

# 자연어 처리 DAY 1

---

Jaegul Choo

Associate Professor, Graduate School of AI, KAIST

1.

# Intro to Natural Language Processing(NLP)

Academic Disciplines related to NLP

Trends of NLP

- Natural language processing (NLP), which aims at properly understanding and generating human languages, emerges as a crucial application of artificial intelligence, with the advancements of deep neural networks.
- This course will cover various deep learning approaches as well as their applications such as language modeling, machine translation, question answering, document classification, and dialog systems.

## Natural language processing (major conferences: ACL, EMNLP, NAACL)

- Includes state-of-the-art deep learning-based models and tasks
- Low-level parsing
  - Tokenization, stemming
- Word and phrase level
  - Named entity recognition(NER), part-of-speech(POS) tagging, noun-phrase chunking, dependency parsing, coreference resolution
- Sentence level
  - Sentiment analysis, machine translation
- Multi-sentence and paragraph level
  - Entailment prediction, question answering, dialog systems, summarization

## Text mining(major conferences: KDD, The WebConf (formerly, WWW), WSDM, CIKM, ICWSM)

- Extract useful information and insights from text and document data
  - e.g., analyzing the trends of AI-related keywords from massive news data
- Document clustering (e.g., topic modeling)
  - e.g., clustering news data and grouping into different subjects
- Highly related to computational social science
  - e.g., analyzing the evolution of people's political tendency based on social media data

## Information retrieval (major conferences: SIGIR, WSDM, CIKM, RecSys)

- **Highly related to computational social science**
  - This area is not actively studied now
  - It has evolved into a recommendation system, which is still an active area of research

- Text data can basically be viewed as a sequence of words, and **each word can be represented as a vector** through a technique such as Word2Vec or GloVe.
- **RNN-family models** (LSTMs and GRUs), which take the sequence of these vectors of words as input, are the main architecture of NLP tasks.
- Overall performance of NLP tasks has been improved since **attention modules and Transformer models**, which replaced RNNs with self-attention, have been introduced a few years ago.
- As is the case for Transformer models, most of the advanced NLP models have been originally developed for improving machine translation tasks.

- In the early days, **customized models for different NLP tasks** had developed separately.
- Since Transformer was introduced, huge models were released by stacking its basic module, self-attention, and these models are trained with large-sized datasets through language modeling tasks, one of the **self-supervised training setting that does not require additional labels** for a particular task.
  - e.g., BERT, GPT-3 ...
- Afterwards, above models were applied to other tasks through **transfer learning**, and they outperformed all other customized models in each task.
- Currently, these models has now become essential part in numerous NLP tasks, so **NLP research become difficult with limited GPU resources**, since they are too large to train. ☹

End of Document  
Thank You.



# 2.

## Bag-of-Words

Bag-of-Words

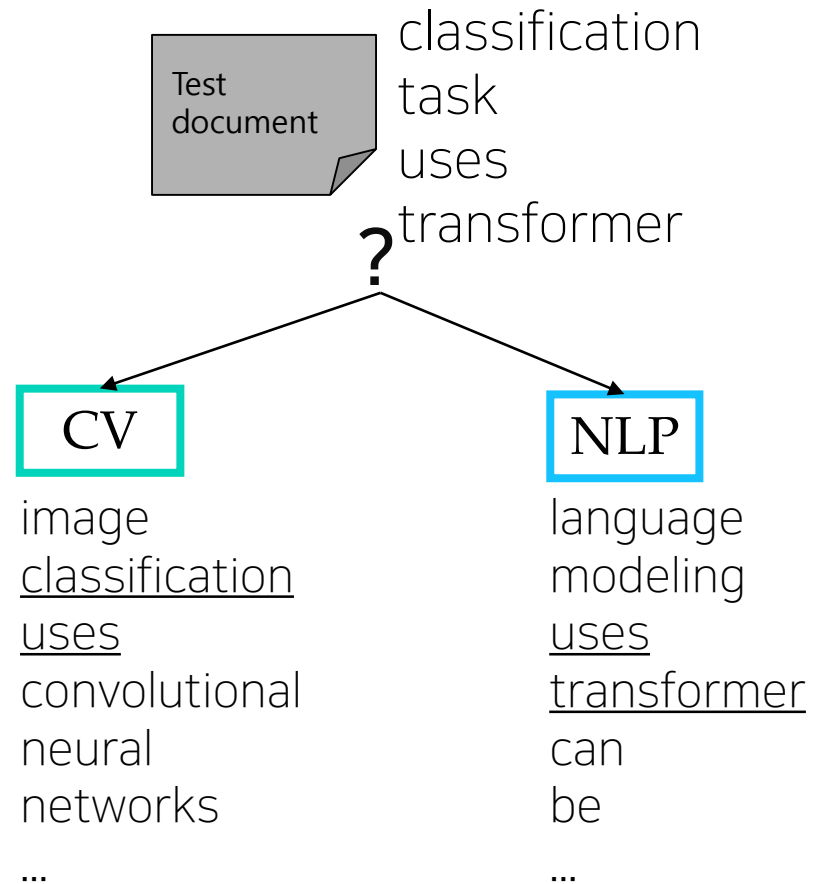
Bag-of-Words Example (NaiveBayes Classifier)

- **Step 1. Constructing the vocabulary containing unique words**
  - Example sentences: "John really really loves this movie", "Jane really likes this song"
  - Vocabulary: {"John", "really", "loves", "this", "movie", "Jane", "likes", "song"}

- Step 2. Encoding unique words to one-hot vectors
  - Vocabulary: {"John", "really", "loves", "this", "movie", "Jane", "likes", "song"}
    - John: [1 0 0 0 0 0 0 0]
    - Jane: [0 0 0 0 0 1 0 0]
    - really: [0 1 0 0 0 0 0 0]
    - likes: [0 0 0 0 0 0 1 0]
    - loves: [0 0 1 0 0 0 0 0]
    - song: [0 0 0 0 0 0 0 1]
    - this: [0 0 0 1 0 0 0 0]
    - movie: [0 0 0 0 1 0 0 0]
- For any pair of words, the distance is  $\sqrt{2}$
- For any pair of words, cosine similarity is 0

- **A sentence/document can be represented as the sum of one-hot vectors**
  - Sentence 1: "John really really loves this movie"
    - John + really + really + loves + this + movie: [1 2 1 1 1 0 0 0]
  - Sentence 2: "Jane really likes this song"
    - Jane + really + likes + this + song: [0 1 0 1 0 1 1 1]

## Bag-of-Words for Document Classification



## Bayes' Rule Applied to Documents and Classes

- For a document  $d$  and a class  $c$

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(c|d)$$

MAP is "maximum a posteriori" = most likely class

$$= \operatorname{argmax}_{c \in C} \frac{P(d|c)P(c)}{P(d)}$$

Bayes Rule

$$= \operatorname{argmax}_{c \in C} P(d|c)P(c)$$

Dropping the denominator

- **Bayes' Rule Applied to Documents and Classes**

- For a document  $d$ , which consists of a sequence of words  $w$ , and a class  $c$
- The probability of a document can be represented by multiplying the probability of each word appearing
- $P(d|c)P(c) = P(w_1, w_2, \dots, w_n|c)P(c) \rightarrow P(c) \prod_{w_i \in W} P(w_i|c)$  (by conditional independence assumption)

- **Example**

- For a document  $d$ , which consists of sequence of words  $w$ , and a class  $c$

	Doc(d)	Document (words, w)	Class (c)
Training	1	Image recognition uses convolutional neural networks	CV
	2	Transformer can be used for image classification task	CV
	3	Language modeling uses transformer	NLP
	4	Document classification task is language task	NLP
Test	5	Classification task uses transformer	?

- $P(c_{cv}) = \frac{2}{4} = \frac{1}{2}$
- $P(c_{NLP}) = \frac{2}{4} = \frac{1}{2}$



- **Example**

- For each word  $w_i$  we can calculate conditional probability for class  $c$ 
  - $P(w_k | c_i) = \frac{n_k}{n}$ , where  $n_k$  is occurrences of  $w_k$  in documents of topic  $c_i$

Word	Prob	Word	Prob
$P(w_{\text{"classification"}}   c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"classification"}}   c_{NLP})$	$\frac{1}{10}$
$P(w_{\text{"task"}}   c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"task"}}   c_{NLP})$	$\frac{2}{10}$
$P(w_{\text{"uses"}}   c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"uses"}}   c_{NLP})$	$\frac{1}{10}$
$P(w_{\text{"transformer"}}   c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"transformer"}}   c_{NLP})$	$\frac{1}{10}$

- For a test document  $d_5 = \text{"Classification task uses transformer"}$ 
  - We calculate the conditional probability of the document for each class
  - We can choose a class that has the highest probability for the document
- $P(c_{CV}|d_5) = P(c_{CV}) \prod_{w \in W} P(w|c_{CV}) = \frac{1}{2} \times \frac{1}{10} \times \frac{1}{10} \times \frac{1}{10} \times \frac{1}{10} = 0.00005$
- $P(c_{NLP}|d_5) = P(c_{NLP}) \prod_{w \in W} P(w|c_{NLP}) = \frac{1}{2} \times \frac{1}{13} \times \frac{2}{13} \times \frac{1}{13} \times \frac{1}{13} \approx 0.00003$

Word	Prob	Word	Prob
$P(w_{\text{"classification"}} c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"classification"}} c_{NLP})$	$\frac{1}{10}$
$P(w_{\text{"task"}} c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"task"}} c_{NLP})$	$\frac{2}{10}$
$P(w_{\text{"uses"}} c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"uses"}} c_{NLP})$	$\frac{1}{10}$
$P(w_{\text{"transformer"}} c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"transformer"}} c_{NLP})$	$\frac{1}{10}$

End of Document  
Thank You.

3.

## Word Embedding: Word2Vec, GloVe

What is Word Embedding?

Word2Vec

GloVe

# What is Word Embedding?

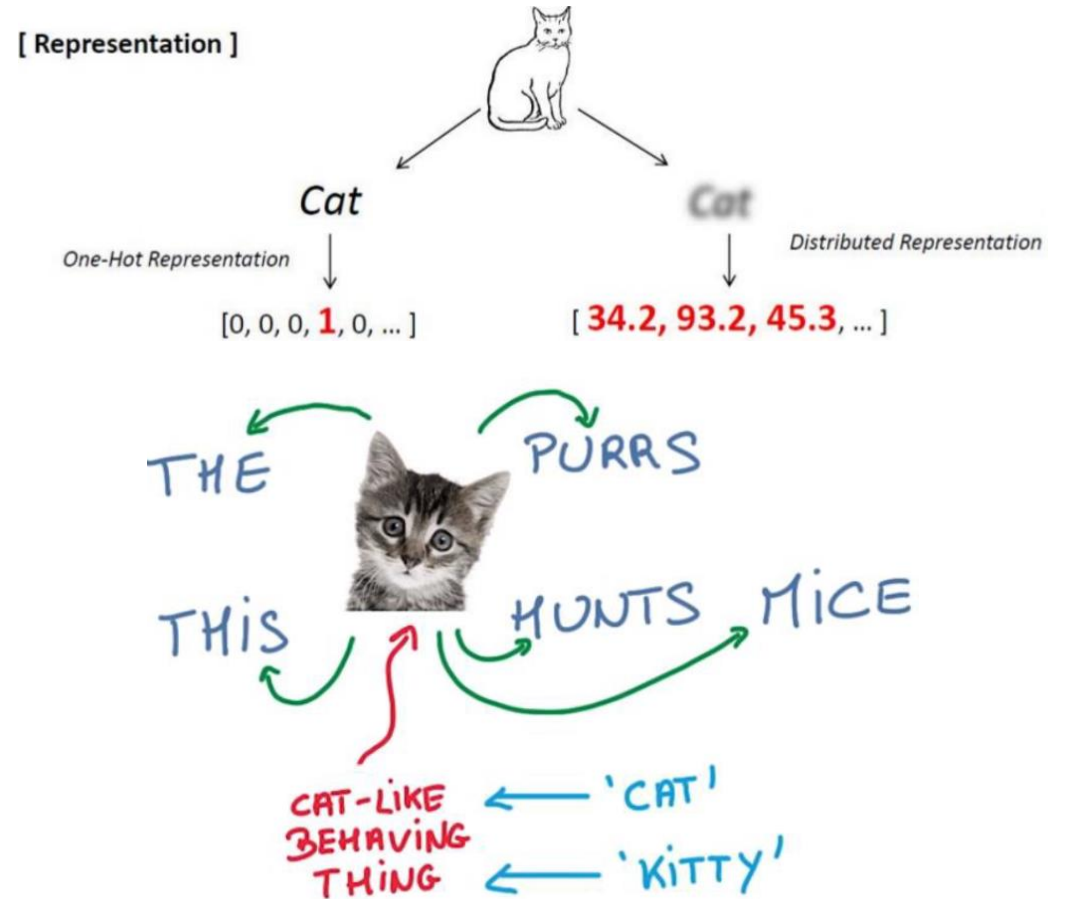
Word Embedding: Word2Vec, GloVe

- Express a word as a vector
- 'cat' and 'kitty' are similar words, so they have similar vector representations → short distance
- 'hamburger' is not similar with 'cat' or 'kitty', so they have different vector representations → far distance

# Word2Vec

Word Embedding: Word2Vec, GloVe

- An algorithm for training vector representation of a word from context words (adjacent words)
- Assumption: words in similar context will have similar meanings
- e.g.,
  - The cat purrs.
  - The cat hunts mice.

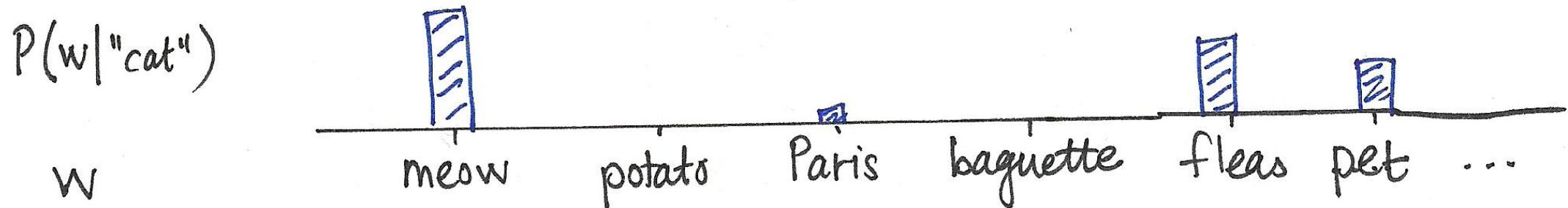


Distributed Representations of Words and Phrases and their Compositionality, NeurIPS'13

# Idea of Word2Vec

Word Embedding: Word2Vec, GloVe

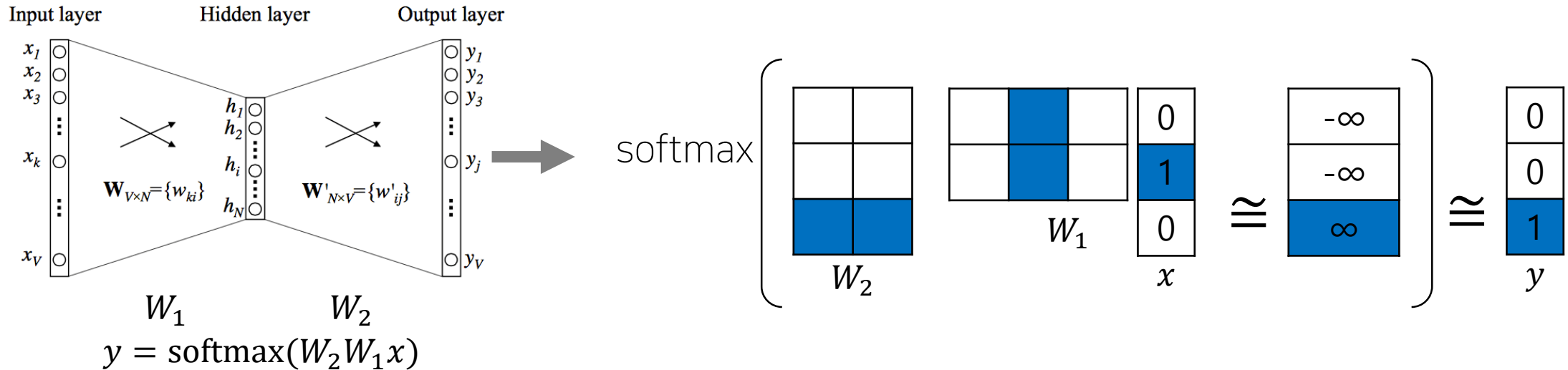
- "You shall know a word by the company it keeps" –J.R. Firth 1957
- Suppose we read the word "cat"
  - What is the probability  $P(\underline{w}|\text{cat})$  that we'll read the word  $\underline{w}$  nearby?



- Distributional Hypothesis: The meaning of "cat" is captured by the probability distribution  $P(\underline{w}|\text{cat})$

# How Word2Vec Algorithm Works

Word Embedding: Word2Vec, GloVe



- Sentence : "I study math."
- Vocabulary: {"I", "study", "math"}
- Input: "study"  $[0, 1, 0]$
- Output: "math"  $[0, 0, 1]$
- Columns of  $W_1$  and rows of  $W_2$  represent each word
- E.g., 'study' vector : 2<sup>nd</sup> column in  $W_1$ , 'math' vector : 3<sup>rd</sup> row in  $W_2$ .
- The 'study' vector in  $W_1$  and the 'math' vector in  $W_2$  should have a high inner-product value.

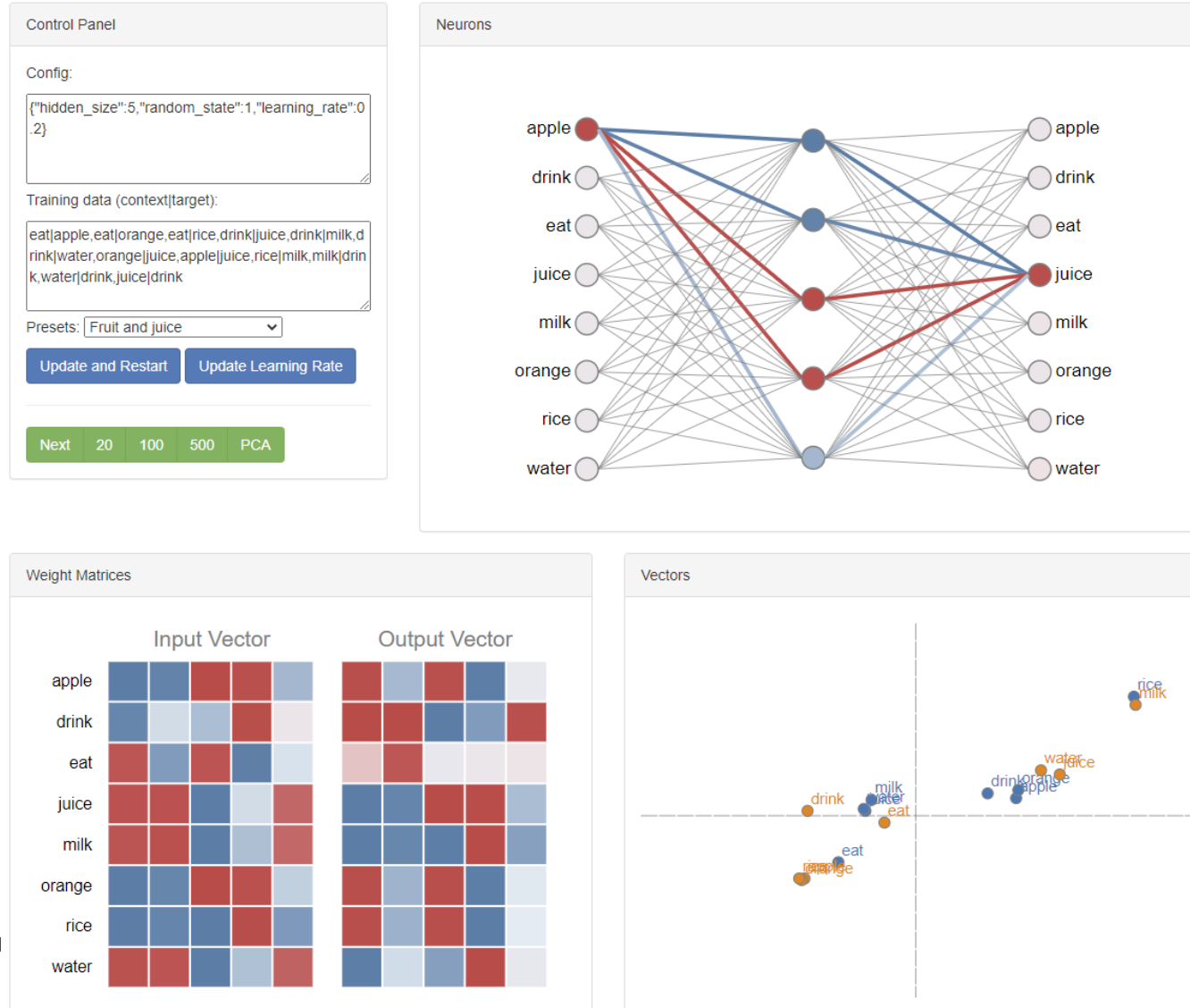


# Another Example

Word Embedding: Word2Vec, GloVe

## wevi: word embedding visual inspector

Everything you need to know about this tool - Source code



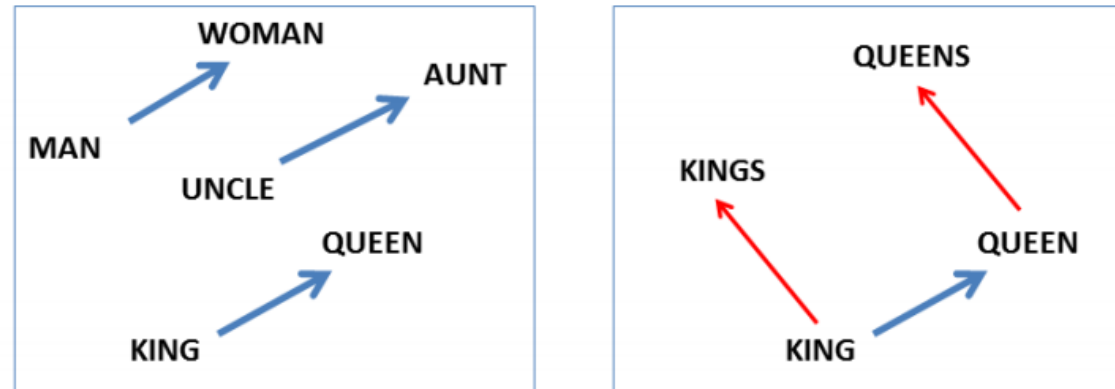
<https://ronxin.github.io/wevi/>

- A vector representation of 'eat' in  $W_1$  has similar pattern with vectors of 'apple', 'orange', and 'rice' in  $W_2$
- When the input is 'eat', the model can predict 'apple', 'orange', or 'rice' for output, because the vectors have high inner product values

# Property of Word2Vec

Word Embedding: Word2Vec, GloVe

- The word vector, or the relationship between vector points in space, represents the relationship between the words.
- The same relationship is represented as the same vectors.



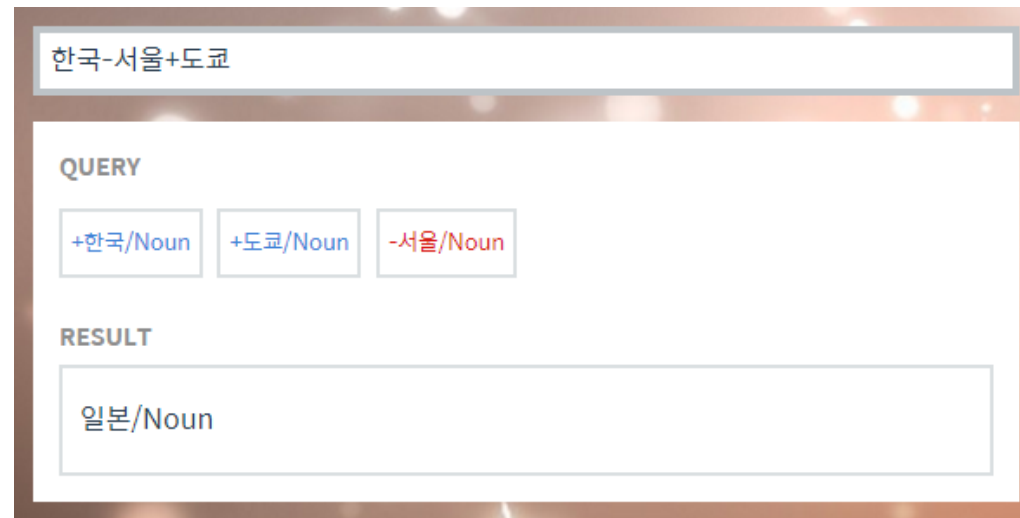
(Mikolov et al., NAACL HLT, 2013)

- e.g.,
- $\text{vec}[\text{queen}] - \text{vec}[\text{king}] = \text{vec}[\text{woman}] - \text{vec}[\text{man}]$

# Property of Word2Vec – Analogy Reasoning

Word Embedding: Word2Vec, GloVe

- Korean Word2Vec : <http://w.elnn.kr/search>



The screenshot shows a web interface for Korean Word2Vec analogy reasoning. At the top, there is a text input field containing the query "한국-서울+도쿄". Below this, the interface is divided into two main sections: "QUERY" and "RESULT". The "QUERY" section contains three buttons: "+한국/Noun" (blue text), "+도쿄/Noun" (blue text), and "-서울/Noun" (red text). The "RESULT" section contains a single text box displaying the answer "일본/Noun".

# Property of Word2Vec – Analogy Reasoning

Word Embedding: Word2Vec, GloVe

- More examples: <http://wonjaekim.com/archives/50>

	데모	<a href="http://w.elnn.kr/">http://w.elnn.kr/</a>
버락_오바마-미국+러시아	블라디미르/Noun_푸틴/Noun	-
버락_오바마-미국+스타워즈	아나킨/Noun_스카이워커/Noun	-
아카라카-연세대학교+고려대학교	입실렌티/Noun	입실렌티/Noun
아이폰-휴대폰+노트북	아이패드/Noun	아이패드/Noun
컴퓨터공학-자연과학+인문학	법학/Noun	게임학/Noun
플레이스테이션-소니+마이크로소프트	엑스박스/Noun_360/Number	MSX/Alpha
한국-서울+파리	프랑스/Noun	프랑스/Noun

- Example: <https://github.com/dhammack/Word2VecExample>
- Word intrusion detection
  - staple hammer saw drill
  - math shopping reading science
  - rain snow sleet sun
  - eight six seven five three owe nine
  - breakfast cereal dinner lunch
  - england spain france italy greece germany portugal australia

- **Word2Vec improves performances in most areas of NLP**
  - Word similarity
  - Machine translation
  - Part-of-speech (PoS) tagging
  - Named entity recognition (NER)
  - Sentiment analysis
  - Clustering
  - Semantic lexicon building

# Application of Word2Vec – Machine Translation

Word Embedding: Word2Vec, GloVe

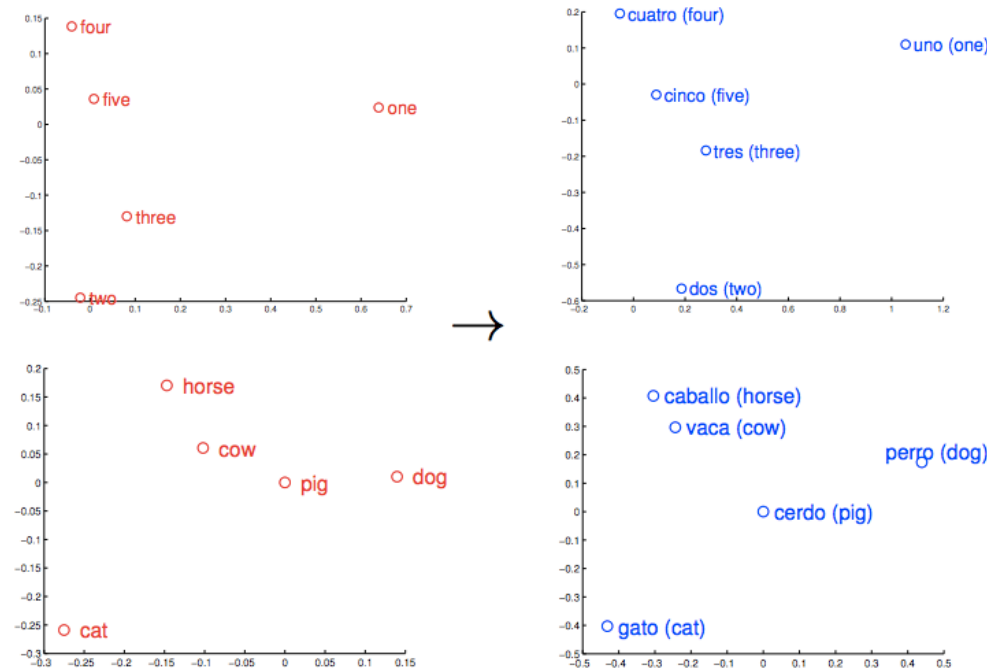
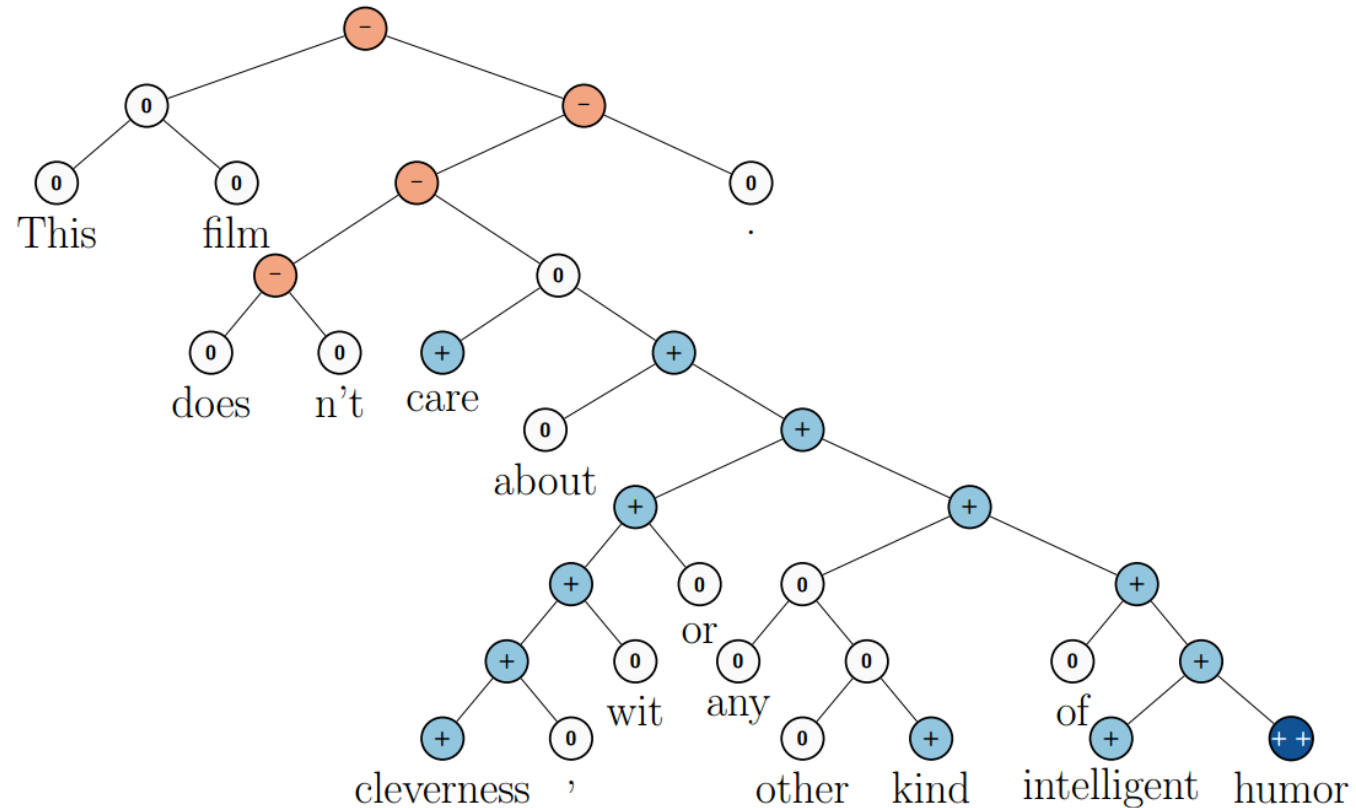


Figure 1: Distributed word vector representations of numbers and animals in English (left) and Spanish (right). The five vectors in each language were projected down to two dimensions using PCA, and then manually rotated to accentuate their similarity. It can be seen that these concepts have similar geometric arrangements in both spaces, suggesting that it is possible to learn an accurate linear mapping from one space to another. This is the key idea behind our method of translation.

# Application of Word2Vec – Sentiment Analysis

## Word Embedding: Word2Vec, GloVe





# Application of Word2Vec – Image Captioning

Word Embedding: Word2Vec, GloVe

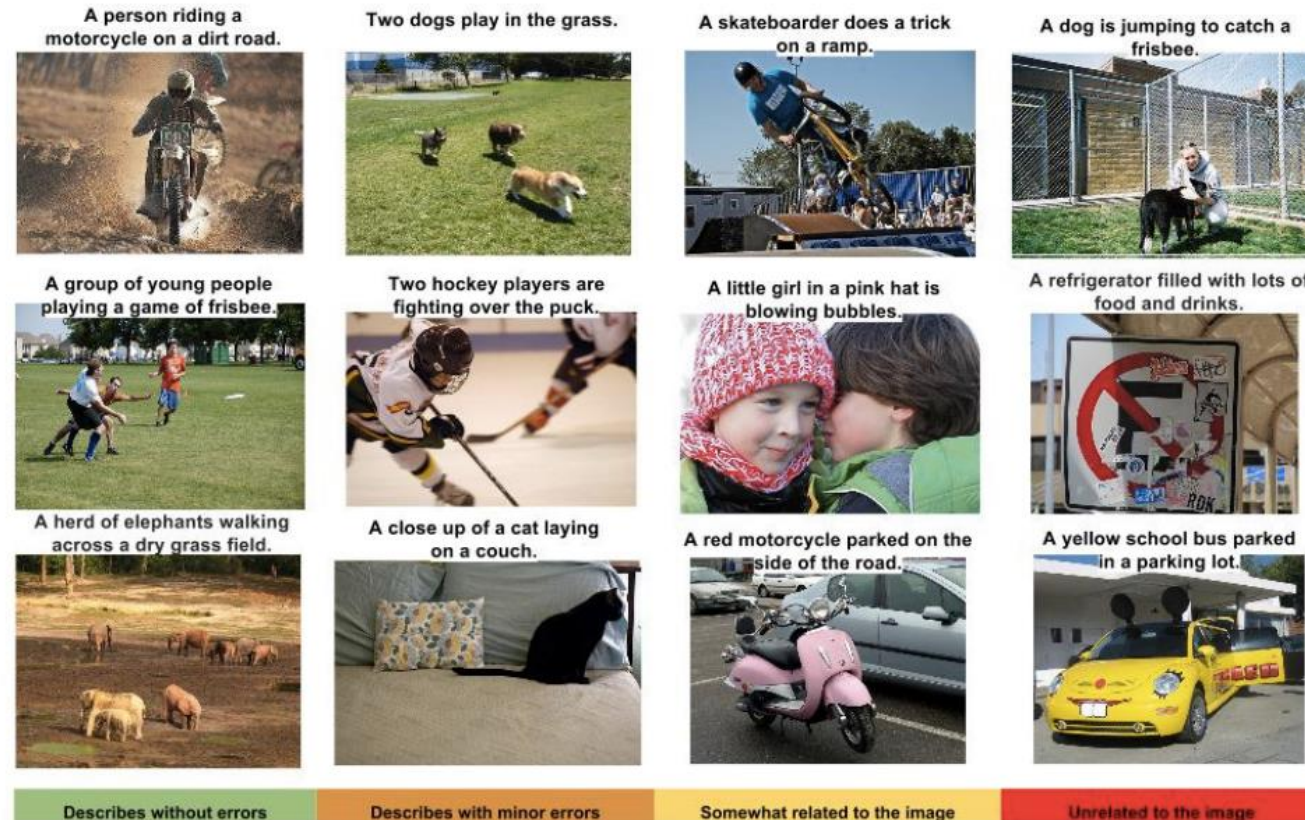
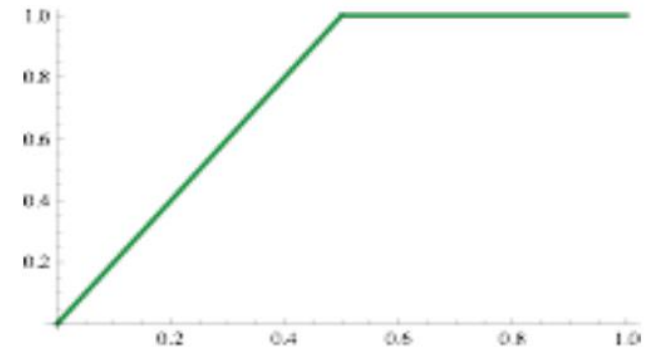


Figure 5. A selection of evaluation results, grouped by human rating.

## GloVe: Global Vectors for Word Representation

- Rather than going through each pair of an input and an output words, it first computes the co-occurrence matrix, to avoid training on identical word pairs repetitively.
- Afterwards, it performs matrix decomposition on this co-occurent matrix.

$$J(\theta) = \frac{1}{2} \sum_{i,j=1}^W f(P_{ij})(u_i^T v_j - \log P_{ij})^2 \quad f \sim$$



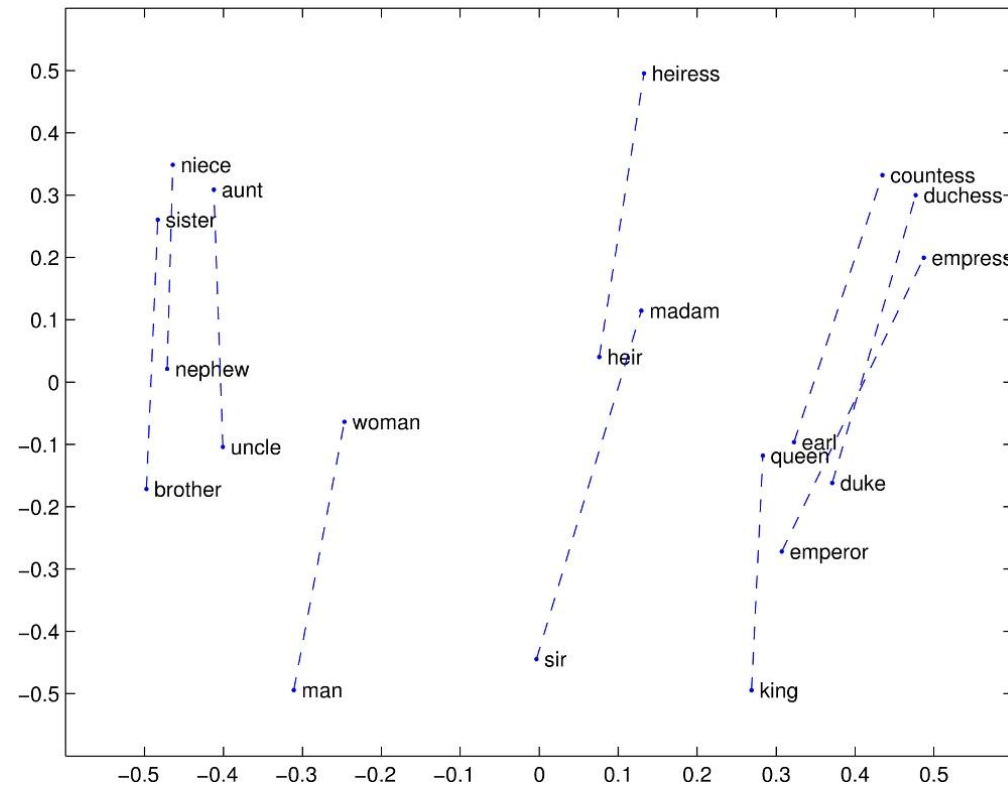
- Fast training
- Works well even with a small corpus

GloVe: Global Vectors for Word Representation, EMNLP'14

# Property of GloVe – Linear Substructure

Word Embedding: Word2Vec, GloVe

- man - woman

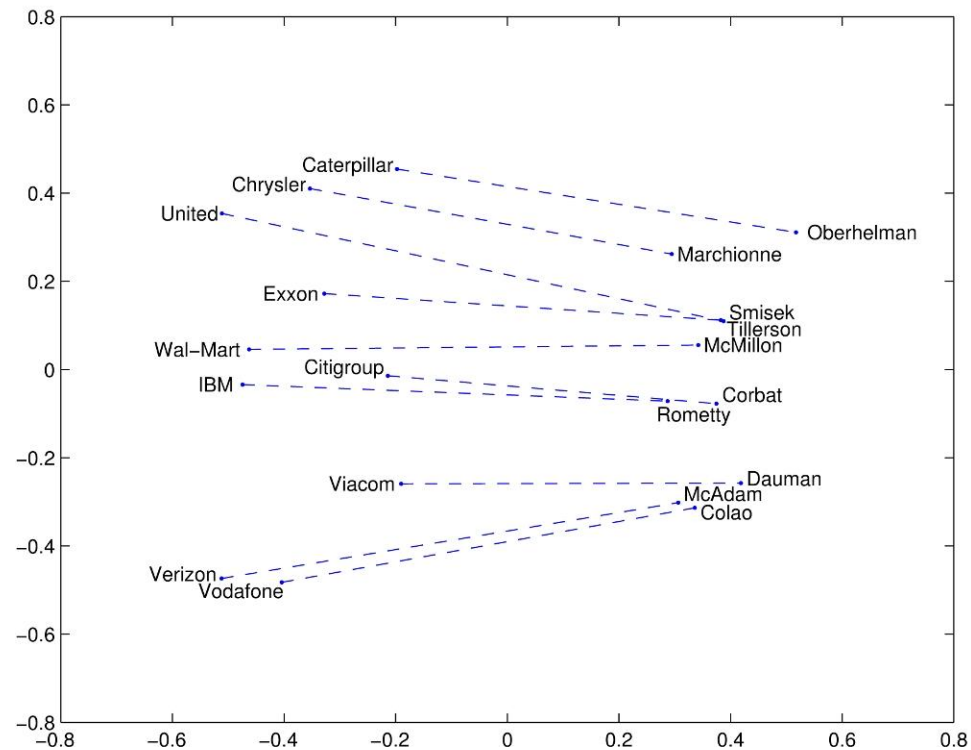


Distributed Representations of Words and Phrases and their Compositionality, NeurIPS'13

# Property of GloVe – Linear Substructure

Word Embedding: Word2Vec, GloVe

- company – ceo

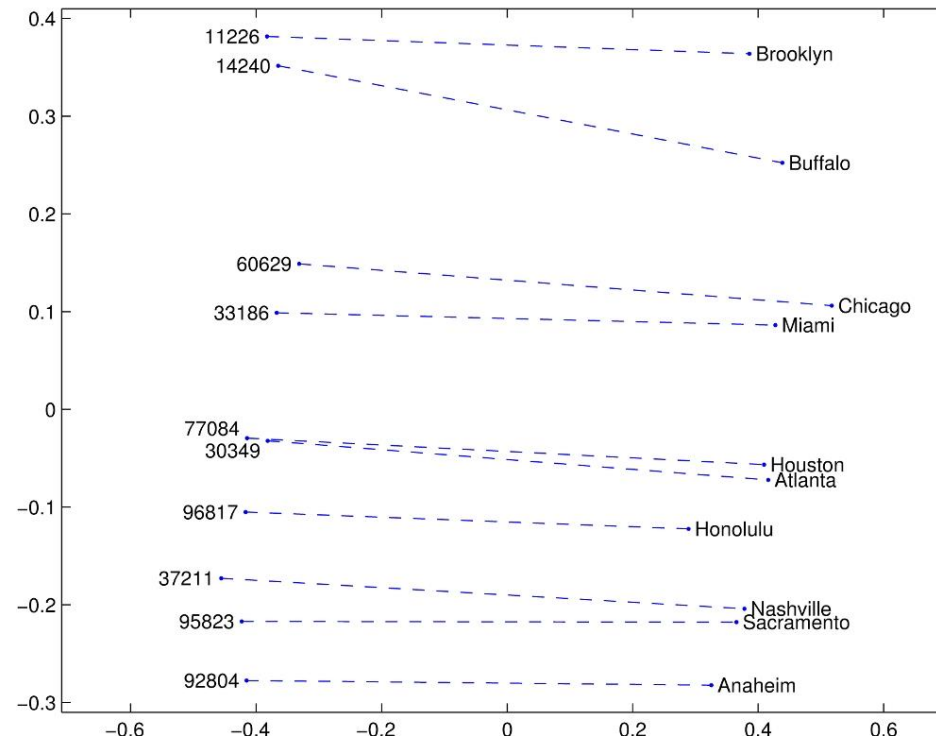


Distributed Representations of Words and Phrases and their Compositionality, NeurIPS'13

# Property of GloVe – Linear Substructure

Word Embedding: Word2Vec, GloVe

- city – zip code

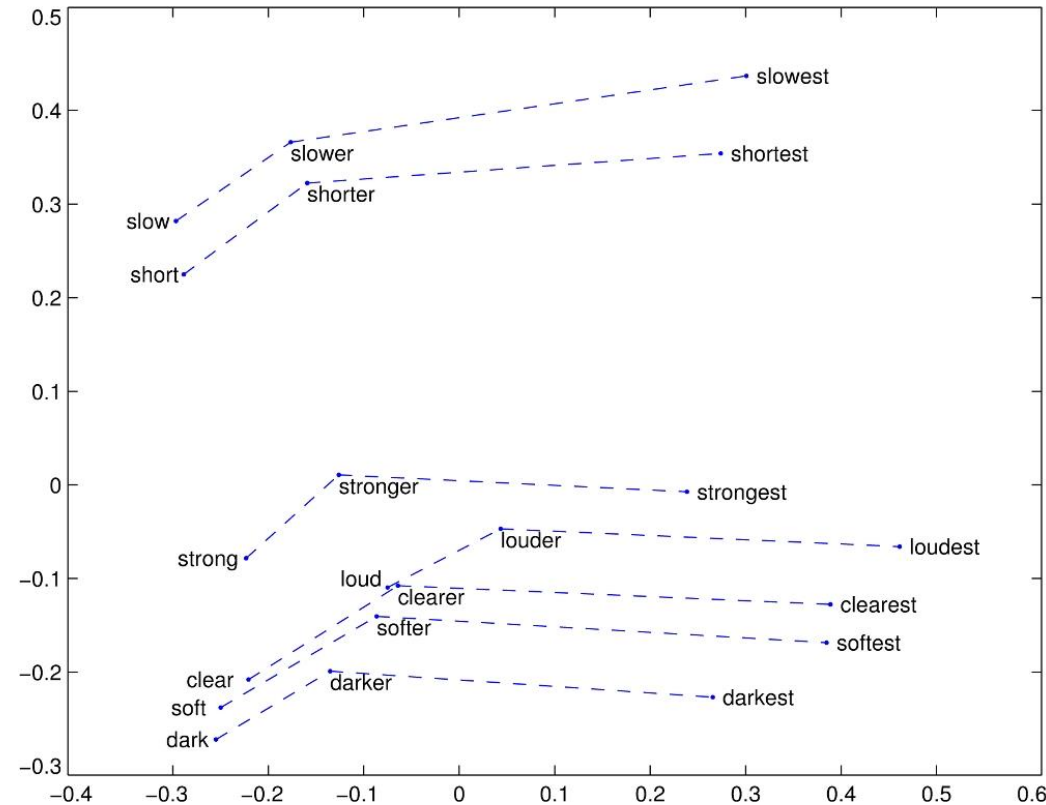


Distributed Representations of Words and Phrases and their Compositionality, NeurIPS'13

# Property of GloVe – Linear Substructure

Word Embedding: Word2Vec, GloVe

- comparative – superlative



Distributed Representations of Words and Phrases and their Compositionality, NeurIPS'13

# GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning

## Introduction

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

## Getting started (Code download)

- Download the latest [latest code](#) (licensed under the [Apache License, Version 2.0](#)).  
Look for "Clone or download"
- Unpack the files: `unzip master.zip`
- Compile the source: `cd GloVe-master && make`
- Run the demo script: `./demo.sh`
- Consult the included README for further usage details, or ask a [question](#)

<https://nlp.stanford.edu/projects/glove/>

## Download pre-trained word vectors

- Pre-trained word vectors. This data is made available under the [Public Domain Dedication and License](#) v1.0 whose full text can be found at: <http://www.opendatacommons.org/licenses/pddl/1.0/>.
  - [Wikipedia 2014](#) + [Gigaword 5](#) (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): [glove.6B.zip](#)
  - Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): [glove.42B.300d.zip](#)
  - Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): [glove.840B.300d.zip](#)
  - Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): [glove.twitter.27B.zip](#)
- Ruby [script](#) for preprocessing Twitter data

## Citing GloVe

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. [GloVe: Global Vectors for Word Representation](#). [pdf] [bib]

# References

---

- Word2Vec
  - Distributed Representations of Words and Phrases and their Compositionality, NeurIPS'13
- GloVe
  - GloVe: Global Vectors for Word Representation, EMNLP'14



End of Document  
Thank You.