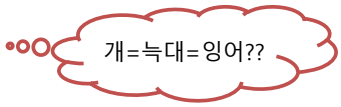
KU KONKUK UNIVERSITY

Text Representation

이산 표현 (Discrete Representation)

- 원-핫 인코딩 (one-hot encoding)
  - 단어를 벡터로 표현하는 가장 간단한 방법
  - 단어 사전(dictionary)을 구성하고 해당 단어를 1로, 그 밖의 단어는 0으로 표현
    - 사전: 개, 고양이, 늑대, 사자, 송어, 잉어
    - 단어: 개 [1, 0, 0, 0, 0, 0], 고양이 [0, 1, 0, 0, 0, 0], 늑대 [0, 0, 1, 0, 0, 0]
- 원-핫 인코딩의 한계
  - 대용량 메모리 필요
    - 사전의 크기: 음성 인식 20K, 구문 분석: 50K, 대용량 사전: 500K, 구글 1T 말뭉치: 13M
  - 유사성 비교 불가능
    - Sim(개, 늑대) = AND([1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0]) = 0
    - Sim(개, 잉어) = AND([1, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 1]) = 0



분산 표현 (Distributed Representation)

- 분산 표현
  - 단어를 문맥에 기반하여 표현하는 방법
  - 비슷한 문맥에서 등장하는 단어는 비슷한 의미를 가질 것이라는 가정에서 출발

신종 **코로나** 바이러스 집단 감염증이 일상생활을 통해 지속 확산되고 있다.

부산 수영구 댄스 동호회발 신종 **코로나** 바이러스 집단 감염이 목욕탕으로 퍼지는 등 확산되고 있다.

코로나? 신종, 바이러스, 집단, 감염, 확산, 일상, 생활, …, 목욕탕

프랑스는 지금까지 북부도시 릴에서 2명의 **메르스** 바이러스 감염환자가 발생했다.

메르스? 바이러스, 감염, 프랑스, 지금, …, 환자, 발생

Sim(코로나, 메르스) → AND(코로나, 메르스) ≠0

공기 행렬 (Co-Occurrence Matrix)

[예문] I like deep learning. I like NLP. I enjoy flying.

Window size: 1

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Sim(NLP, learning) = ?

그림 출처: Kira Radinsky 교수 강의자료

Edited by Harksoo Kim

코사인 유사도 (Cosine Similarity)

- 코사인 유사도
  - 길이로 정규화된 내적을 바탕으로 두 벡터 사이의 유사도를 측정하는 척도

두 벡터 요소의 구성이 비슷하면 큰 값을 가짐

$$\vec{X} \cdot \vec{Y} = |\vec{X}||\vec{Y}| \cos(\theta)$$

$$\cos(\vec{X}, \vec{Y}) = \frac{\vec{X} \cdot \vec{Y}}{|\vec{X}||\vec{Y}|}$$

$$\vec{X} \cdot \vec{Y} = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

$$|\vec{X}| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

$$|\vec{Y}| = \sqrt{y_1^2 + y_2^2 + \cdots + y_n^2}$$

두 벡터가 직각: 0
두 벡터가 동일: 1

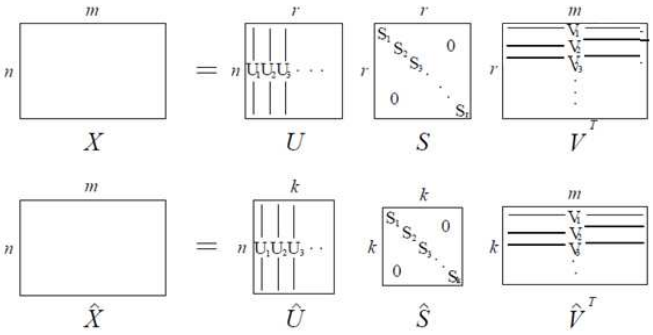
cos(NLP, learning) = 1/2 = 0.5

```
import torch

NLP = torch.FloatTensor([0,1,0,0,0,0,0,1])
learning = torch.FloatTensor([0,0,0,1,0,0,0,1])

print(torch.cosine_similarity(NLP, learning, dim=0))

tensor(0.5000)
```

Edited by Harksoo Kim

Problems with Co-Occurrence Vectors

- 차원의 저주 (Curse of dimensionality)
  - 차원이 증가하면서 학습데이터의 수가 차원의 수보다 적어서 성능이 저하되는 현상 → 희소 데이터 문제 (sparse data problem)
- 특이값 분해 (SVD; Singular Value Decomposition)
  - 행렬을 특정한 구조로 분해하는 방식



$\hat{X}$ is the best rank k approximation to X, in terms of least squares

그림 출처: Kira Radinsky 교수 강의자료

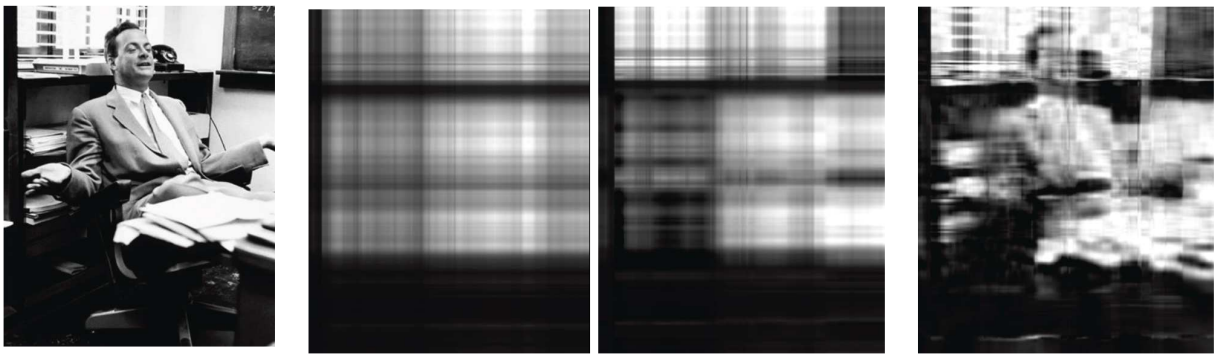
Effect of SVD

Original

1st Singular

1st & 2nd Singular

.....



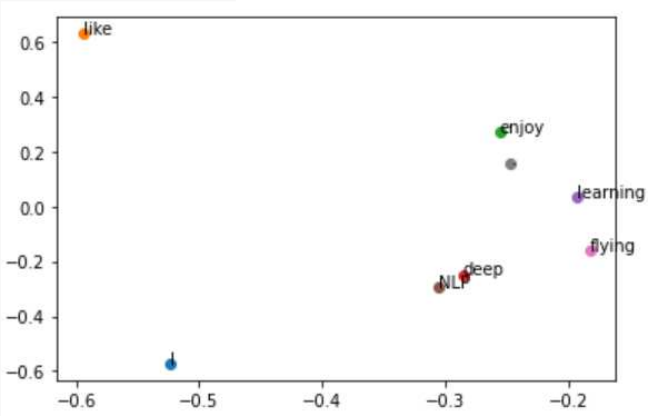
$R_i = c_i \cdot SV_1$ $R_i = c_i \cdot SV_1 + b_i \cdot SV_2$

그림 출처: Kira Radinsky 교수 강의자료

Simple SVD Word Vectors in Python

[예문] I like deep learning. I like NLP. I enjoy flying.

```
import numpy as np
import matplotlib.pyplot as plt

dic= ["I", "like", "enjoy", "deep", "learning", "NLP", "flying", "." ]
X = np.array([[0,2,1,0,0,0,0,0],
              [2,0,0,1,0,1,0,0],
              [1,0,0,0,0,0,1,0],
              [0,1,0,0,1,0,0,0],
              [0,0,0,1,0,0,0,1],
              [0,1,0,0,0,0,0,1],
              [0,0,1,0,0,0,0,1],
              [0,0,0,0,1,1,1,0]])
U, S, Vt =np.linalg.svd(X, full_matrices=False)

for i in range(len(dic)):
  plt.scatter(U[i,0], U[i,1])
  plt.text(U[i,0], U[i,1], dic[i])

plt.show()
```



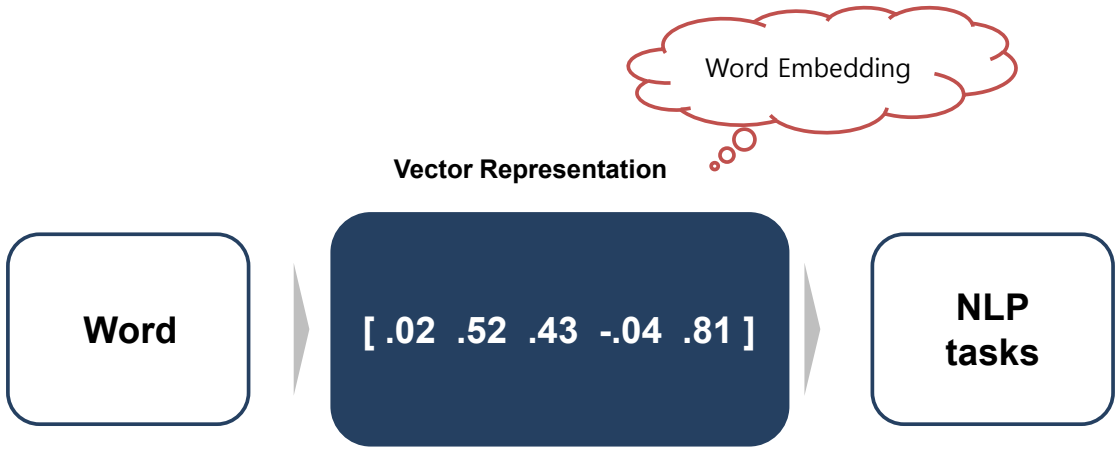
참고: Kira Radinsky 교수 강의자료

From SVD To Word2Vec

- SVD의 문제점
  - 계산에 너무 오랜 시간이 소요됨
    - n*m 행렬 계산 → $O(mn^2)$
  - 유연성이 떨어짐
    - 새로운 단어나 문서가 추가될 경우에 SVD를 처음부터 다시 수행
- 해결 방안
  - Learning representations by back-propagation errors (Rumelhart et al., 1986)
  - Neural probabilistic language model (Bengio et al., 2003)
  - NLP from Scratch (Collobert & Weston,2008)
  - Word2Vec (Mikolov et al., 2013)
    - Instead of capturing co-occurrence counts directly: Predict surrounding words of every word

Distributed Representation (Again)

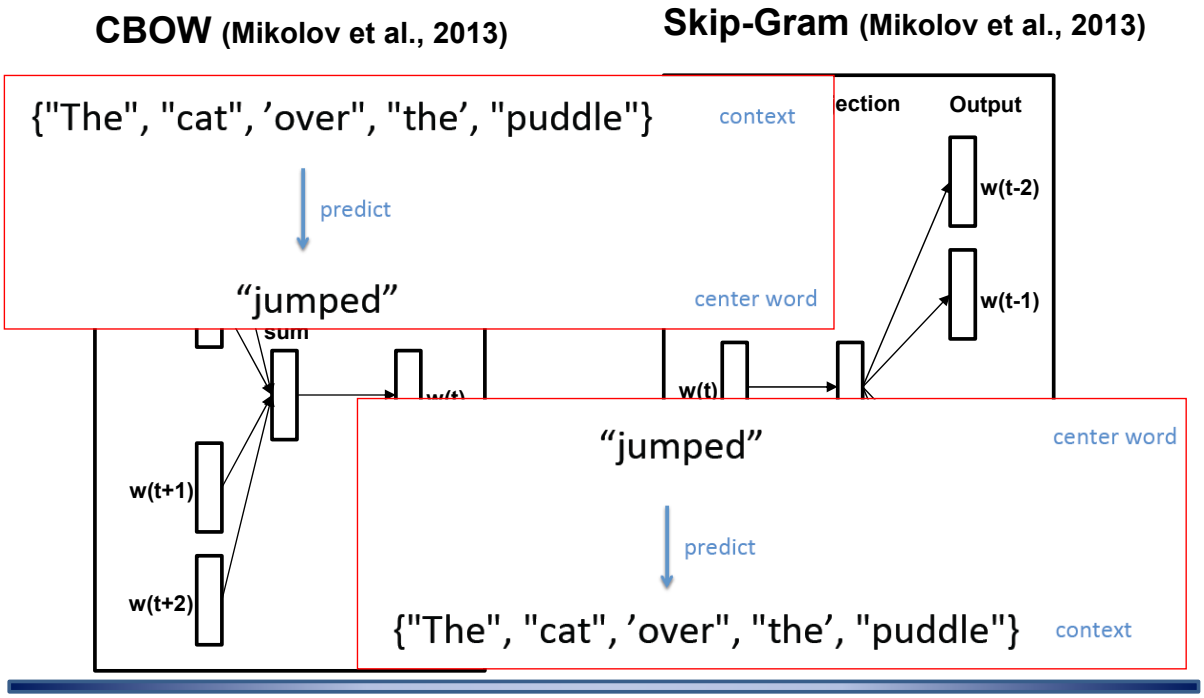
고차원 one-hot 벡터 → 저차원 실수 벡터
기본 아이디어: 비슷한 문맥에서 나온 단어는 비슷한 뜻을 가짐!



- **Word2vec (Mikolov, 2013)**
- **Glove (Pennington, et al., 2014)**

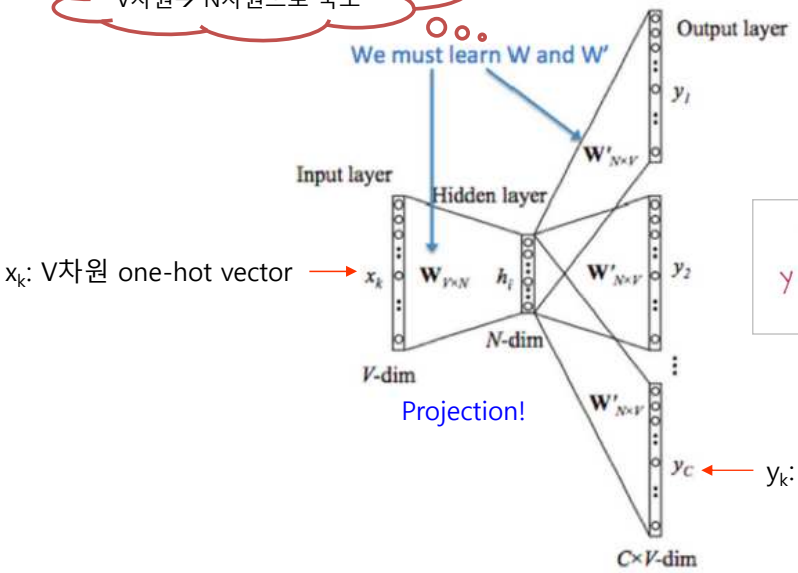
그림 출처: NLP 겨울학교(오혜연 교수)

Word2Vec



How will it work?

N차원 word embedding:
V차원→ N차원으로 축소



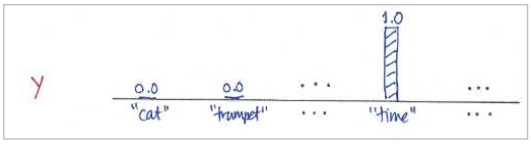
x_k: V차원 one-hot vector

Cost function? Cross entropy

$$H(\hat{y}, y) = -\sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$$

WHY?

$$H(\hat{y}, y) = -y_i \log(\hat{y}_i)$$



y_k: V차원 one-hot vector

그림 출처: Kira Radinsky 교수 강의자료

SoftMax

Cost function? Cross entropy

$$H(\hat{y}, y) = -\sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$$

WHY?

$$H(\hat{y}, y) = -y_i \log(\hat{y}_i)$$

Softmax: Linear regression generalization to multi-class

$$h_\theta(x) = \begin{bmatrix} P(y=1|x;\theta) \\ P(y=2|x;\theta) \\ \vdots \\ P(y=K|x;\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x)} \begin{bmatrix} \exp(\theta^{(1)\top} x) \\ \exp(\theta^{(2)\top} x) \\ \vdots \\ \exp(\theta^{(K)\top} x) \end{bmatrix}$$

Output: a K-dimensional vector (whose elements sum to 1)

확률로 만들려면?
V만큼의 계산 필요!

- Hierarchical SoftMax
- Negative Sampling
- Sub-Sampling Frequent Words

GloVe (Global Vectors)

Main Insight: Ratio of co-occurrence probabilities can encode meaning

Prediction → Probability

→ **Use global information (co-occurrence over corpus) while learning word vectors**

	x = solid	x = gas	x = water	x = random
$P(x\|\text{ice})$	large	small	large	small
$P(x\|\text{steam})$	small	large	large	small
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$	large	small	~1	~1

Solid가 문맥으로 주어졌을 때, ice와 steam의 비율이 크도록 학습!

Main Insight of GloVe

- Ratio of co-occurrence probabilities can encode meaning!

How can we capture this behavior in the word vector space?

Log-bilinear model: $w_i \cdot w_j = \log P(i|j)$

Vector differences: $w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$

x=solid, a=ice, b=steam

Think: a = "ice", b = "steam"

The training objective of GloVe is to **learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence**.

Object Function of GloVe

그림 출처: Kira Radinsky 교수 강의자료

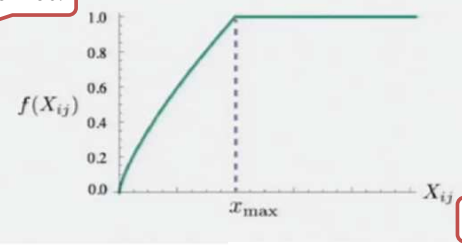
The log-bilinear model

$$J = \frac{1}{2}\sum_{ij} f(P_{ij})\left(w_i \cdot \tilde{w}_j - \log P_{ij}\right)^2$$

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

No Prob.

- X - cooccurrence matrix
- w - word vectors
- $\tilde{w}$ - context word vectors
- b - bias
- $\tilde{b}$ - bias



3/4 100

$$f(x) = \begin{cases} (x/x_{\max})^{\alpha} & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

Fast training, scalable to huge corpora,
Good performance even with small corpus, and small vectors

From One-hot Rep. to Distributed Rep.

- fastText (by facebook ← word2vec by google)
  - 부분 단어(subword)로 학습하여 노이즈에 강함
- GloVe (by stanford)
  - 동시 등장 확률을 함께 학습

질의응답

Homepage: http://nlp.konkuk.ac.kr
E-mail: nlpdrkim@konkuk.ac.kr

Edited by Harksoo Kim