

↑↓
인공지능 Part 5



Contents



1. Text Classification	3
----------------------------------	---



Chapter

2

Text Classification

- Text Preprocessing
- Word Embedding
- Word2Vec

Text Preprocessing



Tokenization

- Word Tokenization

- Punctuation → . , ? ; !

- ex) In natural language processing, word embedding is a term used for the representation of words for text analysis,
→ "In", "natural", "language", "processing", "word", "embedding", "is", "a", "term", "used", "for", "the",
"representation", "of", "words", "for", "text", "analysis"

- NTK library → English corpus

```
from nltk.tokenize import word_tokenize
from nltk.tokenize import WordPunctTokenizer
from tensorflow.keras.preprocessing.text import text_to_word_sequence
```

Tokenization

- 구두점이나 특수 문자 포함
- 줄임말과 단어 내 띄어쓰기
- 문장 토큰화
- 한국어 토큰화
 - 교착어 (조사, 어미 사용)
 - 형태소(Morpheme)
 - 자립 형태소 → 체언(명사, 대명사, 수사), 수식언(관형사, 부사), 감탄사
 - 의존 형태소 → 접사, 어미, 조사, 어간
 - 띄어 쓰기
 - ex) 말뭉치또는코퍼스는자연언어연구를위해특정한목적을가지고언어의표본을추출한집합이다.
 - ex) Howmanystepsshouldyouwalkeachdaytostayhealthyandlivelonger?
- 품사 태깅 (Part-of-speech tagging)

Cleaning & Normalization

- **Cleaning**
- **Normalization**
 1. 규칙에 기반한 단어들의 통합
 2. 대, 소문자 통합
 3. 불필요한 단어 제거
 4. 정규 표현식 (Regular Expression)

Stemming

- 어간 추출 → 주어진 단어에서 핵심 의미를 가지고 있는 부분 찾기

- 형태소분석

- caring → car

ex) formalize → formal

allowance → allow

electrical → electric

- 한국어에서의 어간 추출

언	품사
체언	명사, 대명사, 수사
수식언	관형사, 부사
관계언	조사
독립언	감탄사
용언	동사, 형용사 → 어간(stem) + 어미(ending)

Lemmatization

- 표제어 → 주어진 단어의 사전적 어원을 찾는 과정, 기본 사전형 단어
 - caring → care
- 단어의 형태소 분리
 - 어간(stem), 접사(affix)

Stop Words

- 불용어
- 데이터에서 유의미한 단어 토큰을 위해 의미 없는 단어 토큰 제거
 - NLTK → 179
- 한국어 불용어 제거
 - 토큰화 후 조사, 접속사 제거

ex) Keras's text preprocessing

```
'''
Indian teenager Rameshbabu Praggnanandhaa beat five-time world champion and
world number one chess player Magnus Carlsen in 39 moves, after Carlsen made
mistakes early in their game on February 20.
The 31-year-old Norwegian Carlsen left the online game as soon as it was over.
Later he said that he was still feeling the effects of being ill with the coronavirus,
and that it had affected his game.
'''
```



```
[['indian', 'teenager', 'rameshbabu', 'praggnanandhaa', 'beat', 'five-time', 'world', 'champion', 'world', 'number',
'one', 'chess', 'player', 'magnus', 'carlsen', 'moves', 'carlsen', 'made', 'mistakes', 'early', 'game', 'february'],
['31-year-old', 'norwegian', 'carlsen', 'left', 'online', 'game', 'soon'], ['later', 'said', 'still', 'feeling', 'eff
ects', 'ill', 'coronavirus', 'affected', 'game']]
```



```
[
    [5, 6, 1, 1, 1, 1, 4, 1, 4, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 3, 1],
    [1, 1, 2, 1, 1, 3, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 3]
]
```

Word Embedding



Word Embedding

- **Sparse Representation (희소 표현)**

- 벡터, 행렬의 값이 대부분이 0으로 표현
 - ex) one-hot-vector
 - ex) 강아지 = [0 0 0 0 **1** 0 0 0 0 0 0 0 ... 중략 ... 0]

- **Dense Representation (밀집 벡터)**

- 사용자가 설정한 값으로 모든 단어의 벡터 차원을 맞춤
- 실수로 표현
 - ex) 강아지 = [0.2 1.8 1.1 -2.1 1.1 2.8 ...] # 이 벡터의 차원은 128

Word Embedding

- 희소 표현으로는 유사성 표현할 수 없음 → *다차원 공간에 벡터화*
- 단어 간 유사성을 벡터화. → *워드 임베딩*
- 단어를 벡터로 표현 → 밀집 표현
 - *Embedding Vector*
 - LSA, Word2Vec, FastText, Glove, Keras의 Embedding()

```
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Embedding, Dropout, Concatenate
embedding_layer = Embedding(len(word_index) + 1, EMBEDDING_DIM,
                             weights=[embedding_matrix],
                             input_length=MAX_SEQUENCE_LENGTH,
                             trainable=True)
```


Word2Vec



Practice) Text Classification for 20 Newsgroups with CNN

- *Scikit-Learn의 Twenty Newsgroups → 20 categories*
 - *18,846 News data = 11,314(Train) + 7,532(Test)*
- *Word Embedding*
 - *Word를 R차원의 Vector로 매핑*
 - ex) $W(\text{"cat"}) = (0.2, -0.4, 0.7, \dots)$*
 - ex) $W(\text{"mat"}) = (0.0, 0.6, -0.1, \dots)$*

Practice) Text Classification for 20 Newsgroups with CNN

- *Scikit-Learn* / *Twenty Newsgroups* → 20 categories
 - 18,846 News data = 11,314(Train) + 7,532(Test)

alt.atheism,
comp.graphics,
comp.os.ms-windows.misc,
comp.sys.ibm.pc.hardware,
comp.sys.mac.hardware,
comp.windows.x,
misc.forsale,
rec.autos,
rec.motorcycles,
rec.sport.baseball,

rec.sport.hockey,
sci.crypt,
sci.electronics,
sci.med,
sci.space,
soc.religion.christian,
talk.politics.guns,
talk.politics.mideast,
talk.politics.misc,
talk.religion.misc

Practice) Text Classification for 20 Newsgroups with CNN

```
categories = ['comp.sys.mac.hardware', 'rec.sport.baseball']

newsgroups_train = fetch_20newsgroups(subset='train', shuffle=True,
                                      categories=categories,)
```

```
texts = []

labels=newsgroups_train.target
texts = newsgroups_train.data

MAX_SEQUENCE_LENGTH = 1000
MAX_NB_WORDS = 20000

tokenizer = Tokenizer(num_words=MAX_NB_WORDS)
tokenizer.fit_on_texts(texts)

sequences = tokenizer.texts_to_sequences(texts)
```

```
[13, 11119, 1619, 367, 9, 7974, 7975, 23, 460, 220]
CPU times: user 404 ms, sys: 10.5 ms, total: 415 ms
Wall time: 437 ms
```

Practice) Text Classification for 20 Newsgroups with CNN

```
word_index = tokenizer.word_index

print('Found %s unique tokens.' % len(word_index))
```

Found 18002 unique tokens.

```
data = pad_sequences(sequences, maxlen=MAX_SEQUENCE_LENGTH)
```

```
print (data.shape)
print (data[0][200:250])
```

```
(1175, 1000)
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

```
labels = to_categorical(np.array(labels))
```

```
indices = np.arange(data.shape[0])
np.random.shuffle(indices)
data = data[indices]
labels = labels[indices]
nb_validation_samples = int(VALIDATION_SPLIT * data.shape[0])
```

```
x_train = data[:-nb_validation_samples]
y_train = labels[:-nb_validation_samples]
x_val = data[-nb_validation_samples:]
y_val = labels[-nb_validation_samples:]
```

Practice) Text Classification for 20 Newsgroups with CNN

```
%time
embeddings_index = {}

f = open('glove.6B.100d.txt')
for line in tqdm(f):
    values = line.split(' ')
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()

print ()
print ('Found %s word vectors.' % len(embeddings_index))
```

```
400000it [00:11, 34650.94it/s]
```

Found 400000 word vectors.

CPU times: user 11.1 s, sys: 545 ms, total: 11.7 s

Wall time: 11.5 s

Practice) Text Classification for 20 Newsgroups with CNN

• Model 1

```
sequence_input = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int32')
embedded_sequences = embedding_layer(sequence_input)
l_cov1= Conv1D(128, 5, activation='relu')(embedded_sequences)
l_pool1 = MaxPooling1D(5)(l_cov1)
l_cov2 = Conv1D(128, 5, activation='relu')(l_pool1)
l_pool2 = MaxPooling1D(5)(l_cov2)
l_cov3 = Conv1D(128, 5, activation='relu')(l_pool2)
l_pool3 = MaxPooling1D(35)(l_cov3) # global max pooling

l_flat = Flatten()(l_pool3)
l_dense = Dense(128, activation='relu')(l_flat)
preds = Dense(2, activation='softmax')(l_dense)

model = Model(sequence_input, preds)
```

Practice) Text Classification for 20 Newsgroups with CNN

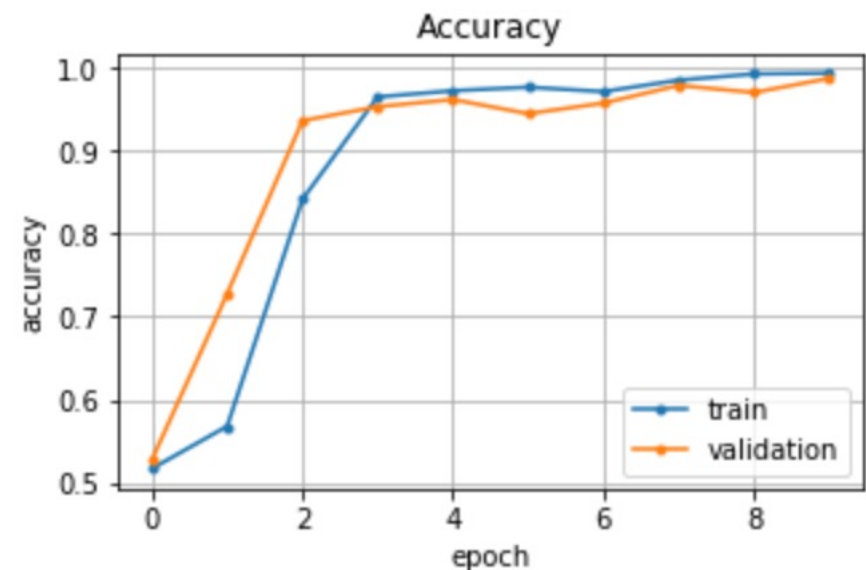
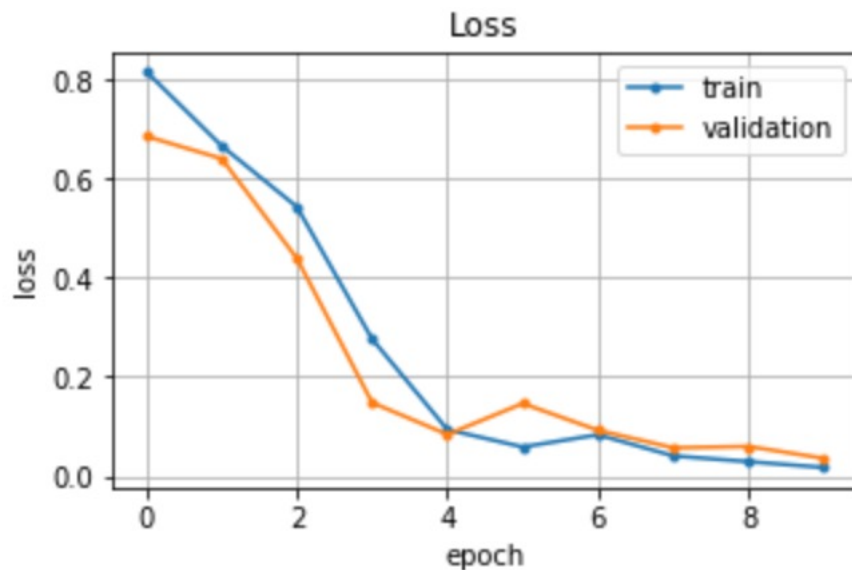
• Model 1

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['acc'])

history = model.fit(x_train, y_train, validation_data=(x_val, y_val),
                    epochs=10, batch_size=128)
```

Epoch 10/10

940/940 [=====] - 4s 4ms/step - loss: 0.0170 - acc: 0.9936 - val_loss: 0.0354 - val_acc: 0.9872



Practice) Text Classification for 20 Newsgroups with CNN

• Model 2

```

convs = []
filter_sizes = [3,4,5]

embedding_layer = Embedding(len(word_index) + 1, EMBEDDING_DIM,
                             weights=[embedding_matrix],
                             input_length=MAX_SEQUENCE_LENGTH,
                             trainable=True)

sequence_input = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int32')
embedded_sequences = embedding_layer(sequence_input)

for fsz in filter_sizes:
    l_conv = Conv1D(filters=128, kernel_size=fsz, activation='relu')(embedded_sequences)
    l_pool = MaxPooling1D(5)(l_conv)
    convs.append(l_pool)

l_merge = Concatenate(axis=1)(convs)
l_cov1 = Conv1D(128, 5, activation='relu')(l_merge)
l_pool1 = MaxPooling1D(5)(l_cov1)
l_cov2 = Conv1D(128, 5, activation='relu')(l_pool1)
l_pool2 = MaxPooling1D(30)(l_cov2)
l_flat = Flatten()(l_pool2)
l_dense = Dense(128, activation='relu')(l_flat)
preds = Dense(2, activation='softmax')(l_dense)

model = Model(sequence_input, preds)

model.summary()

```

Practice) Text Classification for 20 Newsgroups with CNN

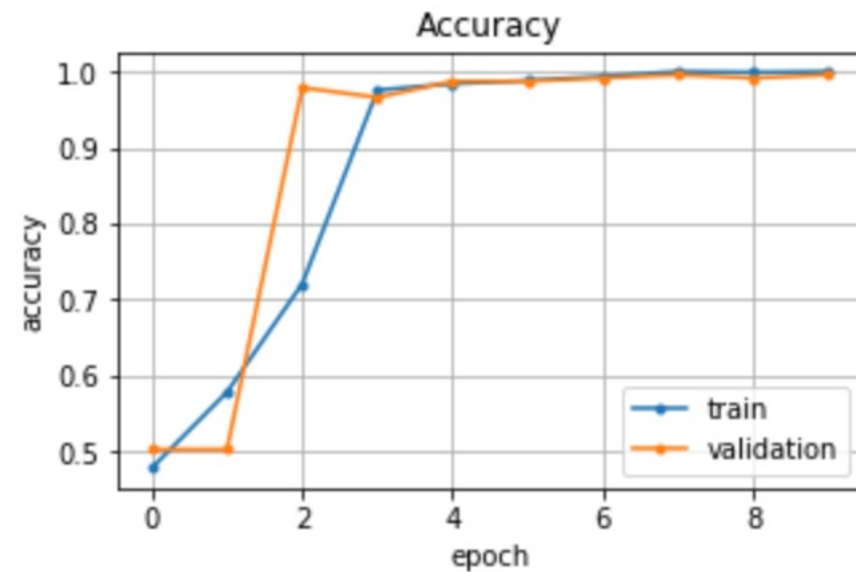
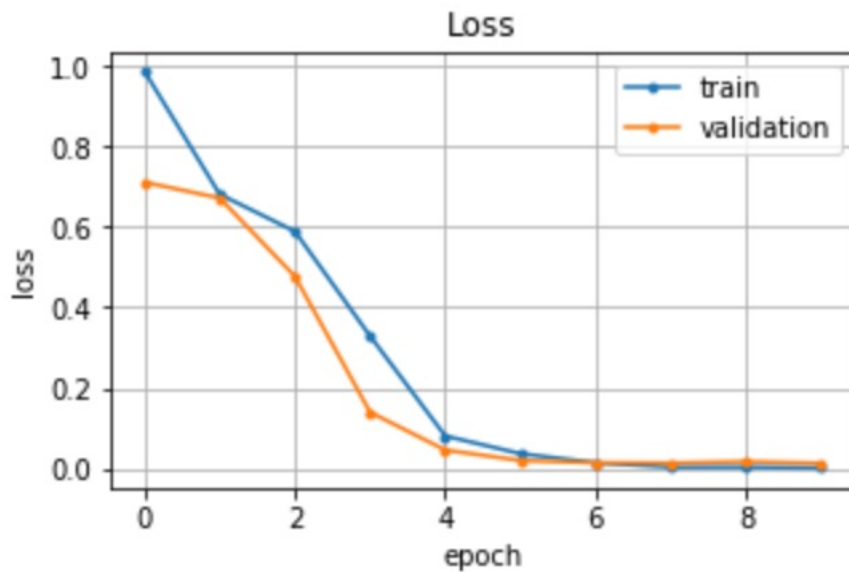
• Model 2

Epoch 9/10

940/940 [=====] - 11s 12ms/sample - loss: 0.0030 - acc: 0.9989 - val
_loss: 0.0165 - val_acc: 0.9915

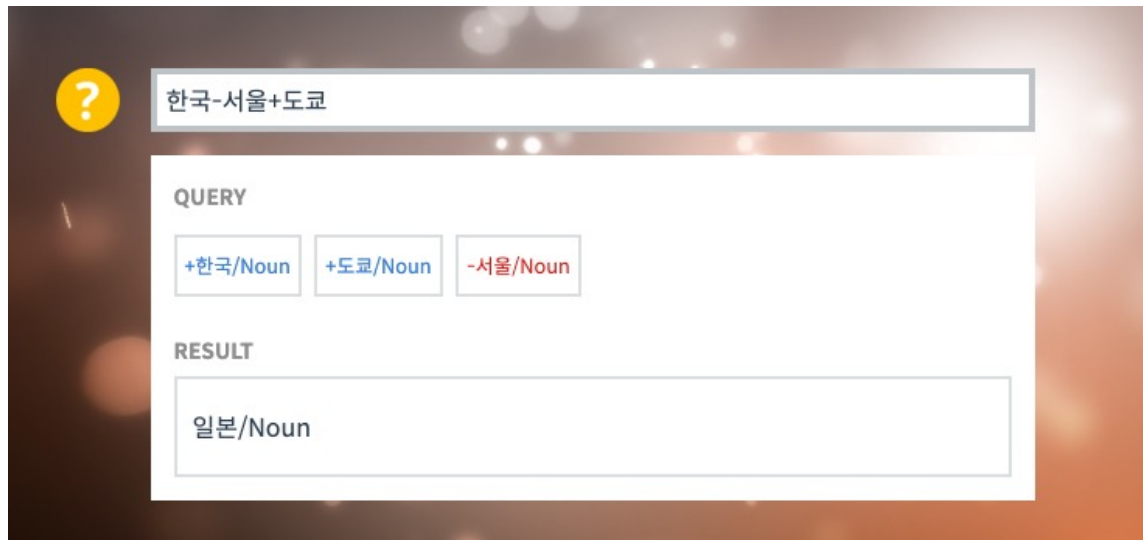
Epoch 10/10

940/940 [=====] - 11s 11ms/sample - loss: 0.0014 - acc: 1.0000 - val
_loss: 0.0127 - val_acc: 0.9957



Word2Vec

- 단어 벡터 간 유사도 반영 → 단어를 수치화
 - Word2Vec



The screenshot shows a web interface for Word2Vec. At the top left is a yellow circle with a question mark. Below it is a search bar containing the text "한국-서울+도쿄". Under the search bar, the word "QUERY" is displayed. Below "QUERY" are three buttons: "+한국/Noun" (blue text), "+도쿄/Noun" (blue text), and "-서울/Noun" (red text). Below these buttons, the word "RESULT" is displayed. Below "RESULT" is a box containing the text "일본/Noun".

<https://word2vec.kr/search/>

Word2Vec

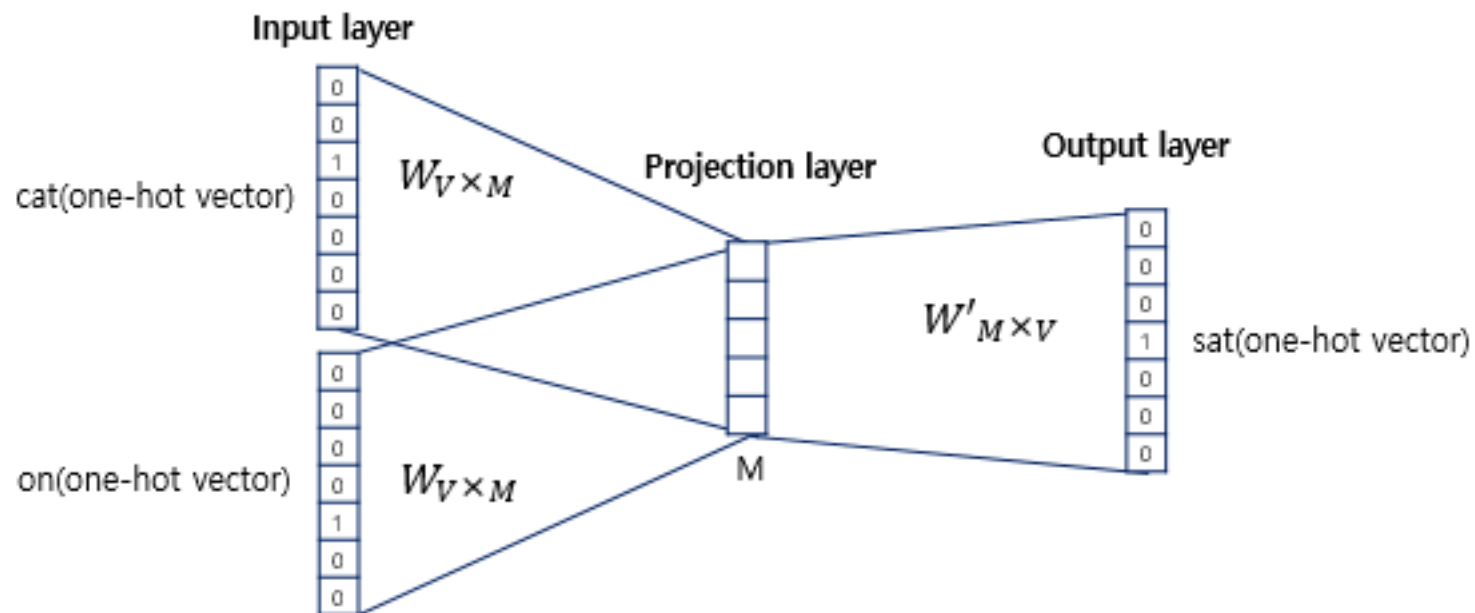
- CBOW(Continuous Bag of Words)

중심 단어 주변 단어
 ↓ ↓
 The fat cat sat on the mat
 The fat cat sat on the mat
 The fat cat sat on the mat
 The fat cat sat on the mat
 The fat cat sat on the mat
 The fat cat sat on the mat
 The fat cat sat on the mat

중심 단어	주변 단어
[1, 0, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 1, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 1, 0, 0]	[0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 0]	[0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 1]	[0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]

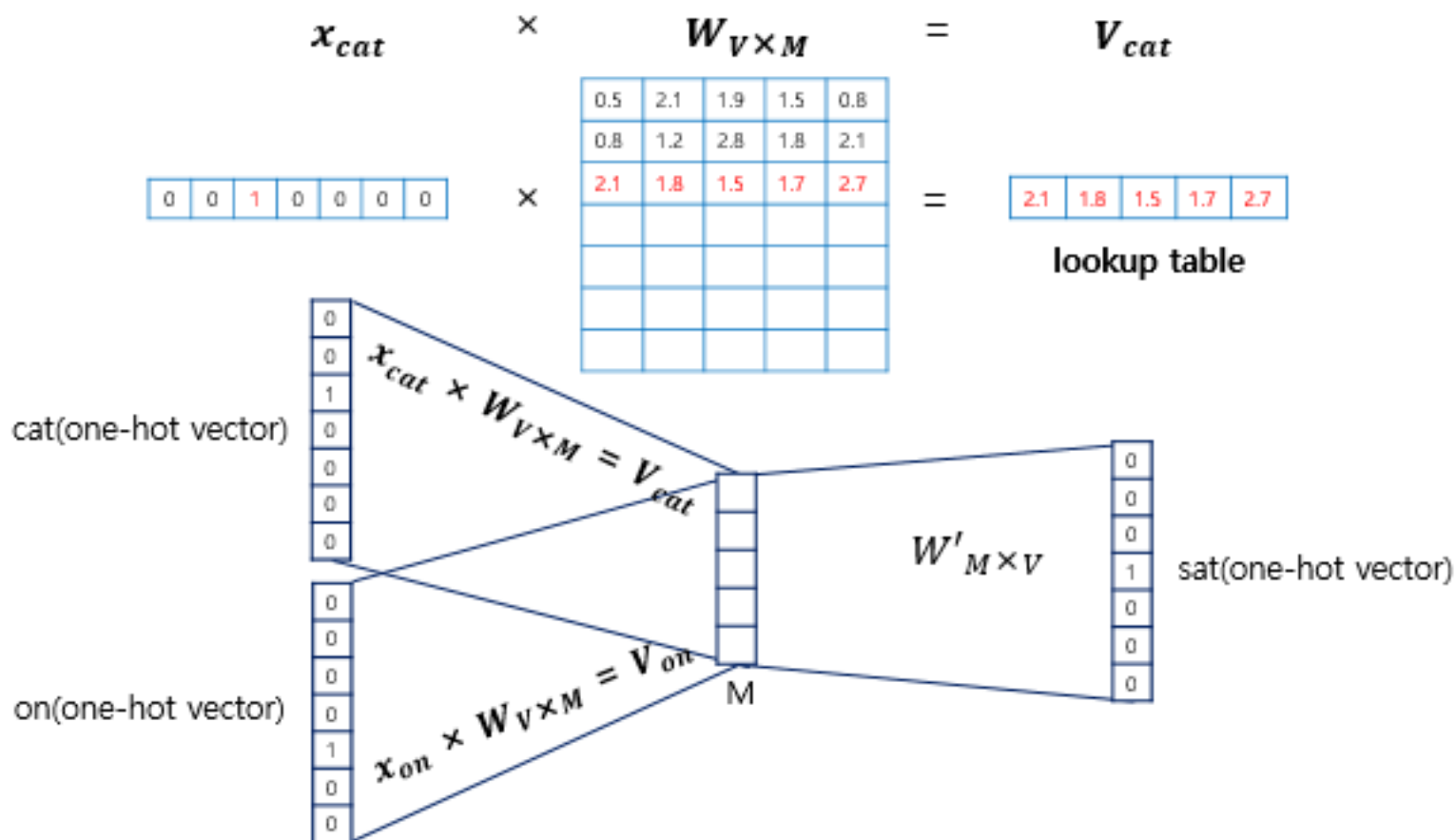
Word2Vec

- CBOW(Continuous Bag of Words)



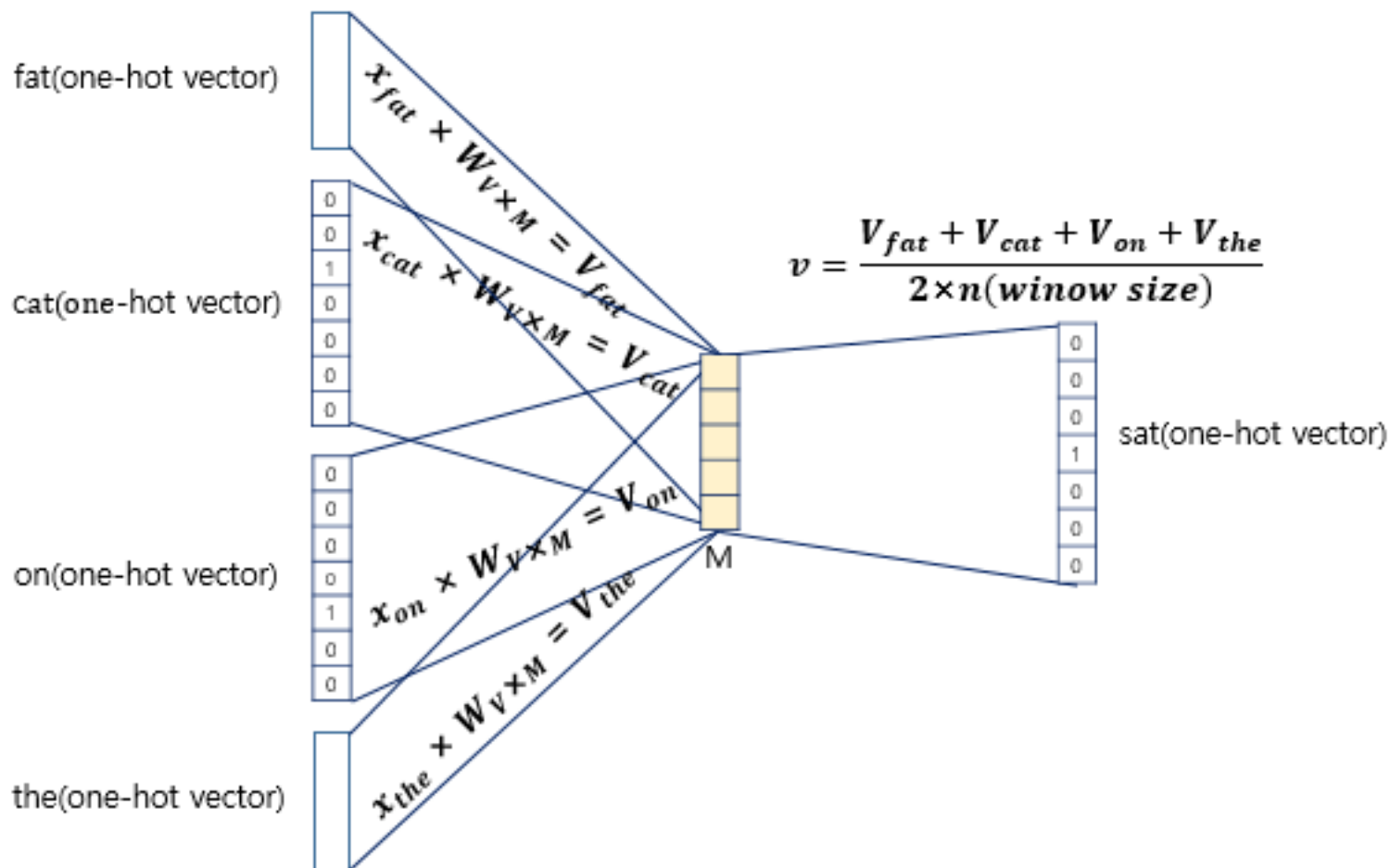
Word2Vec

- CROW(Continuous Bag of Words)



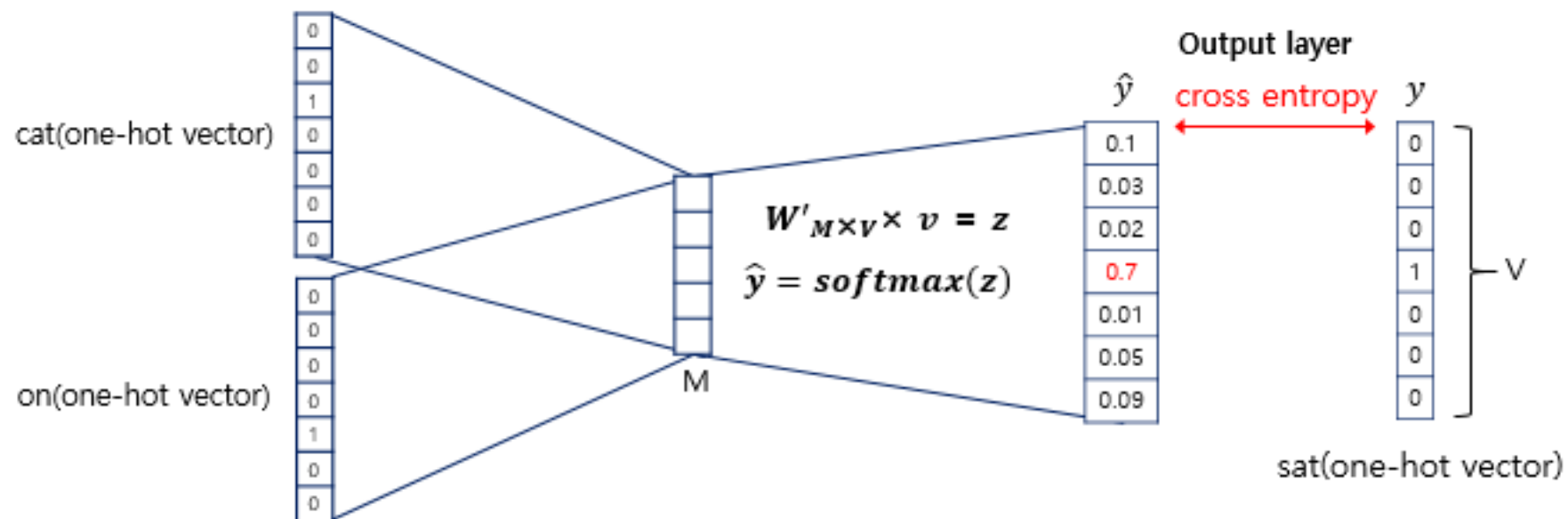
Word2Vec

- CROW(Continuous Bag of Words)



Word2Vec

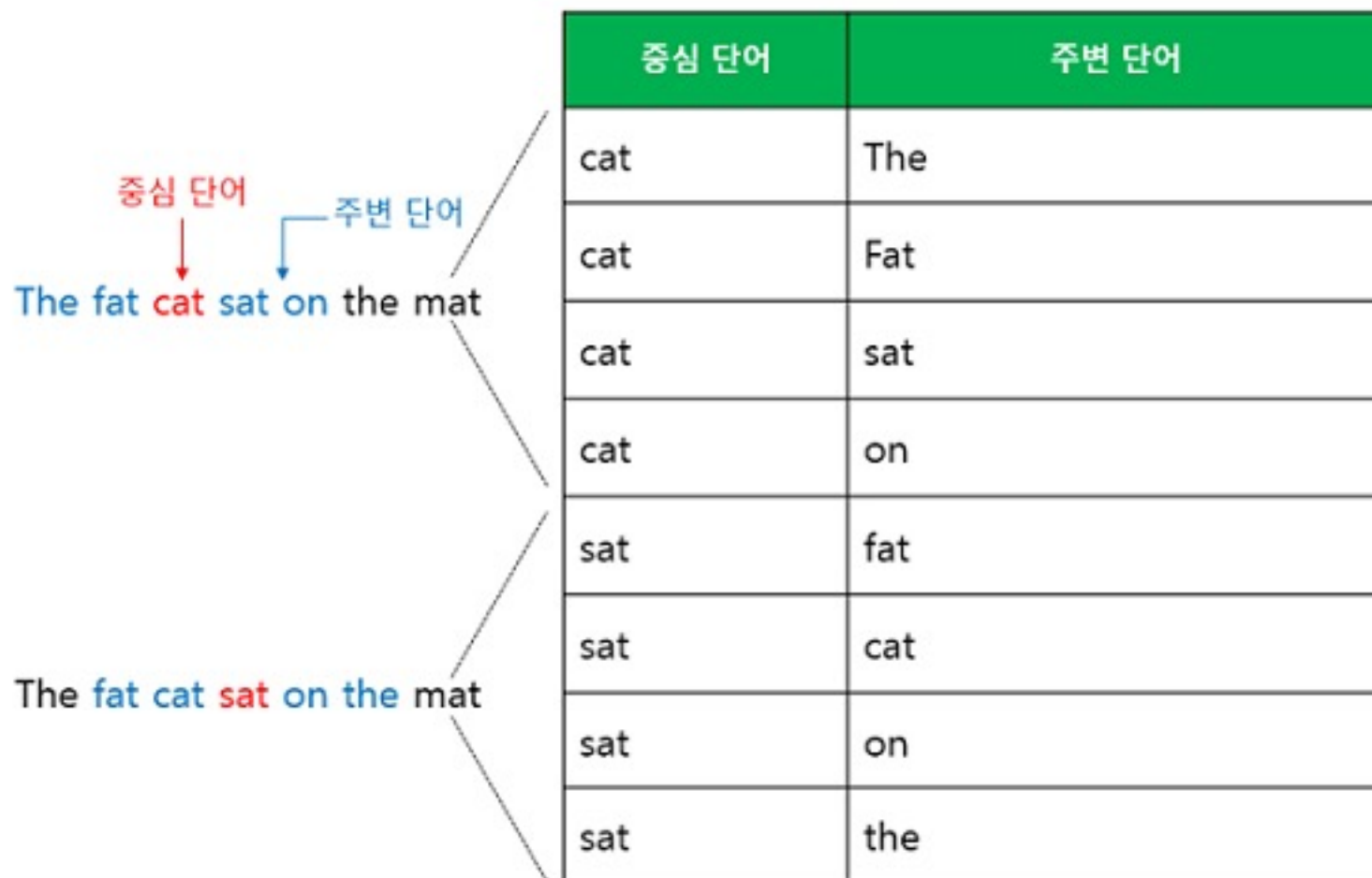
- CROW(Continuous Bag of Words)



Word2Vec

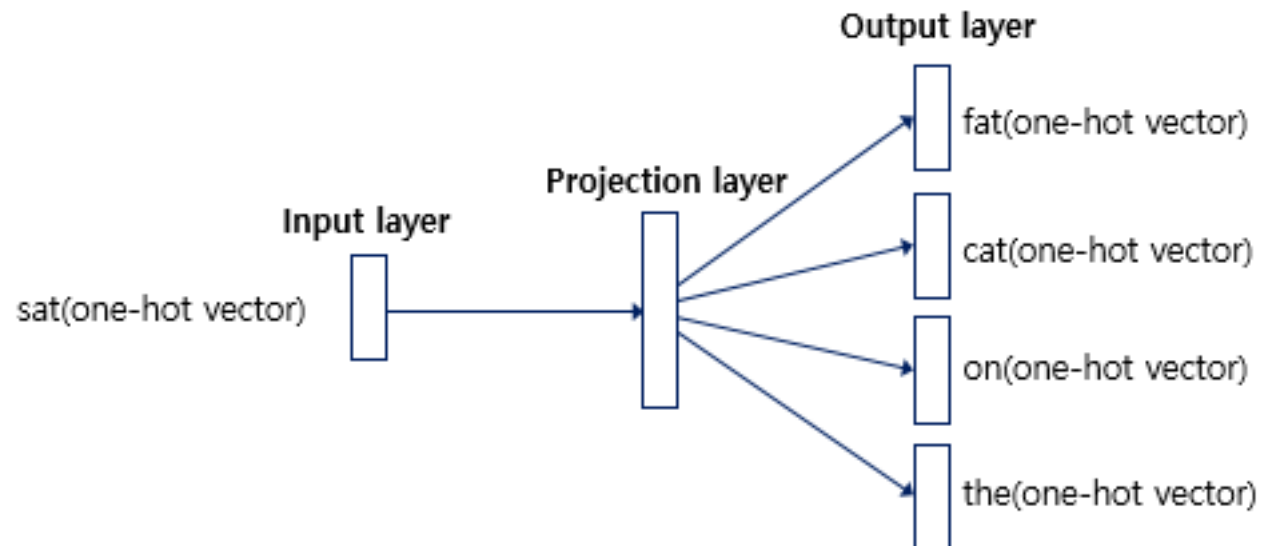
• Skip-Gram

- 중간 단어로 주변 단어를 예측



Word2Vec

- Skip-Gram



Word2Vec

- 단어 벡터 표현
 - Vector Representation of Words
 - Word Embedding
 - 단어 베이스 or 문자 베이스
- *Tensorflow is a library for deep learning.*

↓ ↓ ↓ ↓ ↓ ↓ ↓

121 99 88 66 5 14 21



{121, 99, 88, 66, 5, 14, 21}

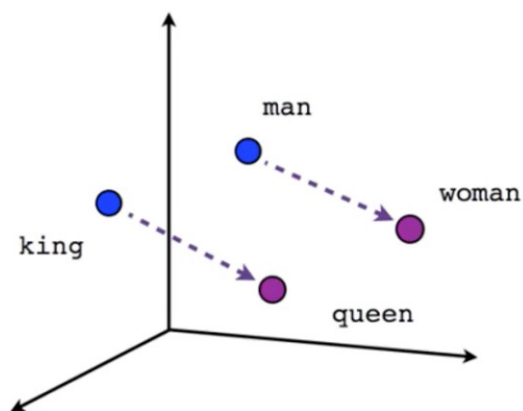
Vector란?

- 비슷한 특징 (Sementics)를 갖는 단어를 추측
- 다음에 올 단어를 추측
 - 번역
 - 자동 문장 생성, 자동 응답 → AI System

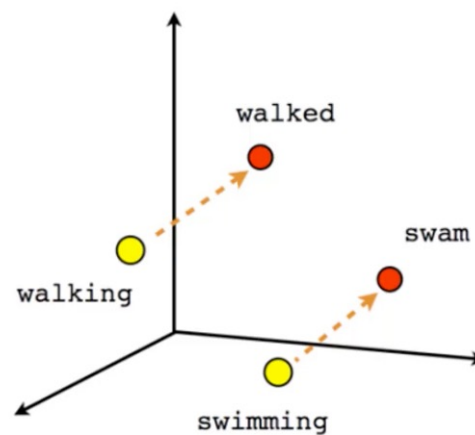
- ✓ 시멘틱스 (Sementics, 의미)
 - 프로그램이 무엇을 어떻게 수행할지 나타내 주며, 특정 기능의 의미가 다른 부분과의 상호 연관에 의해서만 정확히 설명

Vector란?

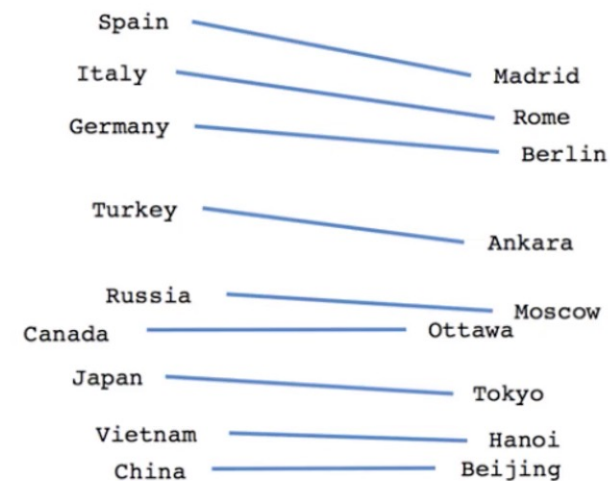
- <https://www.aclweb.org/anthology/N13-1090/>



Male-Female



Verb tense



Country-Capital

Tf-idf

- **TF-IDF(단어 빈도-역 문서 빈도, Term Frequency-Inverse Document Frequency)**
 - 문서의 유사도를 구하는 작업
 - 검색 시스템에서 검색 결과의 중요도를 정하는 작업
 - 문서 내에서 특정 단어의 중요도를 구하는 작업
- **tf(d,t) : 특정 문서 d에서의 특정 단어 t의 등장 횟수**
- **df(t) : 특정 단어 t가 등장한 문서의 수**
- **idf(d, t) : df(t)에 반비례하는 수**

$$idf(d, t) = \log\left(\frac{n}{1 + df(t)}\right)$$

Tf-idf

-	과일이	길고	노란	먹고	바나나	사과	싶은	저는	좋아요
문서1	0	0	0	1	0	1	1	0	0
문서2	0	0	0	1	1	0	1	0	0
문서3	0	1	1	0	2	0	0	0	0
문서4	1	0	0	0	0	0	0	1	1

단어	IDF(역 문서 빈도)
과일이	$\ln(4/(1+1)) = 0.693147$
길고	$\ln(4/(1+1)) = 0.693147$
노란	$\ln(4/(1+1)) = 0.693147$
먹고	$\ln(4/(2+1)) = 0.287682$

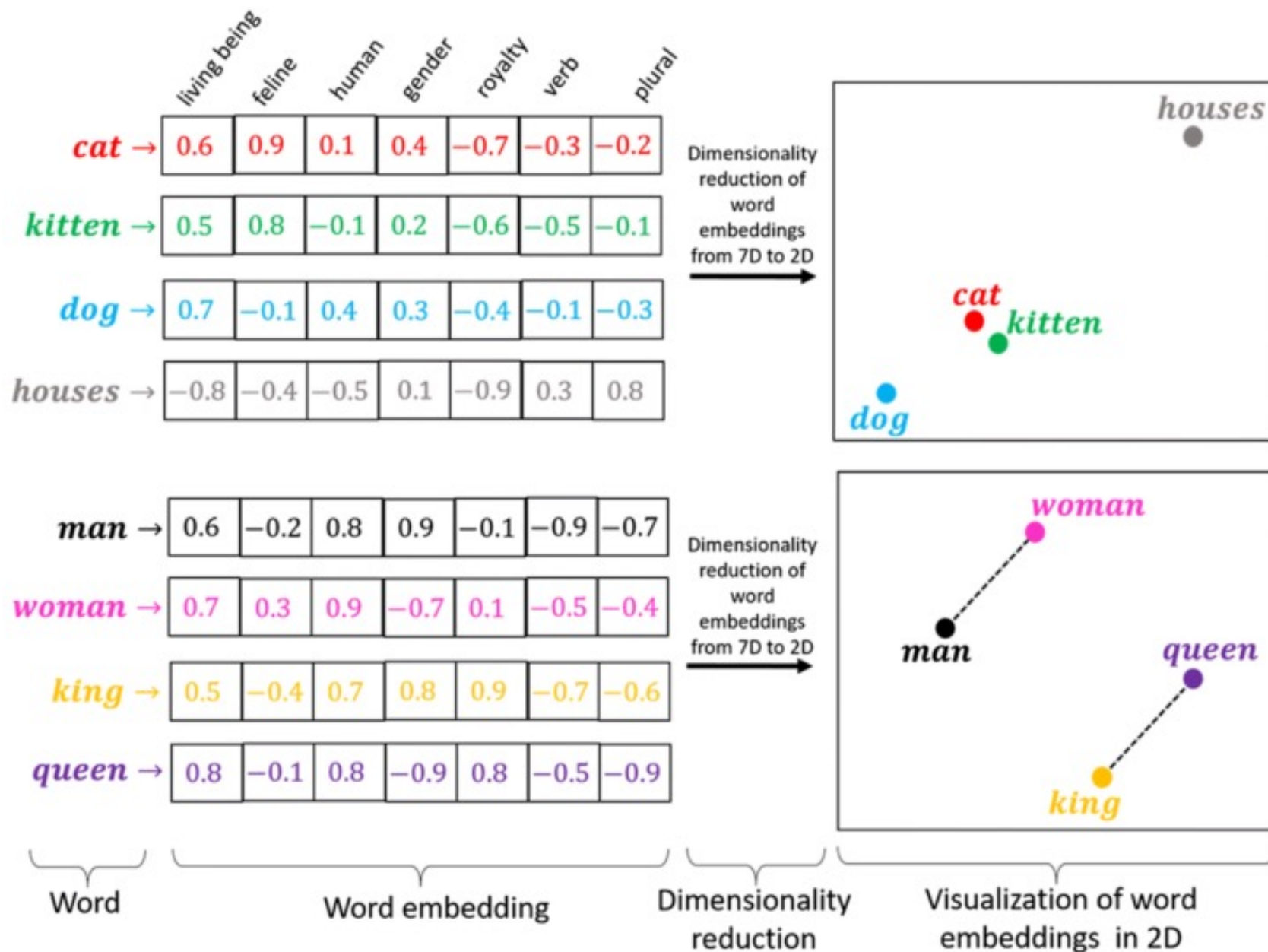
단어	IDF(역 문서 빈도)
바나나	$\ln(4/(2+1)) = 0.287682$
사과	$\ln(4/(1+1)) = 0.693147$
싶은	$\ln(4/(2+1)) = 0.287682$
저는	$\ln(4/(1+1)) = 0.693147$
좋아요	$\ln(4/(1+1)) = 0.693147$

Tf-idf

-	과일이	길고	노란	먹고	바나나	사과	싫은	저는	좋아요
문 서 1	0	0	0	0.287682	0	0.693147	0.287682	0	0
문 서 2	0	0	0	0.287682	0.287682	0	0.287682	0	0
문 서 3	0	0.693147	0.693147	0	0.575364	0	0	0	0
문 서 4	0.693147	0	0	0	0	0	0	0.693147	0.693147


- Word2Vec 라이브러리
- 텍스트 유성 모델링을 위한 패키지 (LDA)
 - LDA(Latent Dirichlet Allocation): 주어진 문서에 대해 각 문서에 어떤 주제들이 존재하는지 서술하는 확률적 토픽 모델
 - 토픽 모델: 문서 집합의 추상적인 주제를 발견하기 위한 통계적 모델 중 하나로, 텍스트 본문의 숨겨진 의미 구조를 발견하기 위해 사용되는 텍스트 마이닝 → **주제별 단어 표현을 그룹핑**

Gensim



한글 Wikipedia 다운로드

- <https://dumps.wikimedia.org/kowiki/latest/>
 - **kowiki-latest-pages-articles.xml.bz2**



The screenshot shows a web browser window with the address bar displaying 'dumps.wikimedia.org/kowiki/latest/'. Below the address bar, there are several tabs and a list of dump files. The list includes various XML and BZ2 files, each with a corresponding date and time. The file 'kowiki-latest-pages-articles.xml.bz2' is highlighted in yellow.

File Name	Date	Time
kowiki-latest-pages-articles-multistream-index5..>	22-Apr-2020	10:20
kowiki-latest-pages-articles-multistream-index5..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream-index6..>	03-Apr-2020	09:38
kowiki-latest-pages-articles-multistream-index6..>	13-Apr-2020	17:33
kowiki-latest-pages-articles-multistream-index6..>	22-Apr-2020	10:27
kowiki-latest-pages-articles-multistream-index6..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream.xml.bz2	22-Apr-2020	11:26
kowiki-latest-pages-articles-multistream.xml.bz..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream1.xml-p..>	22-Apr-2020	10:06
kowiki-latest-pages-articles-multistream1.xml-p..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream2.xml-p..>	22-Apr-2020	10:09
kowiki-latest-pages-articles-multistream2.xml-p..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream3.xml-p..>	22-Apr-2020	10:11
kowiki-latest-pages-articles-multistream3.xml-p..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream4.xml-p..>	22-Apr-2020	10:16
kowiki-latest-pages-articles-multistream4.xml-p..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream5.xml-p..>	22-Apr-2020	10:20
kowiki-latest-pages-articles-multistream5.xml-p..>	23-Apr-2020	21:25
kowiki-latest-pages-articles-multistream6.xml-p..>	03-Apr-2020	09:38
kowiki-latest-pages-articles-multistream6.xml-p..>	13-Apr-2020	17:33
kowiki-latest-pages-articles-multistream6.xml-p..>	22-Apr-2020	10:27
kowiki-latest-pages-articles-multistream6.xml-p..>	23-Apr-2020	21:25
kowiki-latest-pages-articles.xml.bz2	22-Apr-2020	00:47
kowiki-latest-pages-articles.xml.bz2-rss.xml	23-Apr-2020	21:24
kowiki-latest-pages-articles1.xml-plp76864.bz2	22-Apr-2020	00:15

Wikipedia 변환 툴 설치

- Ruby 설치
 - **Windows)** Ruby 설치 (<https://rubyinstaller.org>)
 - Ruby+Devkit 2.7.1-1 (x64)
 - MSYS2 base installation
 - **MaxOS)** *pre installed*

- Ruby 설치 확인

```
▶ ruby -v
ruby 2.6.3p62 (2019-04-16 revision 67580) [universal.x86_64-darwin19]
dowon@DOWON-MacBook ~
▶ █
```

Wikipedia 변환 툴 설치

- wp2txt 설치
 - ***gem install wp2txt***

```
▶ sudo gem install wp2txt
Password:
Fetching wp2txt-0.9.1.gem
Fetching htmlentities-4.3.4.gem
Fetching trollop-2.9.10.gem
Fetching parallel-1.19.1.gem
! The 'trollop' gem has been deprecated and has been replaced by 'optimist'.
! See: https://rubygems.org/gems/optimist
! And: https://github.com/ManageIQ/optimist
Successfully installed trollop-2.9.10
Successfully installed htmlentities-4.3.4
Successfully installed parallel-1.19.1
Successfully installed wp2txt-0.9.1
Parsing documentation for trollop-2.9.10
Installing ri documentation for trollop-2.9.10
Parsing documentation for htmlentities-4.3.4
Installing ri documentation for htmlentities-4.3.4
Parsing documentation for parallel-1.19.1
Installing ri documentation for parallel-1.19.1
Parsing documentation for wp2txt-0.9.1
Installing ri documentation for wp2txt-0.9.1
Done installing documentation for trollop, htmlentities, parallel, wp2txt after 0 seconds
4 gems installed
```

Wikipedia 변환 툴 설치

- bz2 파일 변환
 - **Windows)** 7zip (<https://www.7-zip.org/>)
- wp2txt
 - **Windows)** wp2txt --input-file ./kowiki-latest-pages-articles.xml
- 여러개의 파일을 하나로 결합
 - **Windows)** type kowiki-latest-pages-articles*.txt > kowiki.txt

Mecab

- MeCab 설치
 - <http://taku910.github.io/mecab/>

MeCab (和布蕪)とは

MeCabは 京都大学情報学研究科-日本電信電話株式会社コミュニケーション科学基礎研究所 共同研究ユニットプロジェクトを通じて開発されたオープンソース 形態素解析エンジンです。言語, 辞書, コーパスに依存しない汎用的な設計を 基本方針としています。パラメータの推定に Conditional Random Fields (CRF) を用いており, ChaSenが採用している 隠れマルコフモデルに比べ性能が向上しています。また、平均的に ChaSen, Juman, KAKASIより高速に動作します。ちなみに和布蕪(めかぶ)は, 作者の好物です。

ダウンロード

- MeCab はフリーソフトウェアです。GPL(the GNU General Public License), LGPL(Lesser GNU General Public License), または BSD ライセンスに従って本ソフトウェアを使用, 再配布することができます。詳細は COPYING, GPL, LGPL, BSD各ファイルを参照して下さい。
- MeCab 本体

Source

- mecab-0.996.tar.gz: [ダウンロード](#)
- 辞書は含まれていません。動作には別途辞書が必要です。

Binary package for MS-Windows

- mecab-0.996.exe: [ダウンロード](#)
- Windows 版には コンパイル済みの IPA 辞書が含まれています

Mecab

- MeCab (<http://eunjeon.blogspot.com/>)
- 설치 (MaxOS)
 - *(download) mecab-0.996-ko-0.9.2.tar.gz*
 - *tar xvfz mecab-0.996-ko-0.9.2.tar.gz*
 - *cd mecab-0.996-ko-0.9.2.tar.gz*
 - *./configure*
 - *make*
 - *make check*
 - *sudo make install*
 - *which mecab*

Mecab

- MeCab 사전 설치 (MaxOS)
- 설치 (MaxOS)
 - *(download) mecab-ko-dic-2.1.1-20180720.tar.gz*
 - *tar xvfz mecab-ko-dic-2.1.1-20180720.tar.gz*
 - *cd mecab-ko-dic-2.1.1-20180720.tar.gz*
 - *./configure*
 - *make*
 - *sudo make install*

Mecab

- MeCab (<http://eunjeon.blogspot.com/>)
- 설치 (Windows)
 - C:\mecab 폴더 생성
 - Mecab 실행 파일
 - <https://github.com/Pusnow/mecab-ko-msvc/releases/tag/release-0.9.2-msvc-3>
 - Mecab 사전 파일
 - <https://github.com/Pusnow/mecab-ko-dic-msvc/releases/tag/mecab-ko-dic-2.1.1-20180720-msvc>
 - 다운로드 받은 파일 압축 해제

Mecab

PC > 로컬 디스크 (C:) > mecab



mecab 검색

이름	수정한 날짜	유형	크기
mecab-ko-dic	2019-05-23 오전 10:32	파일 폴더	
tools	2019-05-23 오전 10:32	파일 폴더	
user-dic	2019-05-23 오전 10:32	파일 폴더	
libmecab.dll	2017-07-01 오전 1:49	응용 프로그램 확장	1,864KB
libmecab.lib	2017-07-01 오전 1:49	Object File Library	30KB
mecab.exe	2017-07-01 오전 1:49	응용 프로그램	106KB
mecab.h	2017-07-01 오전 1:48	C/C++ Header	42KB
mecab.lib	2017-07-01 오전 1:49	Object File Library	6KB
mecab-cost-train.exe	2017-07-01 오전 1:49	응용 프로그램	106KB
mecab-cost-train.lib	2017-07-01 오전 1:49	Object File Library	6KB
mecab-dict-gen.exe	2017-07-01 오전 1:49	응용 프로그램	106KB
mecab-dict-gen.lib	2017-07-01 오전 1:49	Object File Library	6KB
mecab-dict-index.exe	2017-07-01 오전 1:49	응용 프로그램	106KB
mecab-dict-index.lib	2017-07-01 오전 1:49	Object File Library	6KB
mecab-ko-dic.zip	2019-05-23 오전 11:18	ALZip ZIP File	82,165KB
mecabrc	2018-07-28 오후 8:12	파일	1KB
mecab-system-eval.exe	2017-07-01 오전 1:49	응용 프로그램	106KB
mecab-system-eval.lib	2017-07-01 오전 1:49	Object File Library	6KB
mecab-test-gen.exe	2017-07-01 오전 1:49	응용 프로그램	106KB
mecab-test-gen.lib	2017-07-01 오전 1:49	Object File Library	6KB

Mecab

- MeCab (<http://eunjeon.blogspot.com/>)
- 설치 (Windows)
 - conda 가상환경 생성
 - `conda create -n "가상환경이름" python=3.7`
 - python wheel 다운로드, 설치
 - https://github.com/Pusnow/mecab-python-msvc/releases/tag/mecab_python-0.996_ko_0.9.2_msvc-2
 - site-packages 폴더로 이동
 - `pip install mecab_python-0.996_ko_0.9.2_msvc-cp37-cp37m-win_amd64.whl`

Mecab

• MeCab 실행

```

▶ mecab
오늘은 날씨가 참 좋군요
오늘      NNG,* ,T,오늘 ,*,*,*,*
은        JX,* ,T,은 ,*,*,*,*
날씨      NNG,* ,F,날씨 ,*,*,*,*
가        JKS,* ,F,가 ,*,*,*,*
참        MAG,성분부사|정도부사,T,참 ,*,*,*,*
좋        VA,* ,T,좋 ,*,*,*,*
군요      EC,* ,F,군요 ,*,*,*,*
EOS

```

• MeCab-Python 설치

- *pip install mecab-python3*

- *python*

```
>>> import MeCab
```

```
>>> m = MeCab.Tagger()
```

```
>>> ret = m.parse("안녕하세요.")
```

```
>>> print(ret)
```

```

[?] python
Python 3.7.6 (default, Jan  8 2020, 13:42:34)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import MeCab
>>> m = MeCab.Tagger()
>>> ret = m.parse("오늘은 뭐 먹을 까?")
>>> print(ret)
오늘은 記号,一般,*,*,*,*,*
뭐      記号,一般,*,*,*,*,*
먹을까 記号,一般,*,*,*,*,*
?       名詞,サ変接続,*,*,*,*,*

```

Mecab

- 은전한닢 프로젝트 (<https://github.com/koshort/pyeunjeon>)

```

▶ mecab
오늘은 날씨가 참 좋군요
오늘      NNG,* ,T,오늘,* ,* ,* ,*
은        JX,* ,T,은,* ,* ,* ,*
날씨      NNG,* ,F,날씨,* ,* ,* ,*
가        JKS,* ,F,가,* ,* ,* ,*
참        MAG,성분부사|정도부사,T,참,* ,* ,* ,*
좋        VA,* ,T,좋,* ,* ,* ,*
군요      EC,* ,F,군요,* ,* ,* ,*
EOS

```

- MeCab-Python 설치

- `pip install mecab-python3`

- `python`

```
>>> import MeCab
```

```
>>> m = MeCab.Tagger()
```

```
>>> ret = m.parse("안녕하세요.")
```

```
>>> print(ret)
```

```

[?] python
Python 3.7.6 (default, Jan  8 2020, 13:42:34)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import MeCab
>>> m = MeCab.Tagger()
>>> ret = m.parse("오늘은 뭐 먹을 까?")
>>> print(ret)
오늘은 記号,一般,* ,* ,* ,* ,*
뭐      記号,一般,* ,* ,* ,* ,*
먹을까 記号,一般,* ,* ,* ,* ,*
?      名詞,サ変接続,* ,* ,* ,* ,*

```

Mecab

- *kowiki-latest-pages-articles.xml*
- *wp2txt --input-file kowiki-latest-pages-articles.xml*
- *type kowiki-latest-pages-articles* > kowiki.txt*

Mecab

- (Windows) mecab 실행 파일을 path에 추가
- ***mecab -b 100000 -Owakati kowiki.txt -o kowiki_wakachi.txt***
 - wakati: 단순 띄어쓰기

Mecab

- *activate root*
- *jupyter notebook*