiguity

AI-enhanced Natural Language Processing



Data Preprocessing

- Representation Learning
  - Word2Vec. GloVe. ELMo. BERT...
- Named Entity Recognition and Normalization
- ...

Model Building

- Seq2Seq
- AutoEncoders
- Attention Mechanism
- Reinforcement Learning
- Generative Adversarial Networks (GAN)
- ...

Applications

- News/Document Summarization
- Toxic Comment Classification
- Rumor Detection
- Text/Report Generation
- Sentiment Analysis
- Latent Aspect Mining
- Dialogue System

First meet with text

FROM THE VIEW OF INFORMATION RETRIEVAL

Information Retrieval

Analyzing the textual content of individual Web pages

- given user's query
- determine a maximally related subset of documents

Retrieval

- **index** a collection of documents (access efficiency)
- **rank** documents by importance (accuracy)

Categorization (classification)

- assign a document to one or more categories
- *Man-made (*Yahoo & Dmoz *) v.s. automation*

Indexing

Inverted index

- effective for very large collections of documents
- associates lexical items to their occurrences in the collection

Terms ω

- lexical items: *words* or *expressions*

Vocabulary V

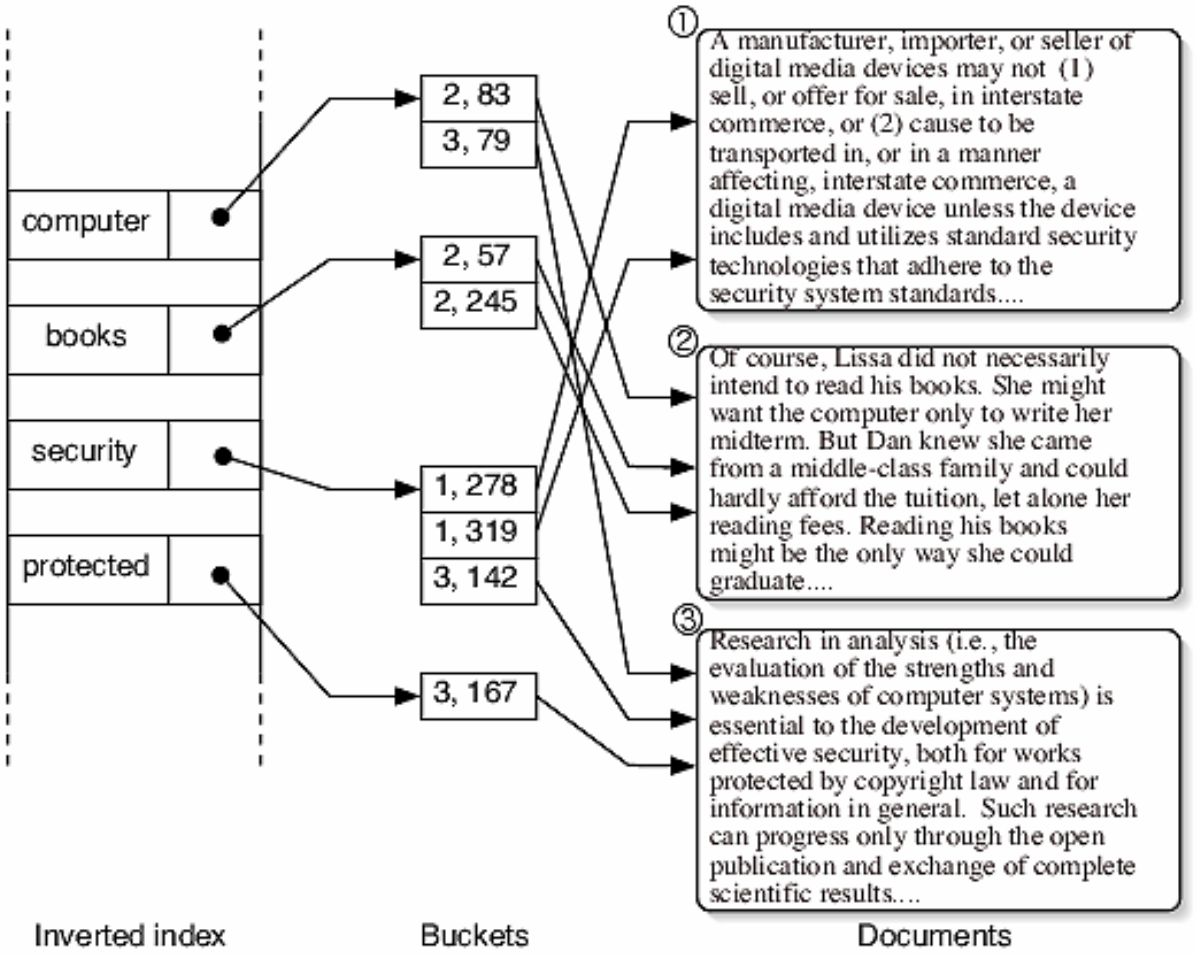
- the set of terms of *interest*

Inverted Index

The simplest example

- a dictionary
  - each key is a term $\omega \in V$
  - associated value $b(\omega)$ points to a *bucket* (posting list)
    - a bucket is a list of pointers marking all occurrences of ω in the text collection

Inverted Index



Bucket entries:

1. document identifier (**DID**)
   - the ordinal number within the collection
2. separate entry for each occurrence of the term
   - DID
   - offset (in characters) of term's occurrence within this document
     - present a user with a short context
       - E.g., google results
     - enables *vicinity* queries

Lexical Processing

Performed prior to indexing or converting documents to vector representations

- Tokenization
  - extraction of terms from a document
- Text conflation and vocabulary reduction
  - Stemming
    - reducing words to their root forms
    - *Porter algorithm, http://www.tartarus.org/~martin/PorterStemmer/*
  - Removing stop words
    - common words, such as articles, prepositions, non-informative adverbs
    - 20-30% index size reduction

Tokenization

Extraction of terms from a document

- stripping out
  - administrative metadata
  - structural or formatting elements

Example

- removing HTML tags
- removing punctuation and special characters
- folding character case (e.g. all to lower case)

Stemming

Want to reduce all morphological variants of a word to a single index term

- e.g. a document containing words like *fish* and *fisher* may not be retrieved by a query containing *fishing* (no *fishing* explicitly contained in the document)
  - *But what for fishing rod*

Stemming - reduce words to their root form

- e.g. *fish* – becomes a new index term

Porter stemming algorithm (1980)

- relies on a preconstructed *suffix list* with associated rules
  - e.g. *if suffix=IZATION and prefix contains at least one vowel followed by a consonant, replace with suffix=IZE*
    - BINARIZATION => BINARIZE

Issues about stop words

What are stop words ?

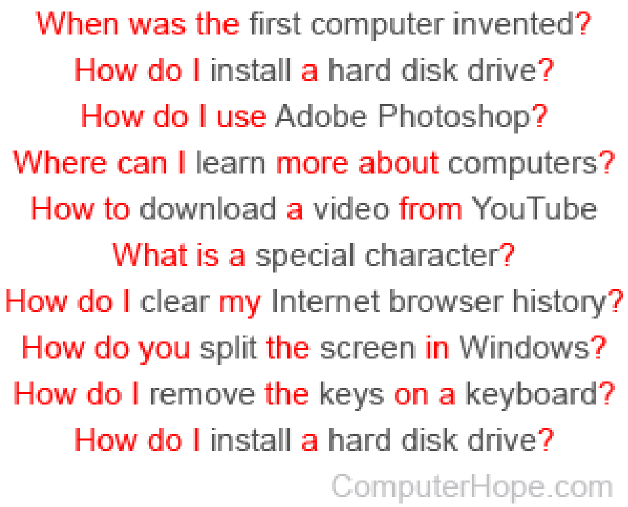
- A static set or dynamic one ?
- Depend on your application ?

How to remove stop words ?

- Dictionary-based matching

When to remove stop words ?

- Avoid to broke the semantics

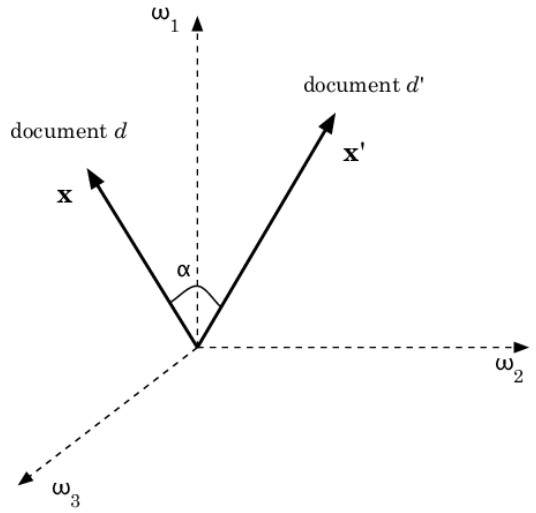


Document Representation: Vector-space Model

Text documents are mapped to a high-dimensional vector space

Each document *d*

- represented as a sequence of terms $\omega(t)$
  - $d = (\omega(1), \omega(2), \omega(3), ..., \omega(|d|))$



ω_1, ω_2 *and* ω_3 are terms in document, *x and x′* are document vectors

Vector-space representations are ***sparse***, *|V| >> |d|* *(the number of distinct terms in any single document)*

Term frequency (TF)

A term that appears many times within a document is likely to be more important than a term that appears only once

n_{ij} - Number of occurrences of a term ω_j in a document d_i

Term frequency

$$TF_{ij} = \frac{n_{ij}}{|d_i|}$$

Inverse document frequency (IDF)

A term that occurs in a few documents is likely to be a better *discriminator* than a term that appears in most or all documents

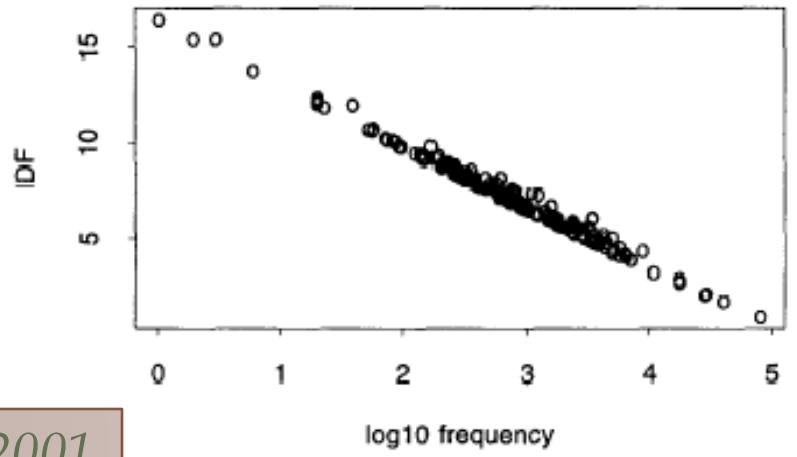
n_j - Number of documents which contain the term ω_j

n - total number of documents in the set

Inverse document frequency

$$IDF_j = \log \frac{n}{n_j}$$

absolute measure of term importance



A theoretical justification of IDF, Papineni 2001

Document Similarity

Ranks documents by measuring the similarity between each document and the query

Similarity between two documents *d* and *d′* is a function $s(d, d') \in R$

In a vector-space representation the *cosine coefficient* of two document vectors is a measure of similarity

$$\cos(x, x') = \frac{x^T x'}{\|x\| \cdot \|x'\|}$$

$O(|d| + |d'|)$

tf-idf weighting has many variants

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times\mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0,\log\frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

Columns headed 'n' are acronyms for weight schemes.

Many search engines allow for different weightings for queries v.s. documents

ltn.lnc ?

BM25

Okapi BM25 (1980s)

- A ranking function used by search engines to rank matching documents according to their relevance to a given query.

BM11, BM15

k, b are free parameters
generally k=2, b=0.75

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i,D)\cdot(k_1+1)}{f(q_i,D)+k_1\cdot(1-b+b\cdot\frac{|D|}{\text{avgdl}})},$$

$$\text{IDF}(q_i) = \log\frac{N-n(q_i)+0.5}{n(q_i)+0.5},$$

Alternative TF-IDF

Word Representation

-- Motivating Continuous, Distributed Representations

Bag of Words

Very good drama although it appeared to have a few blank areas leaving the viewers to fill in the action for themselves. I can imagine life being this way for someone who can neither read nor write. This film simply smacked of the real world: the wife who is suddenly the sole supporter, the live-in relatives and their quarrels, the troubled child who gets knocked up and then, typically, drops out of school, a jackass husband who takes the nest egg and buys beer with it. 2 thumbs up… very very very good movie.

('the', 8),
(',', 5),
('very', 4),
('.', 4),
('who', 4),
('and', 3),
('good', 2),
('it', 2),
('to', 2),
('a', 2),
('for', 2),
('can', 2),
('this', 2),
('of', 2),
('drama', 1),
('although', 1),
('appeared', 1),
('have', 1),
('few', 1),
('blank', 1)
…..

錢, 不是問題　　不, 錢是問題

Example: Opinion Mining

今天的牛排很讚又便宜
這家餐廳的前菜很有名
肉質很嫩

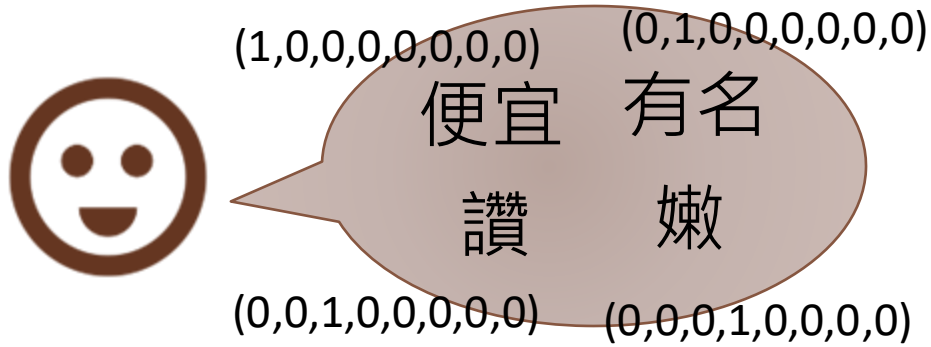
牛小排煮的有點太老
難吃又貴
價格太貴服務又差

One-hot encoding

Vocabulary space

Term vector

	便宜	有名	讚	嫩	難吃	太貴	差	老
便宜	1	0	0	0	0	0	0	0
有名	0	1	0	0	0	0	0	0
讚	0	0	1	0	0	0	0	0
嫩	0	0	0	1	0	0	0	0
難吃	0	0	0	0	1	0	0	0
太貴	0	0	0	0	0	1	0	0
差	0	0	0	0	0	0	1	0
老	0	0	0	0	0	0	0	1



= (1,1,1,1,0,0,0,0)

Word embedding (dense space)

Concept space

Term vector

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8
便宜	1	3	2	3	0	-2	2	0
有名	3	1	4	2	0	2	0	1
讚	0	0	1	0	0	1	-1	0
嫩	1	3	0	1	0	0	0	0
難吃	0	0	0	0	1	0	2	0
太貴	0	0	0	0	3	1	0	1
差	-1	0	1	-1	0	2	1	0
老	0	-2	-1	0	1	3	2	1

Synonymy and Polysemy

Synonymy (同義詞)

- the same concept can be expressed using different sets of terms
  - e.g. *bandit*, *brigand*, *thief*
- *negatively affects recall ?*

Polysemy (一詞多義)

- identical terms can be used in very different semantic contexts
  - e.g. *bank*
    - repository where important material is saved
    - the slope beside a body of water
- *negatively affects precision ?*

Hybrid cases

Concept matching v.s. term matching

A query may conceptually be very close to a given set of documents, but corresponding consine similarity is small, because …

- *Using different language*
  - (計程車, 出租车, 的士)
- *Using different domain lexicons*
  - (智慧型行動運算裝置, 手機), (高效能溫度調節器, 冰箱)
- *A small fraction of the entire dictionary for present terms*
  - (COVID-19, 新冠病毒, SARS-CoV-2病毒)

How do we represent the meaning of a word?

Dictionary based solutions

WordNet: a resource containing list of synonyms sets and hypernyms (“is a” relationships)

e.g. synonym sets containing “good”:

```
from nltk.corpus import wordnet as wn
for synset in wn.synsets("good"):
    print "(%s)" % synset.pos(),
    print ", ".join([l.name() for l in synset.lemmas()])
```

```
(adj) full, good
(adj) estimable, good, honorable, respectable
(adj) beneficial, good
(adj) good, just, upright
(adj) adept, expert, good, practiced,
proficient, skillful
(adj) dear, good, near
(adj) good, right, ripe
…
(adv) well, good
(adv) thoroughly, soundly, good
(n) good, goodness
(n) commodity, trade good, good
```

e.g. hypernyms of “panda”:

```
from nltk.corpus import wordnet as wn
panda = wn.synset("panda.n.01")
hyper = lambda s: s.hypernyms()
list(panda.closure(hyper))
```

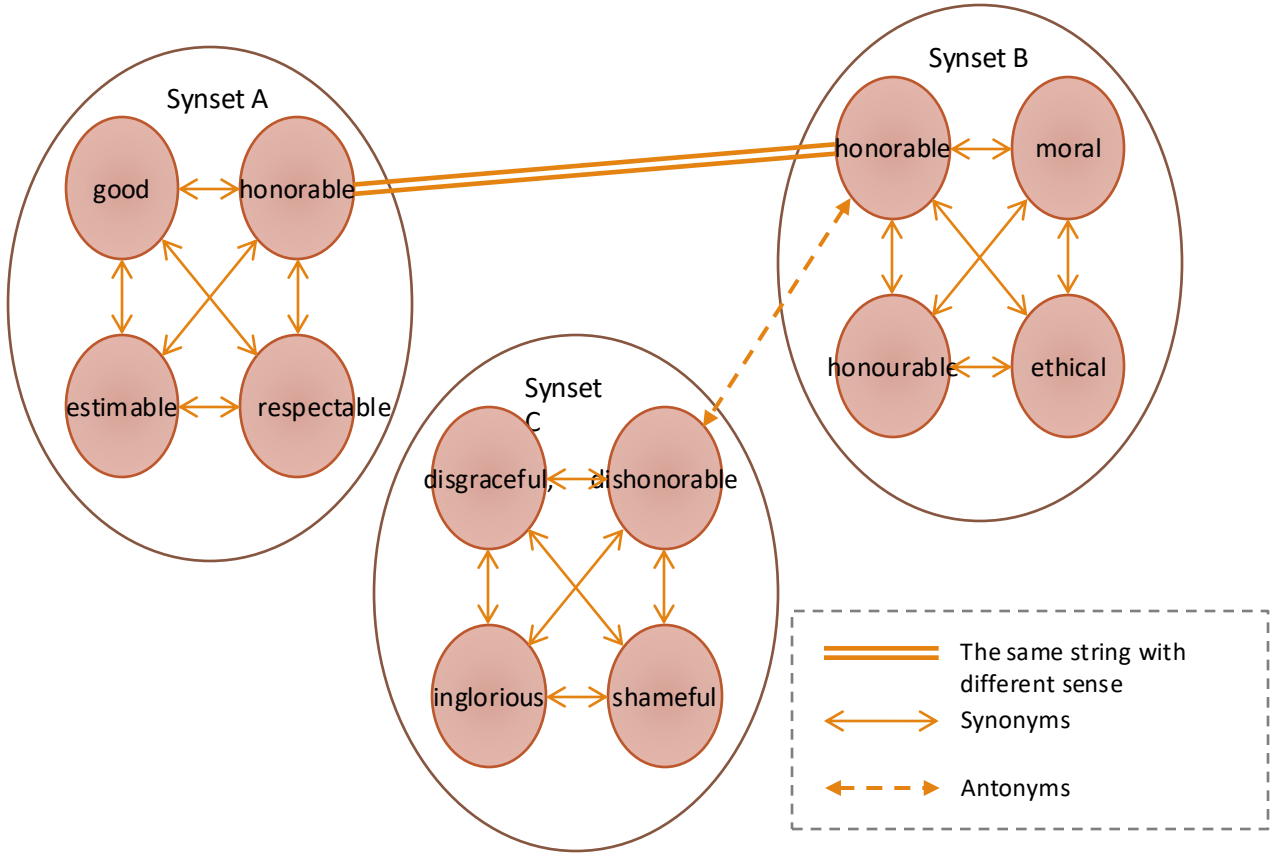
```
[Synset('procyonid.n.01'),
Synset('carnivore.n.01'),
Synset('placental.n.01'),
Synset('mammal.n.01'),
Synset('vertebrate.n.01'),
Synset('chordate.n.01'),
Synset('animal.n.01'),
Synset('organism.n.01'),
Synset('living_thing.n.01'),
Synset('whole.n.02'),
Synset('object.n.01'),
Synset('physical_entity.n.01'),
Synset('entity.n.01')]
```

Cited from Stanford cs224n

How do we represent the meaning of a word?

Issues

- context is not considered
- New words missing
- Maintenance cost



Hard to measure the similarity!
-- one-hot vector
-- every different words are orthogonal

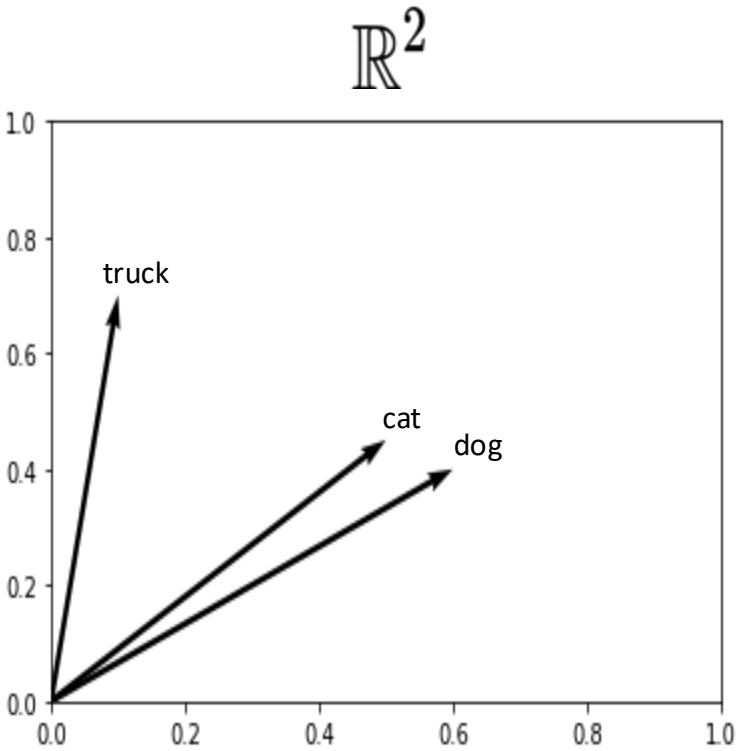
Solution: Continuous, Distributed Representations

By continuous we mean **real-valued**

Distributed means a vector in a space, like

$$\text{dog} = \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}$$

$$\text{cat} = \begin{bmatrix} 0.5 \\ 0.4 \end{bmatrix}$$



Breakthrough Paper: Bengio et al. 2003: A Neural Probabilistic Language Model

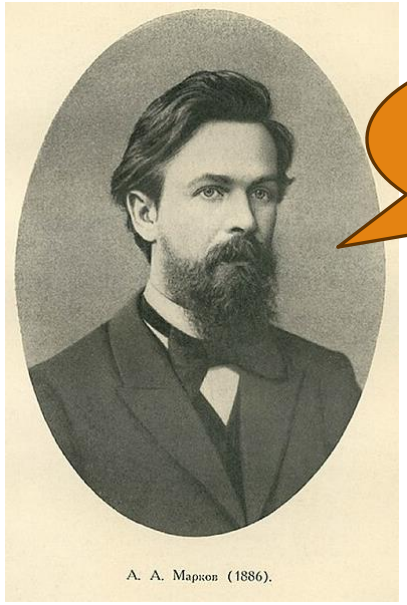
In a nutshell, the idea of the proposed approach can be summarized as follows:

1. associate with each word in the vocabulary a distributed *word feature vector* (a real-valued vector in $\mathbb{R}^m$),
2. express the joint *probability function* of word sequences in terms of the feature vectors of these words in the sequence, and
3. learn simultaneously the *word feature vectors* and the parameters of that *probability function*.

We can also pre-train our vectors with encoded world knowledge (e.g. similarity)

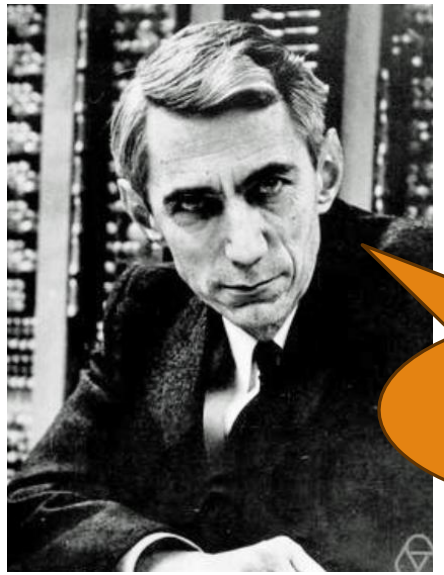
http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

語言模型 (Language Model)



[1913] The chance of a letter appearing depends on the letter before it.

Andrey Markov
1856 - 1922



[1951] Prediction and Entropy of Printed English

Claude Shannon
1916 – 2001

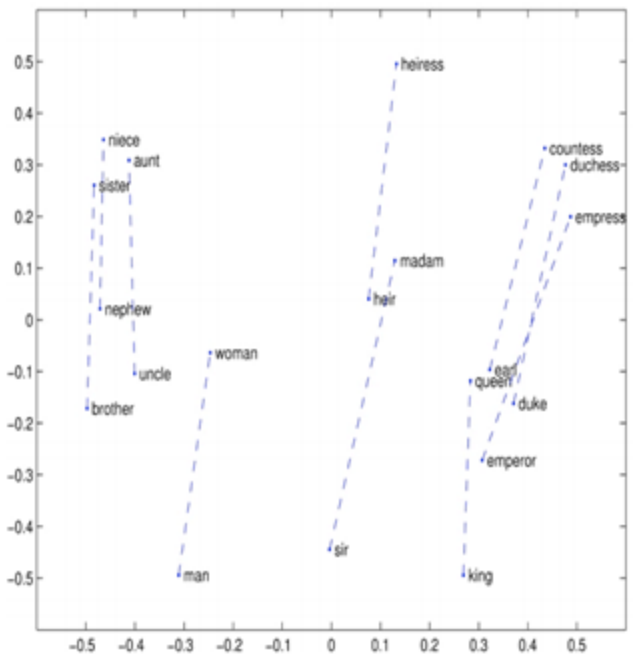
Why Does This Work?

1. Similar words are expected to have similar vector representations (and be closer in vector space)

2. The probability function is a *smooth* function of feature values
   - Small changes in the feature values will induce a small change in the value of the function
   - This is *not* true for unorganized discrete spaces, where small changes in input can lead to large changes in the function value

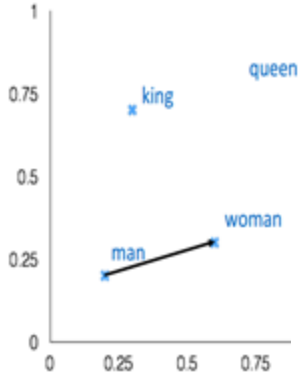
3. Therefore, the presence of one sentence in the training set can effectively distribute probability density in the vector space for a combinatorial number of unseen sentences

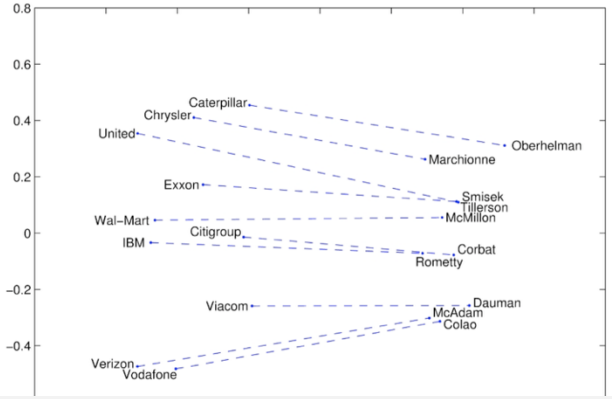
Learning Vector Representation of Words

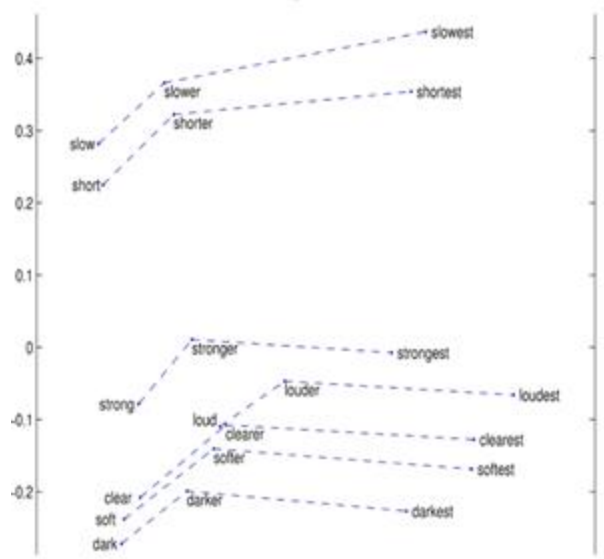


man:woman :: king:?

+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]







0. *frog*
1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

3. litoria

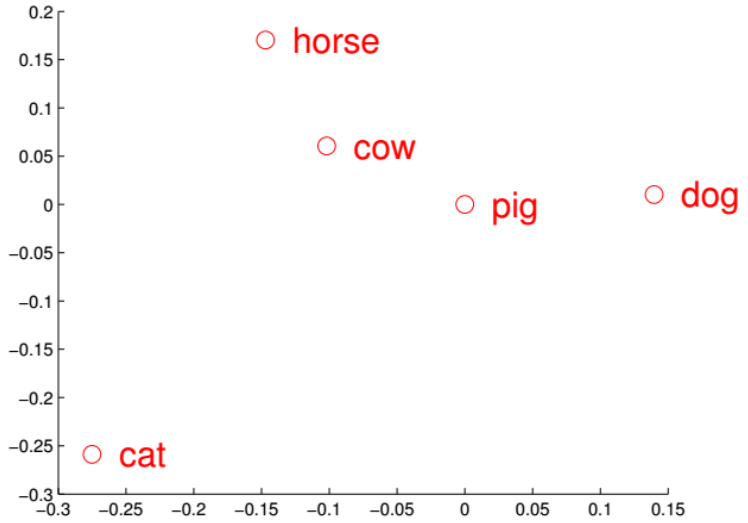
4. leptodactylidae

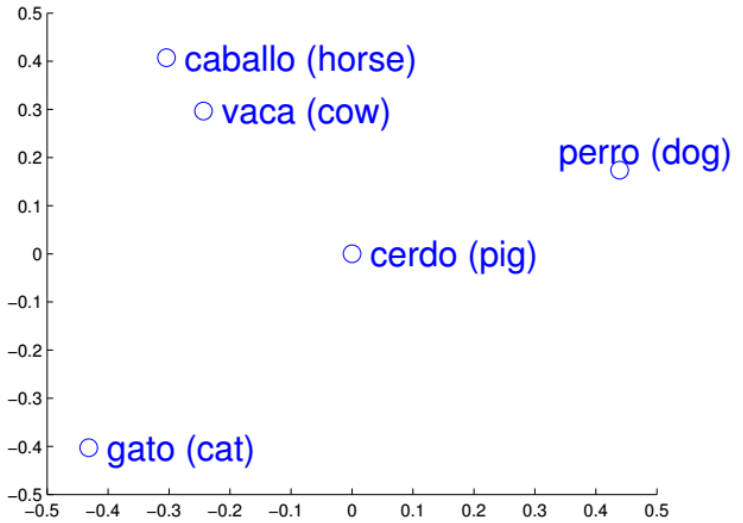
5. rana

7. eleutherodactylus

Representation Vector Usage

Expression	*Nearest token*
Paris - France + Italy	Rome
bigger - big + cold	colder
sushi - Japan + Germany	bratwurst
Cu - copper + gold	Au
Windows - Microsoft + Google	
Montreal Canadiens - Montreal + Toronto	To





Distributional Hypothesis

"A word is characterized by the company it keeps." (Firth 1957)

In other words, similar words will appear in similar contexts

The cat licked its fur

The dog licked its fur

No surprise "cat" and "dog" both appear near "lick" and "fur". We should find they also often appear near "eat", "run", "bite" and so on… but not near words like "read", "sophisticated", or "wheel".

…government debt problems turning into	***banking***	*crises as happened in 2009…*
…saying that Europe needs unified	***banking***	*regulation to replace the hodgepodge…*
…India has just given its	***banking***	*system a shot in the arm…*

Latent Semantic Analysis

"Indexing by latent semantic analysis", S. Deerwester, 1990

Using a corpus of text as input, draw a window of defined length around each word and count co-occurrence statistics. The resulting matrix contains our word vectors.

Count Co-Occurrence

$$\begin{bmatrix} \cdots & w_1 & \cdots \\ & \vdots & \\ \cdots & w_n & \cdots \end{bmatrix}$$

Simple example for word co-occurrence usage

Training corpus

- “I like deep learning.”, “I like NLP.”, “I enjoy programming."

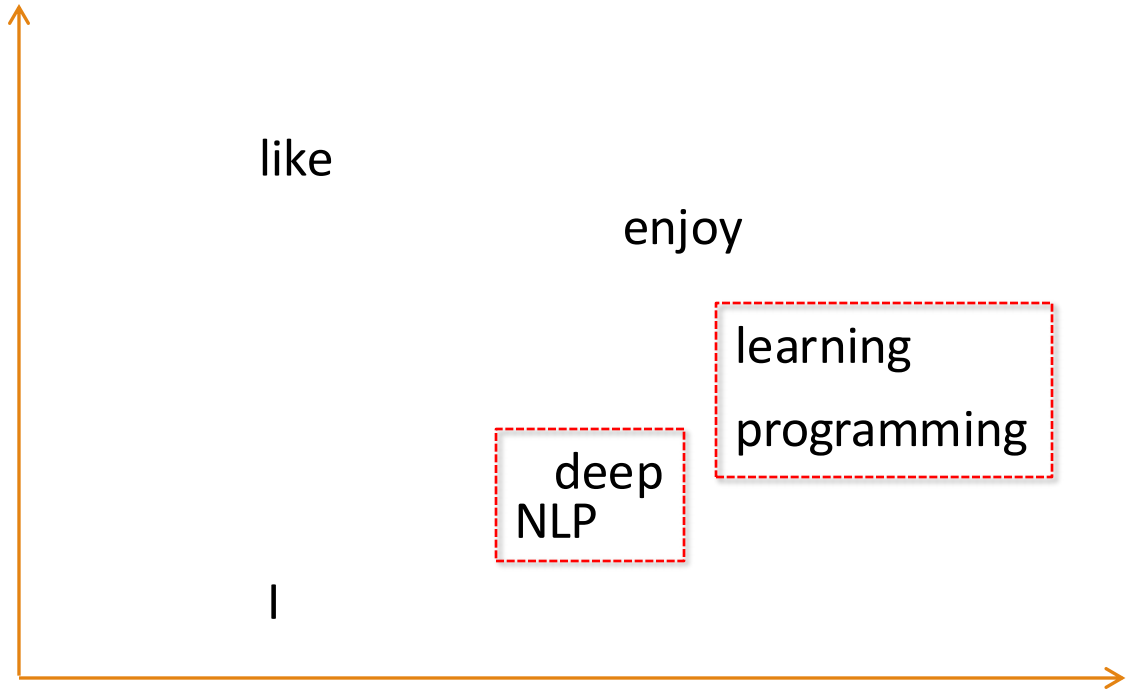
cooccurence	I	like	enjoy	deep	learning	NLP	programming
I	0	2	1	0	0	0	0
like	2	0	0	1	0	1	0
enjoy	1	0	0	0	0	0	1
deep	0	1	0	0	1	0	0
learning	0	0	0	1	0	0	0
NLP	0	1	0	0	0	0	0
programming	0	0	1	0	0	0	0

Simple example for word co-occurrence usage

SVD on this co-occurrence matrix

- Reduce dimension

Use the 2 biggest singular value to represent words



Latent Semantic Analysis

Why need it?

- serious problems for retrieval methods based on term matching
  - vector-space similarity approach works only if the terms of the query are ***explicitly presented*** in the relevant documents
  - *有出現不見得是相關*
  - *沒出現不見得是無關*

- the rich expressive power of natural language
  - often queries contain terms that express *concepts* related to text to be retrieved

TK Landauer "An *Introduction to Latent Semantic Analysis*"

Latent Semantic Indexing(LSI)

A statistical technique

Uses linear algebra technique called *singular value decomposition (SVD)*

- attempts to estimate the hidden structure that generates terms given concepts
- discovers the most important associative patterns between words and concepts

Data driven

- A large collection of sentences or documents is employed

LSI and Text Documents

Let **X** denote a term-document matrix

$$\boldsymbol{X} = [\boldsymbol{x}_1 \ldots \boldsymbol{x}_n]^T$$

- each row is the vector-space representation of a document
- each column contains occurrences of a term in each document in the dataset

Latent semantic indexing

- compute the SVD of ***X***: $\mathrm{X} = U\Sigma V^T$
  - Σ - singular value matrix *(diagonal matrix)*
- set to zero all but largest *K* singular values - $\hat{\Sigma}$
- obtain the reconstruction of ***X*** by: $\hat{\mathrm{X}} = U\hat{\Sigma}V^T$

LSI Example

A collection of documents:

Linux OS

d1: Indian government goes for open-source software

d2: Debian 3.0 Woody released

d3: Wine 2.0 released with fixes for Gentoo 1.4 and Debian 3.0

d4: gnuPOD released: iPOD on Linux… with GPLed software

d5: Gentoo servers running at open-source mySQL database

d6: Dolly the sheep not totally identical clone

d7: DNA news: introduced low-cost human genome DNA chip

d8: Malaria-parasite genome database on the Web

d9: UK sets up genome bank to protect rare sheep breeds

d10: Dolly's DNA damaged

Genome news

LSI Example

The term-document matrix X^T

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	1	0	0	0	1	0	0	0	0	0
software	1	0	0	1	0	0	0	0	0	0
Linux	0	0	0	1	0	0	0	0	0	0
released	0	1	1	1	0	0	0	0	0	0
Debian	0	1	1	0	0	0	0	0	0	0
Gentoo	0	0	1	0	1	0	0	0	0	0
database	0	0	0	0	1	0	0	1	0	0
Dolly	0	0	0	0	0	1	0	0	0	1
sheep	0	0	0	0	0	1	0	0	0	0
genome	0	0	0	0	0	0	1	1	1	0
DNA	0	0	0	0	0	0	2	0	0	1

LSI Example

The reconstructed term-document matrix after projecting on a subspace of dimension K=2

Σ = diag(2.57, 2.49, 1.99, 1.9, 1.68, 1.53, 0.94, 0.66, 0.36, 0.10)

$\hat{\Sigma}$ = diag(2.57, 2.49, 0, 0, 0, 0, 0, 0, 0, 0)

$\hat{X}^T$

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	0.34	0.28	0.38	0.42	0.24	0.00	0.04	0.07	0.02	0.01
software	0.44	0.37	0.50	0.55	0.31	-0.01	-0.03	0.06	0.00	-0.02
Linux	0.44	0.37	0.50	0.55	0.31	-0.01	-0.03	0.06	0.00	-0.02
released	0.63	0.53	0.72	0.79	0.45	-0.01	-0.05	0.09	-0.00	-0.04
Debian	0.39	0.33	0.44	0.48	0.28	-0.01	-0.03	0.06	0.00	-0.02
Gentoo	0.36	0.30	0.41	0.45	0.26	0.00	0.03	0.07	0.02	0.01
database	0.17	0.14	0.19	0.21	**0.14**	0.04	0.25	**0.11**	0.09	0.12
Dolly	-0.01	-0.01	-0.01	-0.02	0.03	0.08	0.45	0.13	0.14	0.21
sheep	-0.00	-0.00	-0.00	-0.01	0.03	0.06	0.34	0.10	0.11	0.16
genome	0.02	0.01	0.02	0.01	0.10	0.19	1.11	0.34	0.36	0.53
DNA	-0.03	-0.04	-0.04	-0.06	0.11	0.30	**1.70**	0.51	0.55	0.81

Negative value!

	d5
	1
	0
	0
	0
	0
	1
	1
	0
	0
	0
	0

database	0	0	0	0	1	0	0	1	0	0

Problems

- The original matrix still has the same dimension as our vocabulary size - potentially 100,000 x 100,000
- Matrix is extremely sparse
- We can perform Singular Value Decomposition (SVD) for dimensionality reduction, however at a quadratic computational cost
- Adding a new word changes the entire matrix

Probabilistic Model: Some Choices

Unary language model

$$P(w_1, \cdots, w_n) = \prod_i P(w_i)$$

Ridiculous not to consider word order

Binary language model

$$P(w_1, \cdots, w_n) = \prod_i P(w_i | w_{i-1})$$

Better but still limited by short context distance

word2vec models (using window m to get more context)

Continuous Bag of Words (CBOW)

$$P(w_c | w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m})$$

The cat [center word] its fur

Skip-Gram (which we will focus on)

$$P(w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m} | w_c)$$

[left context] licked [right context]

Skip-Gram: Training Data

Corpus

The cat licked its fur. The truck moved.

Context window size = 1

Training Word Pairs

Center	Context
the	cat
the	truck
cat	the
cat	licked
licked	cat
licked	its
…	…

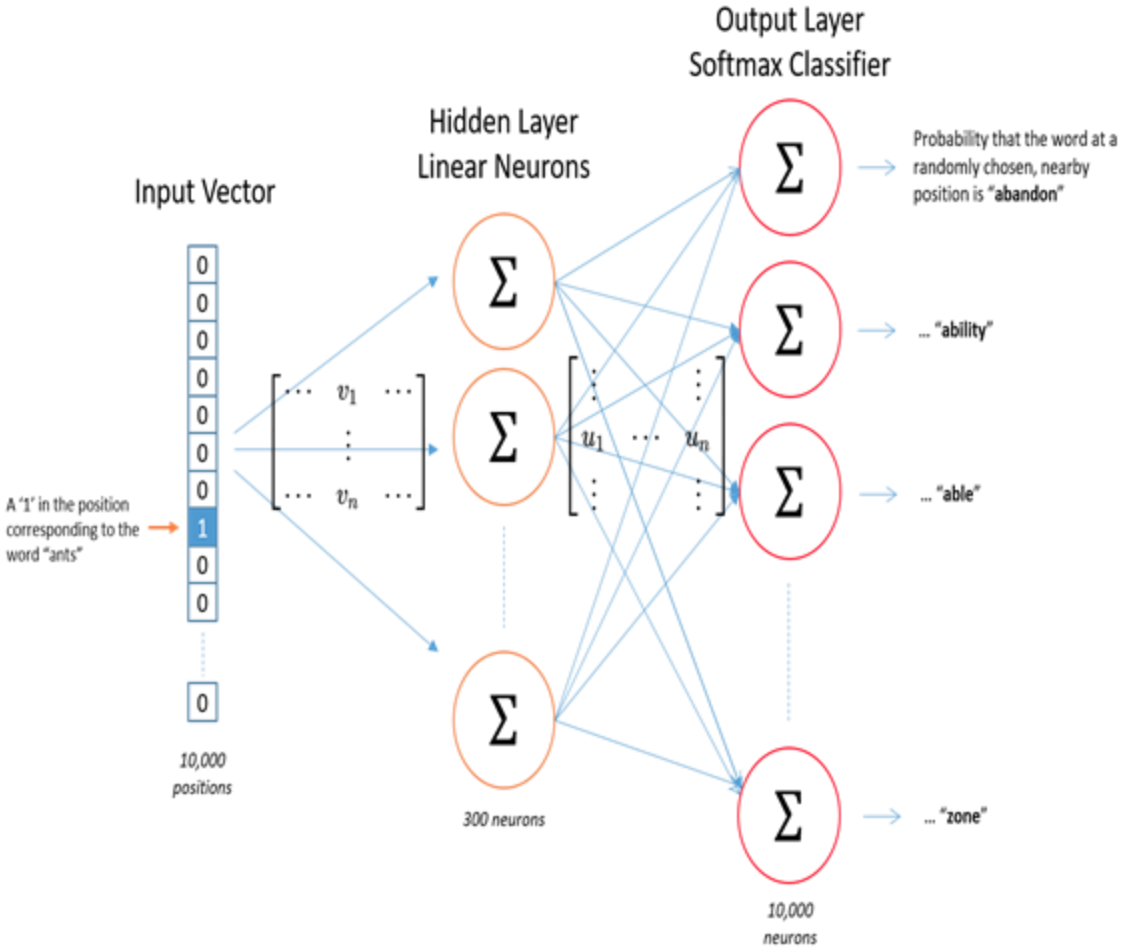
What if context size > 1?

- More word pairs
- Only pass two words to the model at a time

http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

Model Parameters

word2vec uses a single hidden layer feedforward neural network



1. Input to hidden layer matrix has our target word embeddings $\mathbf{v}_i$

2. Hidden to output layer matrix has another set of word embeddings called output embeddings $\mathbf{u}_i$

Recap: Skip-Gram

Ingredients:

1. A probabilistic model
2. Small chunks of data for training
3. Model parameters

Iterative process:

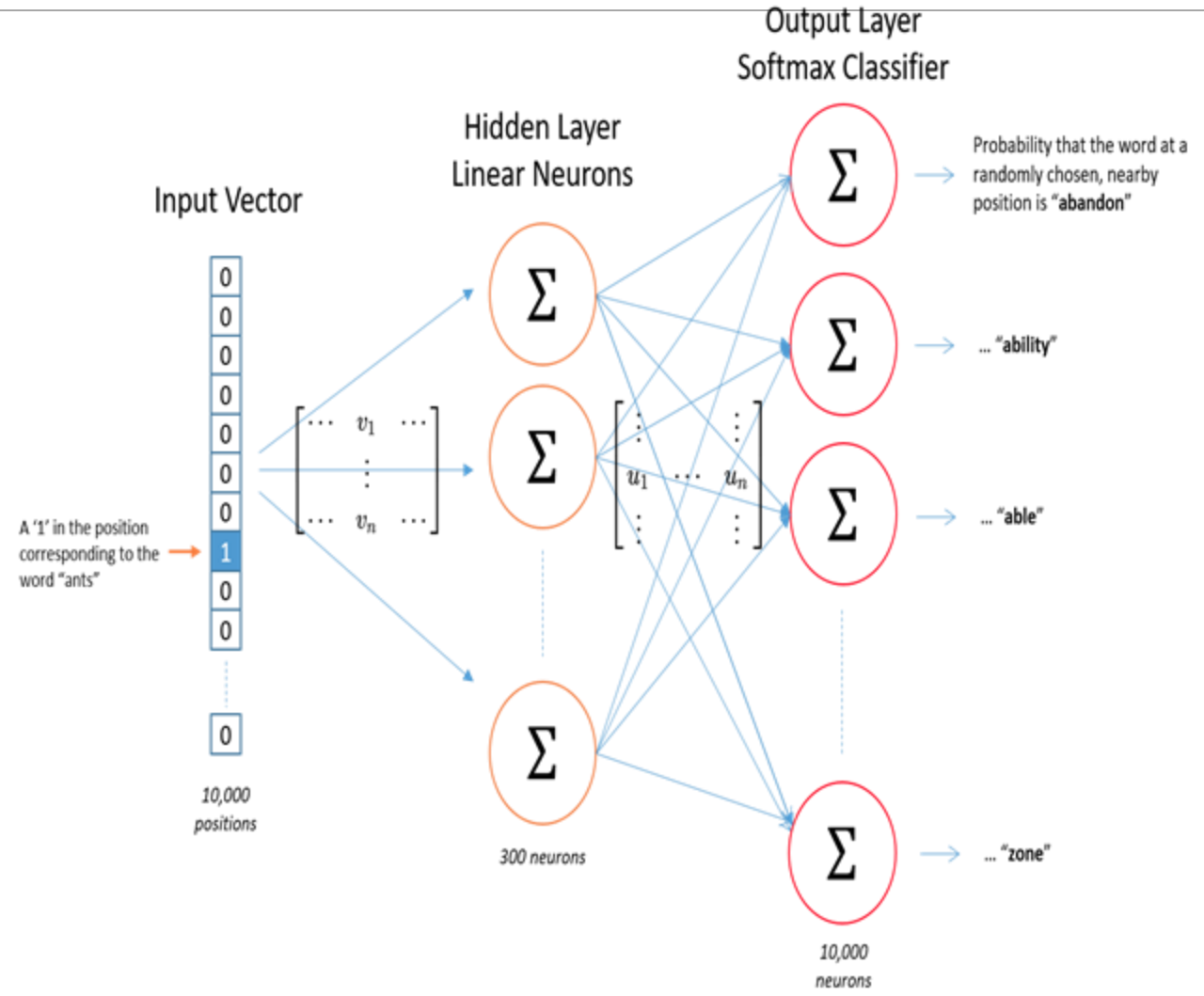
1. Generate probability estimates
2. Calculate the error in the estimates
3. Update the parameters until optimized

$$P(w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m} | w_c)$$

Training word pairs

Neural network parameters **V** and **U**

Generate Probability Estimates



(1) Input vector selects input embedding from hidden layer matrix

(2) Softmax over multiplication with output matrix creates probability distribution over the vocabulary

This should be high for context words, low for others

Network Input

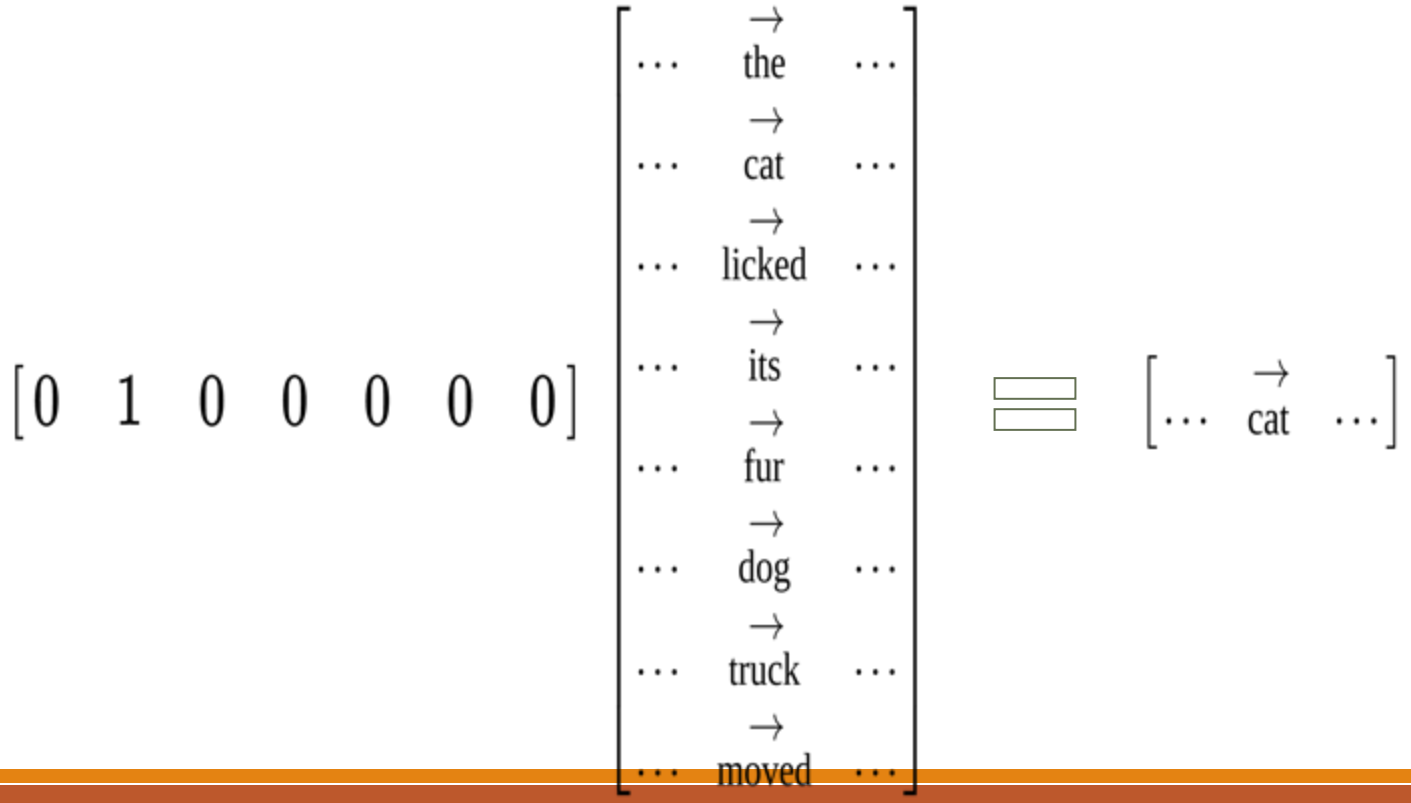
We imagine the words are encoded as “one-hot” vectors (all zeros except a one at the the word index in the embedding matrix)

```
vocab = {
    'the': 1,
    'cat': 2,
    'licked': 3,
    'its': 4,
    'fur': 5,
    'truck': 6,
    'dog': 7,
    'moved': 8}
```

$$\text{the} = \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} \quad \text{cat} = \begin{bmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix} \quad \text{licked} = \begin{bmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix} \quad \text{etc...}$$

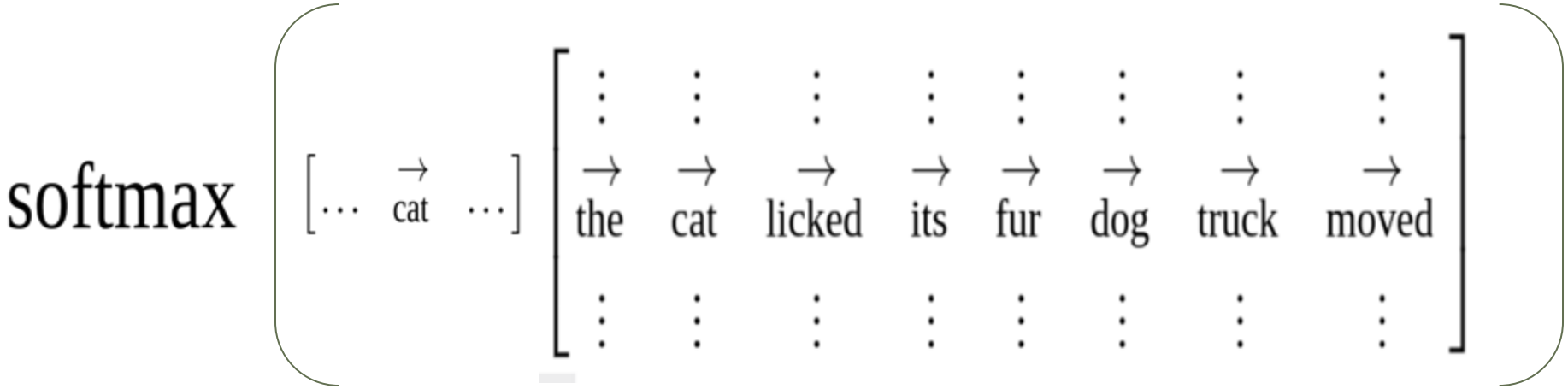
Embedding Lookup

In practice we don't do matrix multiplication, but just use the word index to pick out the vector.



Probability Over Vocab

Our word vector does a dot product with every word's output embedding, then applying softmax gives a probability distribution over the vocabulary



$$\text{probability word } w_i \text{ in context} = \hat{y}_i = P(w_i|\mathbf{v}_c, \mathbf{U}) = \frac{\exp(\mathbf{u}_{w_i}^T \mathbf{v}_c)}{\sum_{j=1}^{V} \exp(\mathbf{u}_{w_j}^T \mathbf{v}_c)}$$

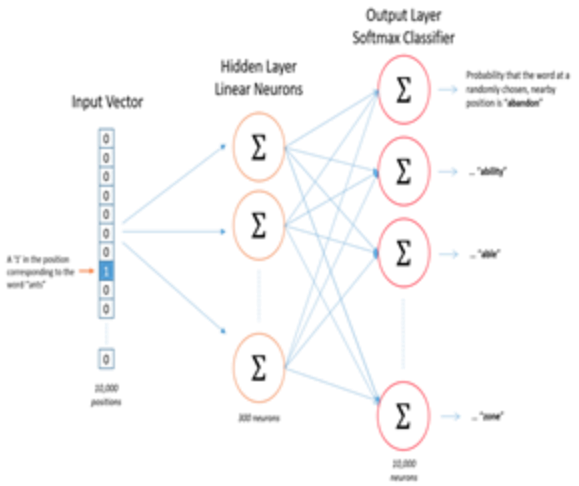
Calculate Error in Estimates

Training Pair:

$$\text{center word} = \text{cat} \quad = [0 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0] = \text{input}$$

$$\text{context word} = \text{licked} \quad = [0 \;\; 0 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0] = \text{target}$$

$$[0 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0]$$



$$\hat{\mathbf{y}} = [0.1 \;\; 0.0 \;\; 0.1 \;\; 0.3 \;\; 0.2 \;\; 0.1 \;\; 0.1 \;\; 0.1]$$

Loss Function

Cross entropy measures the distance between probability distributions

$$\mathrm{CE}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{i=1}^{V} y_i \log(\hat{y}_i)$$

$$\sum \left(\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} * \log \left(\begin{bmatrix} 0.1 & 0.0 & 0.1 & 0.3 & 0.2 & 0.1 & 0.1 & 0.1 \end{bmatrix} \right) \right)$$

GloVe (Global Vectors) 2014

- word2vec is excellent at learning similarity and linear regularities
- But it doesn't make full use of co-occurrence statistics
- GloVe improves word2vec by considering global statistical information
- Pre-Trained vectors available here: https://nlp.stanford.edu/projects/glove/
- The paper is worth reading

GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning
Computer Science Department, Stanford University, Stanford, CA 94305
jpennin@stanford.edu, richard@socher.org, manning@stanford.edu

FastText

FastText is much newer and has also seen good results.

Considers character-level features, and can thus take a guess and generate vectors for out-of-vocabulary words, taking advantage of patterns in morphology

Pre-trained vectors are available in many languages: https://fasttext.cc/

What Vectors Should I Use?
--It depends.

GloVe is very widely used with good results.

FastText is new and adoption may be slow, but can also achieve good results.

Training your own with word2vec may be indicated when you have many "out-of-vocabulary" words - i.e. your dictionary has lots of words for which there are no pre-trained vectors.

This can happen in special areas - for example when dealing with classical Chinese text.

A good idea is to try different vectors on specific tasks.

Evaluation: Example dataset: WordSim353

http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/

Summary

SOP in traditional NLP

Structural analysis, document representation

Basic word understanding

Word vector