



학부연구생 김지현

CONTENT

밑바닥부터 시작하는 딥러닝2

수학과 파이썬 복습

1 벡터와 행렬, 벡터의 내적과 행렬의 곱, 행렬 형상 확인

자연어와 단어의 분산 표현

2 자연어 처리란, 시소러스, 통계 기반 기법

자연어처리 최근 동향

3 구글의 Dialogflow 소개

1 수학과 파이썬 복습

(1) 벡터와 행렬

그림 1-1 벡터와 행렬의 예

$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \quad \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} \xrightarrow{\text{행}} \begin{matrix} \text{연} \\ \text{결} \end{matrix}$$

그림 1-2 벡터의 표현법

$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \quad \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

* Numpy 라이브러리로 파이썬에서 행렬 표현

(2) 벡터의 내적과 행렬의 곱

: 두 벡터가 얼마나 같은 방향을 향하고 있는지를 나타낸다.

$$\mathbf{x} \cdot \mathbf{y} = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

(3) 행렬의 곱과 형상 확인

그림 1-6 형상 확인: 행렬의 곱에서는 대응하는 차원의 원소 수를 일치시킨다.

A B = C

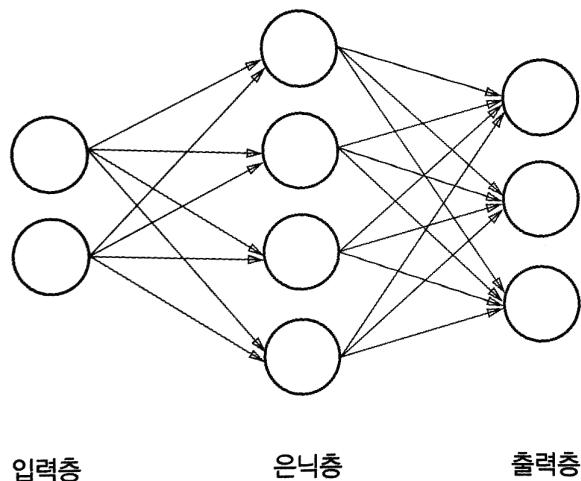
형상 : 3×2 2×4 3×4

일치시킨다.

참고: '100 numpy exercise' 사이트

신경망 (완전연결계층)

그림 1-7 신경망의 예



신경망이 수행하는 계산의 수식

$$h_1 = x_1 w_{11} + x_2 w_{21} + b_1$$

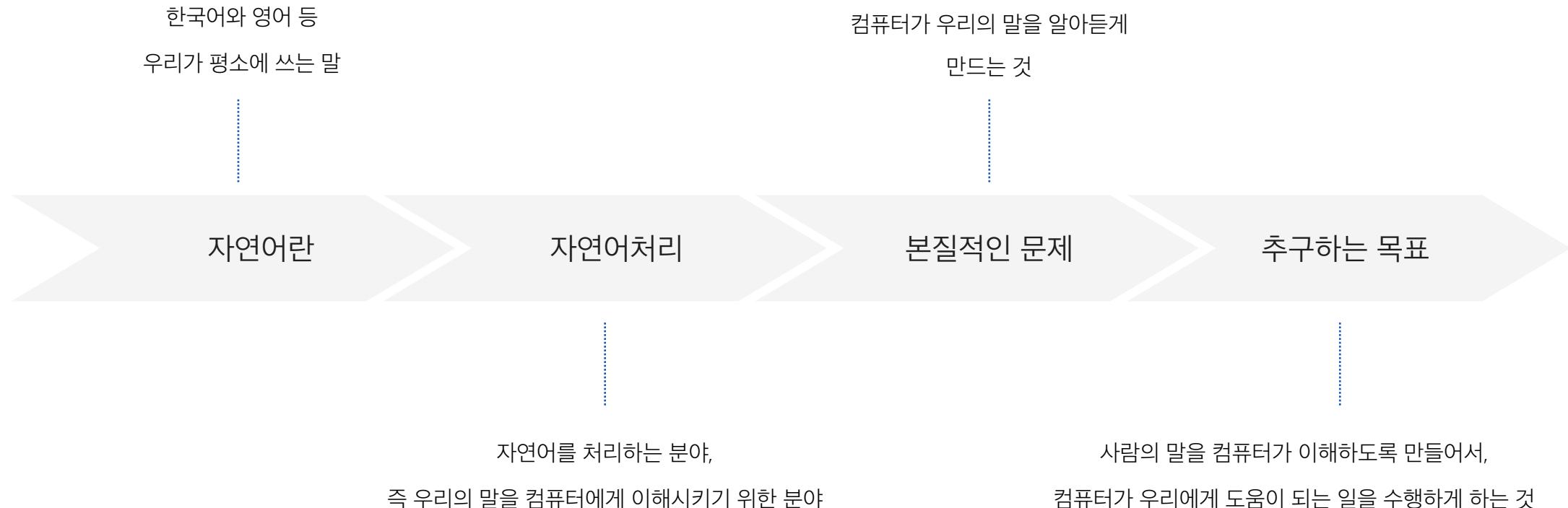
은닉층의 뉴런들

h1: 은닉층 중 첫 번째 뉴런
 (x_1, x_2) : 입력층의 데이터
 w11, w21 : 가중치
 b1 : 편향

- (1) ‘가중치의 합’으로 계산
 - (2) 행렬의 곱으로 한꺼번에 계산

$$(h_1, h_2, h_3, h_4) = (x_1, x_2) \begin{pmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \end{pmatrix} + (b_1, b_2, b_3, b_4)$$

2 자연어 처리 개요



Programming Language



컴퓨터가 이해할 수 있는 언어?

'프로그래밍 언어'나 '마크업 언어' 이러한 언어들은 모든 코드의 의미를 고유하게 해석할 수 있도록 문법이 정의되어 있다. 컴퓨터는 정해진 규칙에 따라서 코드를 해석하기 때문에 컴퓨터가 이해할 수 있는 일반적인 프로그래밍 언어는 기계적이고 고정적이다. 즉 딱딱한 언어이다.

Natural Language

영어나 한국어는?

반면 영어나 한국어는 부드러운 언어이다. 똑같은 의미의 문장도 여러 형태로 표현 가능하고, 문장의 뜻이 애매할 수 있으며 의미나 형태가 유연하다. 세월이 흐르면서 새로운 말이나 새로운 의미가 생겨나고 있던 것이 사라지기도 하는데 이는 자연어는 부드럽기 때문이다.

Now what

컴퓨터에게 자연어를 이해시키려면?

컴퓨터에게 자연어를 이해시키기란 평범한 방법으로 매우 어려운 도전이다. 만약 그 난제를 해결할 수 있다면 수많은 사람들에게 컴퓨터에게 일을 시킬 수 있다.

대표적인 예 - 검색 엔진, 기계 번역, 질의응답 시스템, IME(입력기 전환), 문장 자동요약, 감정분석

2 자연어와 단어의 분산 표현

컴퓨터에게 **단어의 의미** 이해시키는 것이 중요하다

단어의 의미를 잘 파악하는 표현 방법 3가지 기법

- 시소러스를 활용한 기법
- 통계 기반 기법
- 추론 기반 기법(word2vec)

Reference



Lecture 1 | Natural Language Processing with Deep Learning

564,637 views

 4.5K  73  SHARE  SAVE ...



Stanford University School of Engineering
Published on Apr 3, 2017

SUBSCRIBED 99K



링크: https://www.youtube.com/watch?v=OQQ-W_63UgQ&list=PL-1aZsQsQJOS7dv4EMBq_tDMZ9re6dY2B

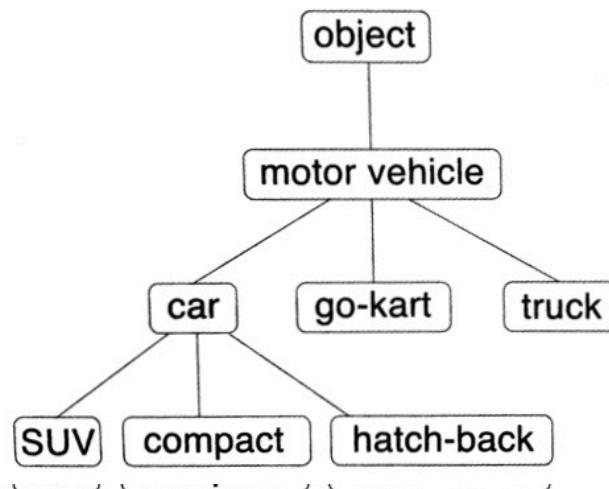
2 자연어와 단어의 분산 표현

(1) 시소러스

그림 2-1 동의어의 예: "car", "auto", "automobile" 등은 "자동차"를 뜻하는 동의어다.

car = auto automobile machine motorcar

그림 2-2 단어들을 의미의 상 · 하위 관계에 기초해 그래프로 표현한다(문헌 [14]를 참고하여 그림).



사람이 직접 단어의 의미를 정의하는 방식
(표준국어대사전처럼 시소러스 형태의 사전)

시소러스란? 유의어 사전

가장 유명한 시소러스는 WordNet

그러나 문제점들.. 多

2 자연어와 단어의 분산 표현

(2) 통계 기반 기법

말뭉치(corpus) 이용

: 말뭉치란 대량의 텍스트 데이터이다.



단어의 분산 표현

'단어'도 벡터로 표현할 수 있을까?



분포 가설(distributional hypothesis)

단어의 의미는 주변 단어에 의해 형성된다

2 자연어와 단어의 분산 표현

(2) 통계 기반 기법

단어 자체는 의미가 없고 그 단어가 사용된 '**맥락(context)**'이 의미를 형성한다.

통계 기반 기법은 맥락에 주목한다

그림 2-3 윈도우 크기가 2인 ‘맥락’의 예. 단어 “goodbye”에 주목한다면, 그 좌우의 두 단어(총 네 단어)를 맥락으로 이용한다.

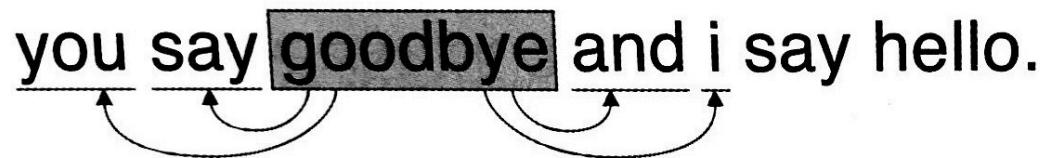


그림 2-4 단어 “you”의 맥락을 세어본다.



‘맥락’이란 특정 단어를 중심에 둔 그 주변 단어를 말하고
맥락의 크기를 (주변 단어를 몇 개나 포함할지)를 ‘**윈도우 크기**(window size)’라고 한다

2 자연어와 단어의 분산 표현

그림 2-5 단어 “you”의 맥락에 포함되는 단어의 빈도를 표로 정리한다.

	you	say	goodbye	and	i	hello	.
you	0	1	0	0	0	0	0

“you”라는 단어를 [0,1,0,0,0,0,0]이라는 벡터로 표현

마찬가지로 say에 대해서도
"say"라는 단어는 벡터 [1,0,1,0,1,1,0]

그림 2-6 단어 "say"의 맥락에 포함되는 단어의 빈도를 표로 정리한다.

you **say** goodbye and i **say** hello .



	you	say	goodbye	and	i	hello	.
say	1	0	1	0	1	1	0

동시 발생 행렬!

co-occurrence matrix

그림 2-7 모든 단어 각각의 맥락에 해당하는 단어의 빈도를 세어 표로 정리한다.

	you	say	goodbye	and	i	hello	.
you	0	1	0	0	0	0	0
say	1	0	1	0	1	1	0
goodbye	0	1	0	1	0	0	0
and	0	0	1	0	1	0	0
i	0	1	0	1	0	0	0
hello	0	1	0	0	0	0	1
.	0	0	0	0	0	1	0

각 행은 단어를 표현한 벡터가 됨.
 이 표는 행렬의 형태를 띠고 있기 때문에 동시발생 행렬이라고 한다

벡터 간 유사도

코사인 유사도(cosine similarity)

두 벡터 $x = (x_1, x_2, x_3, \dots, x_n)$ 과 $y = (y_1, y_2, y_3, \dots, y_n)$ 코사인 유사도

$$\text{similarity}(x, y) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|} = \frac{x_1 y_1 + \dots + x_n y_n}{\sqrt{x_1^2 + \dots + x_n^2} \sqrt{y_1^2 + \dots + y_n^2}}$$

[식 2.1]

Python Code

자연어처리 딥러닝 구현 코드 예제

LEARN MORE

구현 예시1. 말뭉치로부터 동시발생 행렬을 만들어주는 함수

```
def create_co_matrix(corpus, vocab_size, window_size=1):
    corpus_size = len(corpus)
    co_matrix = np.zeros((vocab_size, vocab_size), dtype=np.int32)

    for idx, word_id in enumerate(corpus):
        for i in range(1, window_size + 1):
            left_idx = idx - i
            right_idx = idx + i

            if left_idx >= 0:
                left_word_id = corpus[left_idx]
                co_matrix[word_id, left_word_id] += 1

            if right_idx < corpus_size:
                right_word_id = corpus[right_idx]
                co_matrix[word_id, right_word_id] += 1

    return co_matrix
```

동시 발생 행렬: 분포가설에 기초한 단어를 벡터로 나타내는 방법

구현 예시2. 유사 단어의 랭킹을 표시 해주는 함수

```
def most_similar(query, word_to_id, id_to_word, word_matrix, top=5):
    #(1) 검색어를 꺼낸다.
    if query not in word_to_id:
        print('%s(을 )를 찾을 수 없습니다.' % query)
        return
    print('\n[query] ' + query)
    query_id = word_to_id[query]
    query_vec = word_matrix[query_id]

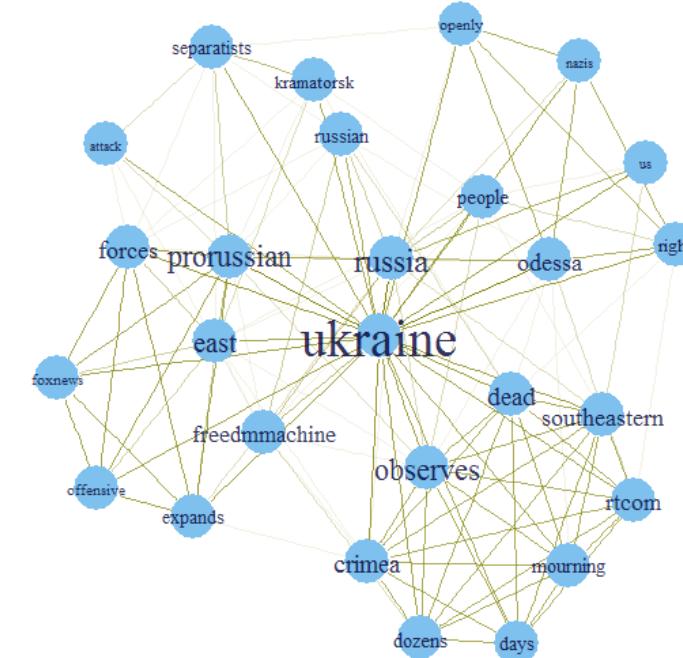
    #(2) 코사인 유사도 계산
    vocab_size = len(id_to_word)
    similarity = np.zeros(vocab_size)
    for i in range(vocab_size):
        similarity[i] = cos_similarity(word_matrix[i], query_vec)

    # (3) 코사인 유사도를 기준으로 내림차순으로 출력
    count = 0
    for i in (-1 * similarity).argsort():
        if id_to_word[i] == query:
            continue
        print(' %s: %s' % (id_to_word[i], similarity[i]))

        count += 1
        if count >= top:
            return
```

동시발생 행렬의 문제점

car? the? drive?



통계 기반 기법 개선

점별 상호정보량(Pointwise Mutual Information(PMI))

PMI는 확률 변수 x와 y에 대해 다음 식으로 정의됨

$$\text{PMI}(x,y) = \log_2 \frac{P(x,y)}{P(x)P(y)}$$

[식 2.2]

$P(x)$: x가 일어날 확률
 $P(y)$: y가 일어날 확률
 $P(x, y)$: x와 y가 동시에 일어날 확률

PMI 값이 높을수록 관련성이 높다

$$\text{PMI}(x,y) = \log_2 \frac{P(x,y)}{P(x)P(y)} = \log_2 \frac{\frac{C(x,y)}{N}}{\frac{C(x)}{N} \frac{C(y)}{N}} = \log_2 \frac{C(x,y) \cdot N}{C(x)C(y)}$$

[식 2.3]

-> 동시발생 행렬로부터 PMI 구하기

통계 기반 기법 개선

동시발생 횟수 관점일때 vs PMI 관점일때

$$\text{PMI}(\text{"the"}, \text{"car"}) = \log_2 \frac{10 \cdot 10000}{1000 \cdot 20} \approx 2.32 \quad [\text{식 2.4}]$$

$$\text{PMI}(\text{"car"}, \text{"drive"}) = \log_2 \frac{5 \cdot 10000}{20 \cdot 10} \approx 7.97 \quad [\text{식 2.5}]$$

PMI를 이용하면 car은 the보다 drive와 관련성이 더 강해졌다

why? 단어가 단독으로 출현하는 횟수가 고려되었기 때문

구현 예시3. 동시발생 행렬을 PPMI 행렬로 변환하는 함수

```
def ppmi(C, verbose=False, eps=1e-8):
    M = np.zeros_like(C, dtype=np.float32)
    N = np.sum(C)
    S = np.sum(C, axis=0)
    total = C.shape[0] * C.shape[1]
    cnt = 0

    for i in range(C.shape[0]):
        for j in range(C.shape[1]):
            pmi = np.log2(C[i, j] * N / (S[j]*S[i] + eps))
            M[i, j] = max(0, pmi)

            if verbose:
                cnt += 1
                if cnt % (total/100) == 0:
                    print('%.1f%% 완료' % (100*cnt/total))

    return M
```

동시발생 행렬						
[0 1 0 0 0 0]						
[1 0 1 0 1 1 0]						
[0 1 0 1 0 0 0]						
[0 0 1 0 1 0 0]						
[0 1 0 1 0 0 0]						
[0 1 0 0 0 0 1]						
[0 0 0 0 0 1 0]]						

PPMI						
[0. 1.807 0. 0. 0. 0. 0.]						
[1.807 0. 0.807 0. 0.807 0.807 0.]						
[0. 0.807 0. 1.807 0. 0. 0.]						
[0. 0. 1.807 0. 1.807 0. 0.]						
[0. 0.807 0. 1.807 0. 0. 0.]						
[0. 0.807 0. 0. 0. 0. 2.807]						
[0. 0. 0. 0. 2.807 0.]]						

3 자연어처리 최근 동향



1. Intent matching
사용자가 원하는 것 파악 (머신러닝 모델 이용)
2. Entities
built in system entity 이용
3. Dialog control
대화의 흐름을 구성

Scalar Lab