자연어처리 Embedding

리미애오

목차

Introduction to Word Embedding

백혜림 hreeee@yonsei.ac.kr

Word : Discrete, not Continuous

- 단어는 discrete symbol & categorical value 형태이지만,
- 우리의 머릿속에서는 다르게 동작
  - 어휘는 계층적 의미 구조를 지니고 있으며,
  - 이에 따라 단어 사이의 유사성을 지님
  - Ex) <파랑>과 <핑크> 중에서 <빨강>에 가까운 단어는 무엇인가?

- One-hot 인코딩으로 표현된 값은 유사도나 모호성을 표현할 수 없다.
  - Dense vector로 표현하는 것이 유리

Feature Vectors

- Feature (특징)
  - 샘플을 잘 설명하는 특징
  - 특징을 통해 우리는 특정 샘플을 수치화할 수 있다.

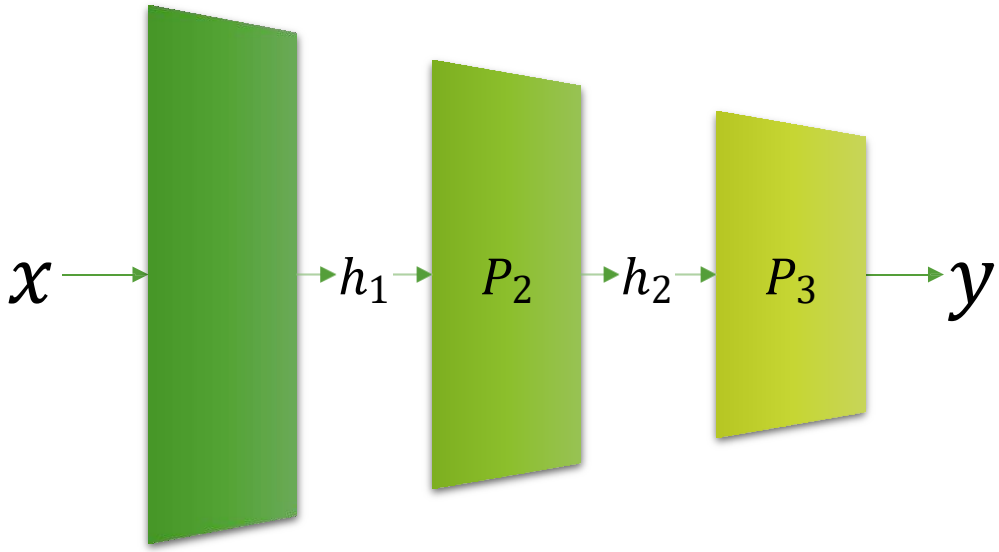
- Feature Vector
  - 각 특징들을 모아서 하나의 vector로 만드는 것

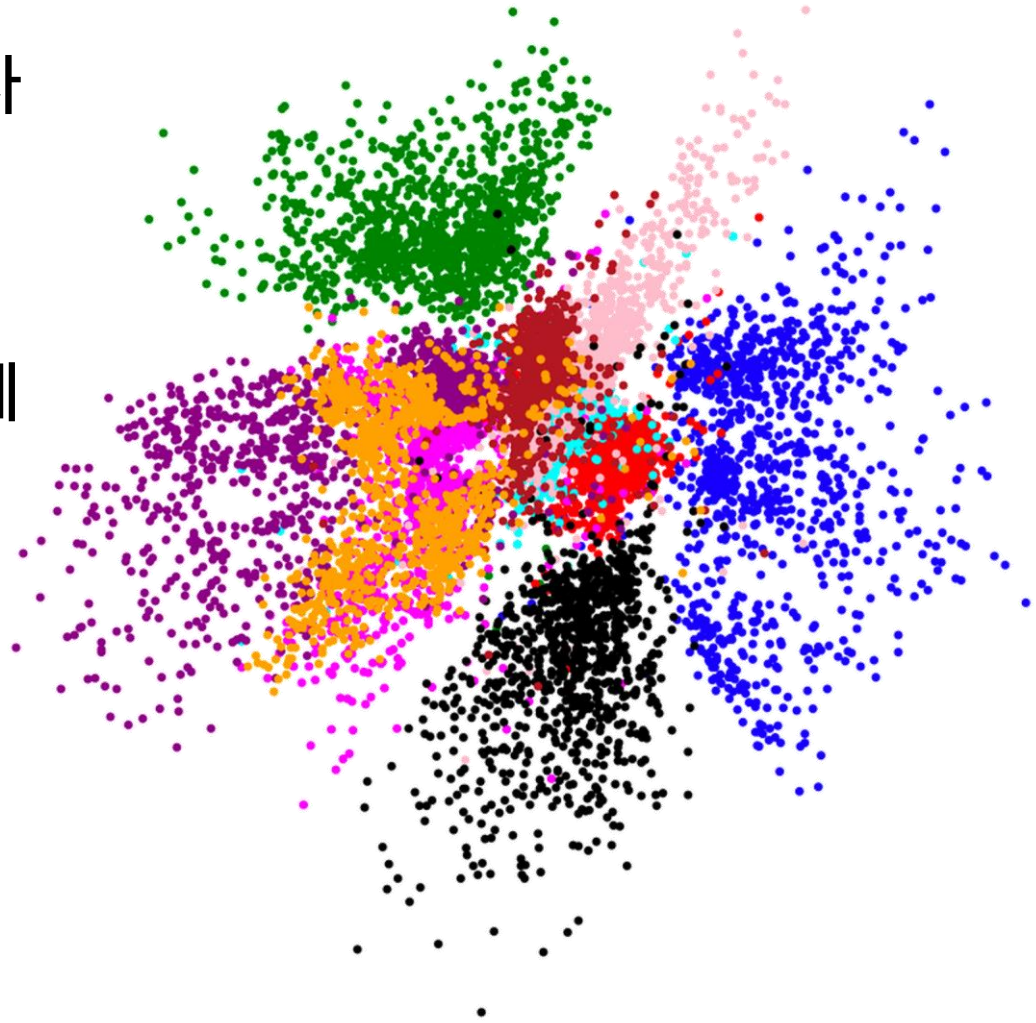
- 단어의 feature vector는 무엇이 될까?

Representation Learning via Dimension Reduction

- 신경망은 x와 y사이의 관계를 학습하는 과정에서 자연스럽게 x의 feature를 추출하는 방법을 학습함

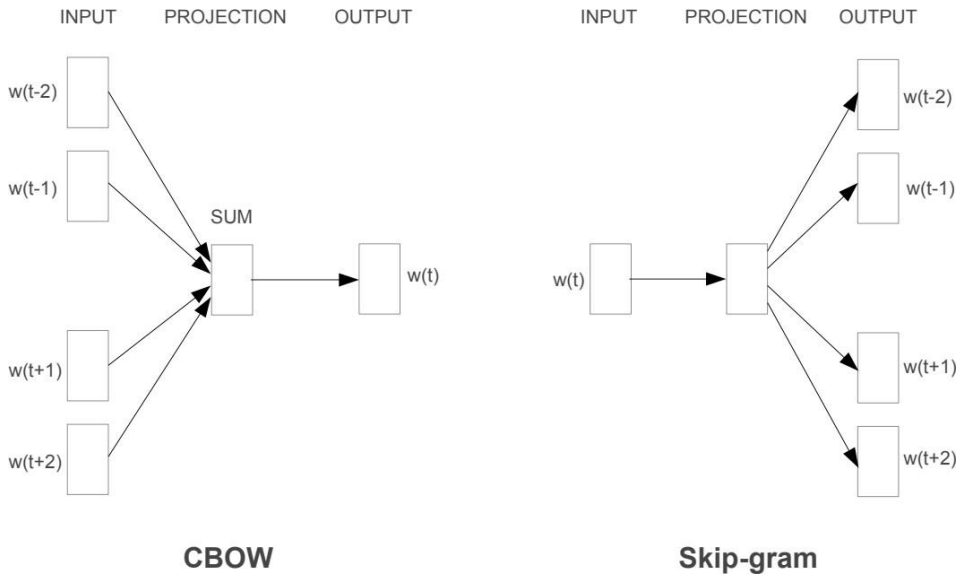
- 레이어 중간의 hidden representation은 y의 값을 구하기 위해 x에서 필요한 정보를 더 작은 차원에 압축 표현 한 것이라 할 수 있음





Word Embedding

- 딥러닝의 시대에 들어와 신경망의 이러한 특성을 활용하여 단어를 연속적인 값으로 표현하고자 하는 시도가 이어짐

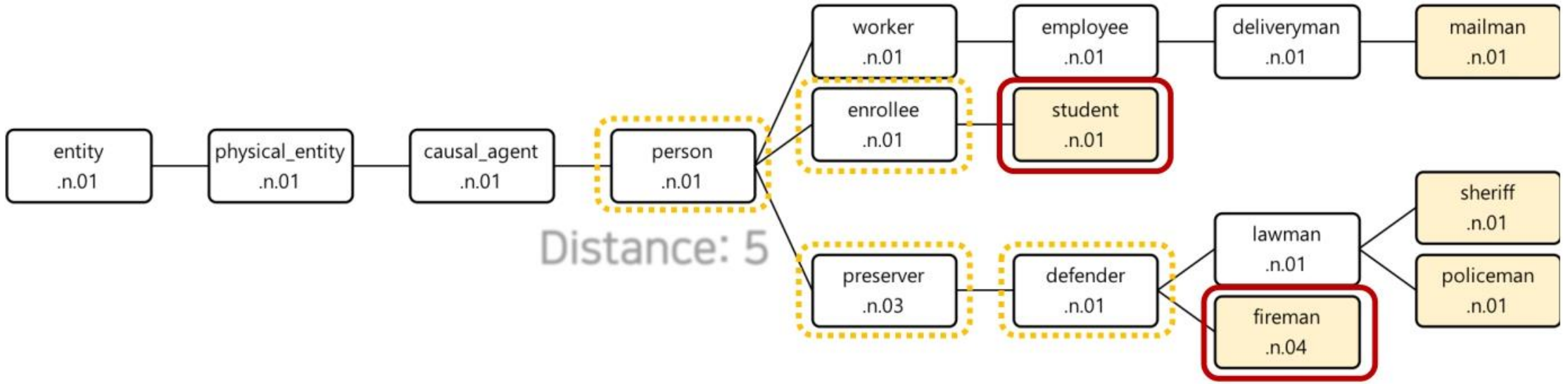


- 이전에 비해 훌륭한 dense vector를 얻을 수 있게 되어, 단어의 필요한 특징을 잘 표현할 수 있게 되었음
  - 유사도 등의 연산에 유리함

WordNet

백혜림 hreeee@yonsei.ac.kr

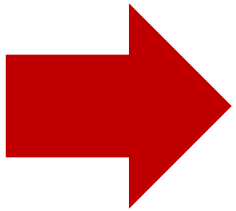
Distance between words



$$\text{similarity}(w, w') = -\log \text{distance}(w, w')$$

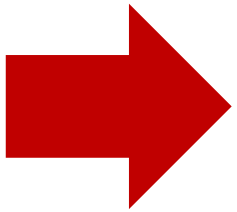
Summary

- WordNet
  - 단어의 계층적 구조를 파악할 수 있음
  - 동의어 집합(synset)을 구할 수 있음
  - 단어 사이의 유사도를 계산할 수 있음



코퍼스 없이도 가능!

- But
  - 특정 도메인(또는 수집 데이터)에 특화된 수치를 계산하고 싶을 때
  - 신조어나 사전에 등록되지 않은 단어

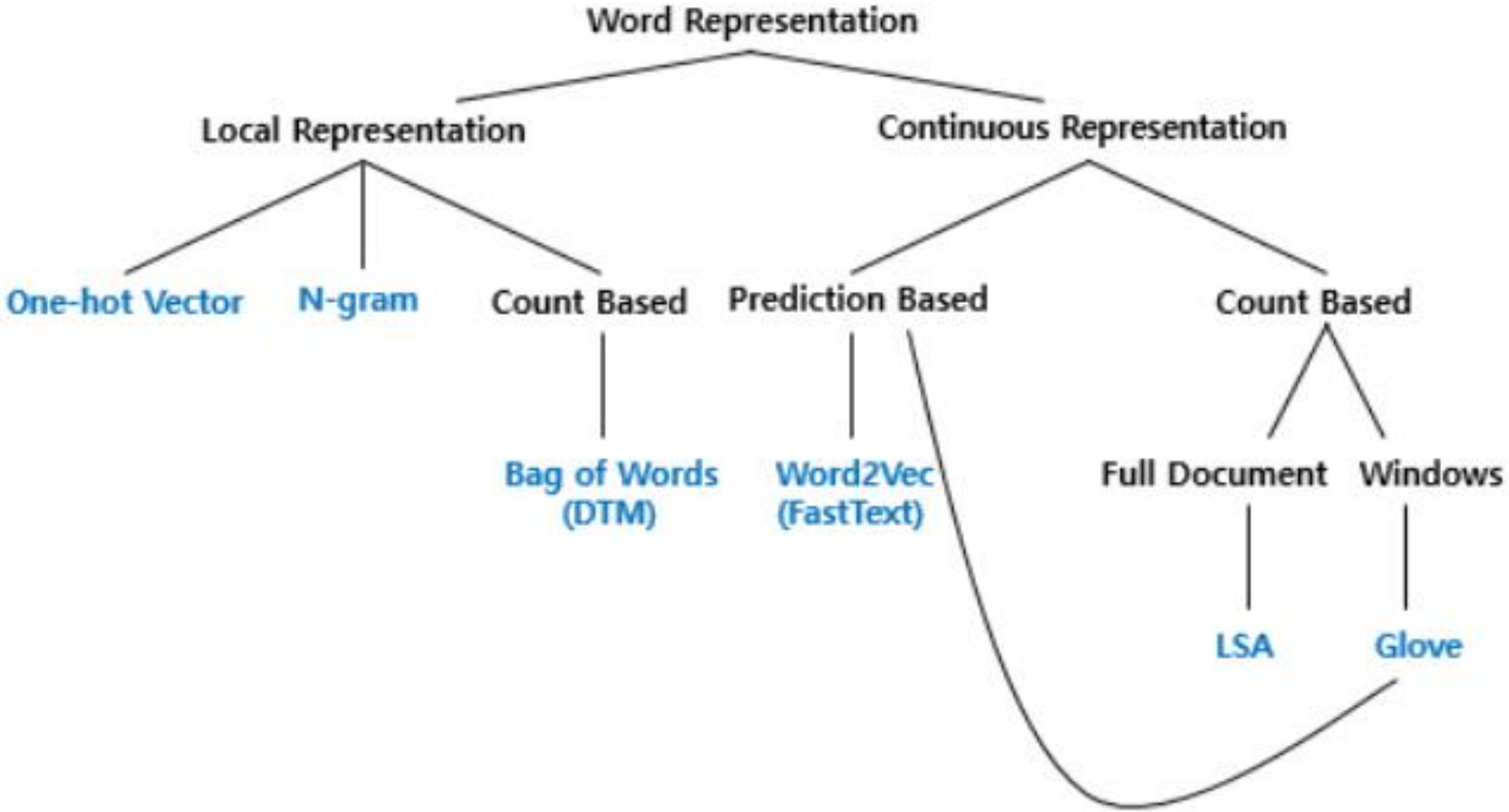


Data-driven방식 필요성!

Data-driven Methods

- Thesaurus기반 방식은 사전에 대해 의존도가 높으므로, 활용도가 떨어질 수 있다!
- 데이터에 기반한 방식은 (데이터가 충분하다면) task에 특화된 활용이 가능!

단어의 표현의 카테고리화



Data-driven Methods

- Local Representation
  - 해당 단어 그 자체만 보고, 특정값을 맵핑하여 단어를 표현하는 방법
  - = discrete representation
  - ex) puppy(강아지) : 1, cut(귀여운) : 2, lovely(사랑스러운) : 3 --> 숫자 맵핑

- Distributed Representation
  - 그 단어를 표현하고자 주변을 참고하여 단어를 표현하는 방법
  - = continuous representation
  - ex) puppy(강아지)라는 단어 근처에 주로 cute(귀여운), lovely(사랑스러운)이라는 단어가 자주 등장하므로, puppy는 cute, lovely한 느낌이다로 단어 정의하는 것!

WordFeature Vectors: Traditional Methods

백혜림 hreeee@yonsei.ac.kr

Bag of words

- 단어들의 순서는 전혀 고려하지 않고, 단어들의 출현 빈도(frequency)에만 집중하는 텍스트 데이터의 수치화 표현 방법
- 단어들의 가방

별 하나 에 추억 과
별 하나 에 사랑 과
별 하나 에 쓸쓸함 과
별 하나 에 동경 과
별 하나 에 시 와
별 하나 에 어머니 , 어머니

별	하나	에	추억	과	사랑	쓸쓸	함	동경	시	와	어머니	,
6	6	6	1	4	1	1	1	1	1	1	2	1

Bag of words

만드는 과정

1. 각 단어에 고유한 정수 인덱스를 부여한다.
2. 각 인덱스의 위치에 단어 토큰의 등장 횟수를 기록한 벡터를 만든다.

문서 단어 행렬(Document-Term Matrix, DTM)

- 다수의 문서에서 등장하는 각 단어들의 빈도를 행렬로 표현한 것
- 각 문서에 대한 BoW를 하나의 행렬로 만든 것
- Bow표현을 다수의 문서에 대해서 행렬로 표현하고 부르는 용어

문서 단어 행렬(Document-Term Matrix, DTM)

문서1 : 먹고 싶은 사과
문서2 : 먹고 싶은 바나나
문서3 : 길고 노란 바나나 바나나
문서4 : 저는 과일이 좋아요

-	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

- 각 문서에서 등장한 단어의 빈도를 행렬의 값으로 표기
- 문서 단어 행렬은 문서들을 서로 비교할 수 있도록 수치화할 수 있다는 점에 의의!

DTM의 한계

1) 희소 표현(Sparse representation)

- 각 문서 벡터의 차원은 원-핫 벡터와 마찬가지로 전체 단어 집합의 크기를 가짐
- 많은 문서벡터가 대부분의 값이 0을 가질 수도 있음 → 희소벡터, 희소행렬
- 희소벡터는 많은 양의 저장공간과 계산을 위한 리소스가 필요

2) 단순 빈도 수 기반 접근

- 예를들어 영어에 대한 DTM을 만들었을 때, 불용어인 The는 어떤 문서든 자주 등장한다.
- 그런데, 유사한 문서인지 비교하고싶은 문서1, 문서2, 문서3에서 동일하게 the가 빈도수가 높다고 해서 이 문서들이 유사한 문서라고 판단 X

TF-IDF

- 텍스트 마이닝(Text Mining)에서 중요하게 사용

- 어떤 단어 w가 문서 d 내에서 얼마나 중요한지 나타내는 수치

- TF(Term Frequency)
  - 단어의 문서 내에 출현한 횟수
  - 숫자가 클수록 문서 내에서 중요한 단어
  - 하지만, **'the'와 같은 단어도 TF값이 매우 클 것**

$$\text{TF-IDF}(w, d) = \frac{\text{TF}(w, d)}{\text{DF}(w)}$$

- IDF(Inverse Document Frequency)
  - 그 단어가 출현한 문서의 숫자의 역수(inverse)
  - 값이 클수록 'the'와 같이 일반적으로 많이 쓰이는 단어

TF-IDF를 feature로 사용할 수 없을까?

- TF-IDF는 문서에서 해당 단어가 얼마나 중요한지 수치화
- 중요한 문서가 비슷한 단어들은 비슷한 의미를 지닐까?
- 각 문서에서의 중요도를 feature로 삼아서 vector를 만든다면?

TF-IDF Matrix

- 단어의 각 문서(문장, 주제)별 TF-IDF 수치를 vector화
  - Row: 단어
  - Column: 문서

예제: 각 단어별 주제에 대한 TF-IDF 수치

단어	정치	경제	사회	생활	세계	연예	스포츠
문재인	높음	높음	높음	낮음	중간	낮음	낮음
BTS	낮음	낮음	낮음	낮음	높음	높음	낮음
류현진	낮음	낮음	낮음	낮음	높음	중간	높음
날씨	낮음	높음	중간	높음	낮음	낮음	높음
주식	높음	높음	중간	낮음	높음	낮음	낮음
버핏	낮음	높음	낮음	낮음	높음	낮음	낮음

Based on Context Window (Co-occurrence)

- 함께 나타나는 단어들을 활용

- 가정:
  - 의미가 비슷한 단어라면 **쓰임새가 비슷**할 것
  - 쓰임새가 비슷하기 때문에, 비슷한 문장 안에서 **비슷한 역할**로 사용될 것
  - **따라서 함께 나타나는 단어들이 유사**할 것

- Context Window를 사용하여 windowing을 실행
  - window의 크기라는 hyper-parameter 추가
  - **적절한 window 크기**를 정하는 것이 중요

Example

각 단어별 context window 내에 함께 나타난 빈도

	문재인	박근혜	이명박	BTS	싸이	방탄	주식	KOSPI	양적완화
문재인		높음	높음	낮음	낮음	낮음	높음	높음	낮음
박근혜	높음		높음	낮음	중간	낮음	높음	높음	낮음
이명박	높음	높음		낮음	낮음	낮음	높음	높음	중간
BTS	낮음	낮음	낮음		높음	높음	중간	낮음	낮음
싸이	낮음	낮음	낮음	높음		높음	중간	낮음	낮음
방탄	낮음	낮음	낮음	높음	높음		낮음	낮음	낮음
주식	높음	높음	높음	중간	중간	낮음		높음	높음
KOSPI	높음	높음	높음	낮음	낮음	낮음	높음		높음
양적완화	낮음	낮음	중간	낮음	낮음	낮음	높음	높음	

주요 단어만 feature로 활용하는 것도 한 방법

Summary

- Thesaurus 기반 방식에 비해 코퍼스(or 도메인) 특화된 표현 가능

- 여전히 sparse한 vector로 표현됨
  - PCA를 통해 차원 축소를 하는 것도 한 방법

Similarity Metrics

Manhattan Distance (L1 distance)

$$\mathrm{d_{L1}}(w, v) = \sum_{i=1}^{d} |w_i - v_i|, \text{ where } w, v \in \mathbb{R}^d.$$

Euclidean Distance (L2 distance)

$$\mathrm{d_{L2}}(w, v) = \sqrt{\sum_{i=1}^{d} (w_i - v_i)^2}, \text{ where } w, v \in \mathbb{R}^d.$$

Infinity Norm

$$d_\infty(w, v) = \max(|w_1 - v_1|, |w_2 - v_2|, \cdots, |w_d - v_d|), \text{ where } w, v \in \mathbb{R}^d$$

Cosine Similarity

$$\text{sim}_{\cos}(w, v) = \overbrace{\frac{w \cdot v}{|w||v|}}^{\text{dot product}} = \overbrace{\frac{w}{|w|}}^{\text{unit vector}} \cdot \frac{v}{|v|}$$

$$= \frac{\sum_{i=1}^{d} w_i v_i}{\sqrt{\sum_{i=1}^{d} w_i^2} \sqrt{\sum_{i=1}^{d} v_i^2}}$$

$$\text{where } w, v \in \mathbb{R}^d$$

Summary

- L1, L2 Norm과 Infinity Norm은 강조하고자 하는 것에 따라 사용

- Cosine Similarity는 벡터의 방향을 중요시 함
  - Feature vector의 각 차원의 상대적인 크기가 중요할 때 사용

Word Embedding

Word Embedding

• 자연어를 컴퓨터가 이해하고 효율적으로 처리하게 하기 위해서는 컴퓨터가 이해할 수 있도록 자연어를 적절히 변환할 필요가 있다.

•단어를 표현하는 방법에 따라서 자연어 처리의 성능이 크게 달라지기 때문에 이에 대한 많은 연구가 있었고 여러가지 방버들이 있다!

- 단어의 의미를 벡터화시킬 수 있는 Word2Vec과 Glove가 많이 사용
- 전통적 방법의 한계를 개선시킨 워드 임베딩(Word Embedding)방법론에 대해 배워보자!

희소표현 (Sparse Representation)

• 원-핫 인코딩을 통해서 나온 원-핫 벡터들은 표현하고자 하는 단어의 인덱스의 값만 1이고, 나머지 인덱스에는 전부 0으로 표현되는 벡터 표현 방법

- 벡터 또는 행렬(matrix)의 값이 대부분이 0으로 표현되는 방법을 희소 표현(sparse representation)

• 문제점

단어의 개수가 늘어나면 벡터의 차원이 한없이 커진다는 점

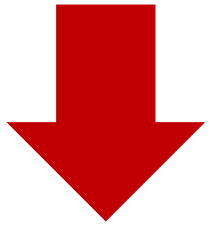
Ex) 강아지 = [0 0 0 0 1 0 0 0 0 0 0 0 ... 중략 ... 0] # 이 때 1 뒤의 0의 수는 9995개.

밀집표현 (Dense Representation)

- 밀집 표현은 벡터의 차원을 단어 집합의 크기로 상정하지 않습니다. 사용자가 설정한 값으로 모든 단어의 벡터 표현의 차원을 맞춥니다

- 이 과정에서 더 이상 0과 1만 가진 값이 아니라 실수값을 가지게 됩니다

Ex) 강아지 = [0 0 0 0 1 0 0 0 0 0 0 0 ... 중략 ... 0] # 이 때 1 뒤의 0의 수는 9995개.



Ex) 강아지 = [0.2 1.8 1.1 -2.1 1.1 2.8 ... 중략 ...] # 이 벡터의 차원은 128

워드 임베딩 (Word Embedding)

- 단어를 밀집 벡터(dense vector)의 형태로 표현하는 방법을 **워드 임베딩 (word embedding)**

- 이 밀집 벡터를 워드 임베딩 과정을 통해 나온 결과라고 하여 **임베딩 벡터(embedding vector)**

- 워드 임베딩 방법론으로는 LSA, Word2Vec, FastText, Glove 등이 있다

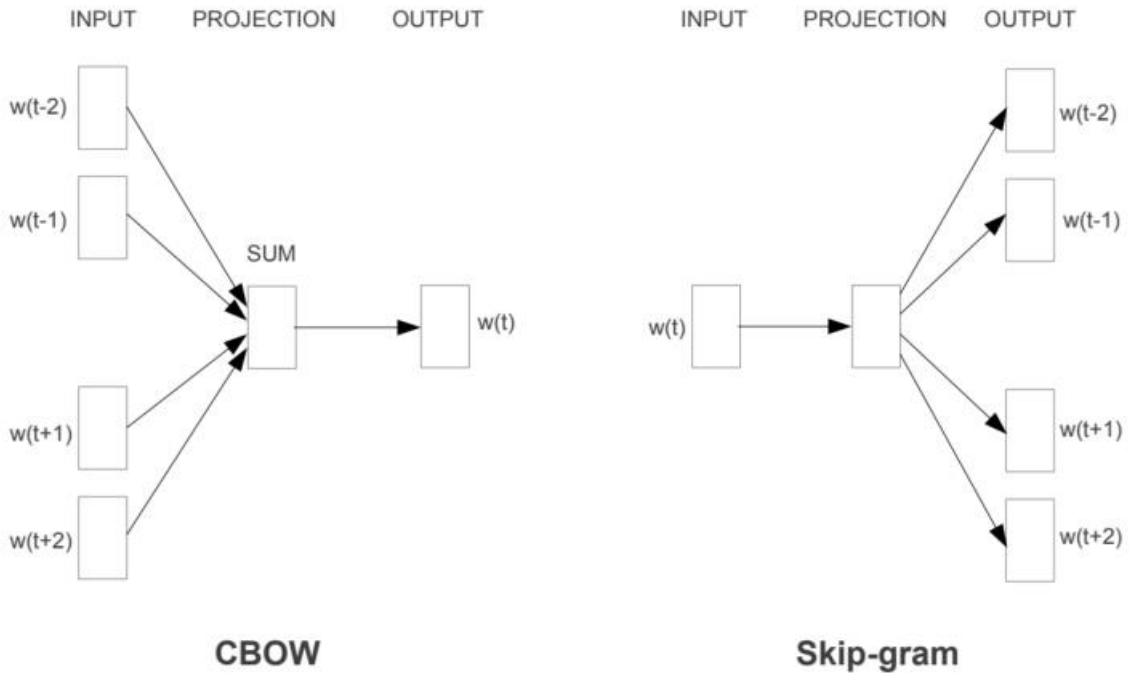
-	원-핫 벡터	임베딩 벡터
차원	고차원(단어 집합의 크기)	저차원
다른 표현	희소 벡터의 일종	밀집 벡터의 일종
표현 방법	수동	훈련 데이터로부터 학습함
값의 타입	1과 0	실수

Word2Vec

Word2Vec [Mikolov et al.2013]

- Objective:
  - 주변(context window)에 **같은 단어가 나타나는 단어일 수록** 비슷한 벡터 값을 가져야 한다.

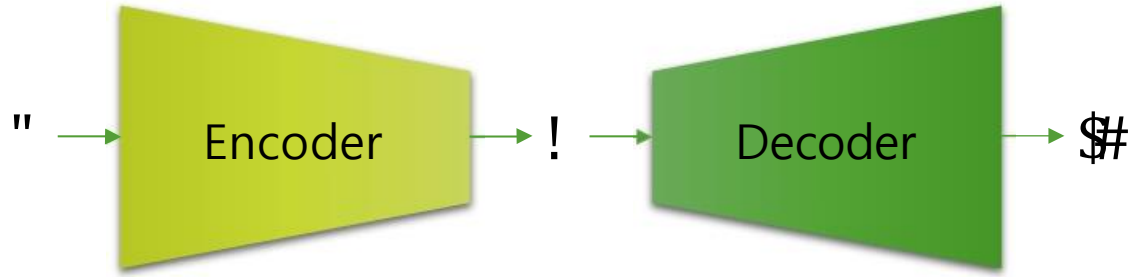
- 문장의 문맥에 따라 정해지는 것이 아님
  - context window의 사이즈에 따라 embedding의 성격이 바뀔 수 있다.



Skip-gram: Basic Concept

- 기본 전략
  - 주변 단어를 예측하도록 하는 과정에서 적절한 단어의 임베딩(정보의 압축)을 할 수 있다.
  - Non-linear activation func.이 없음

- 기본적인 개념은 오토인코더와 굉장히 비슷함
  - y를 성공적으로 예측하기 위해 필요한 정보를 선택/압축



Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

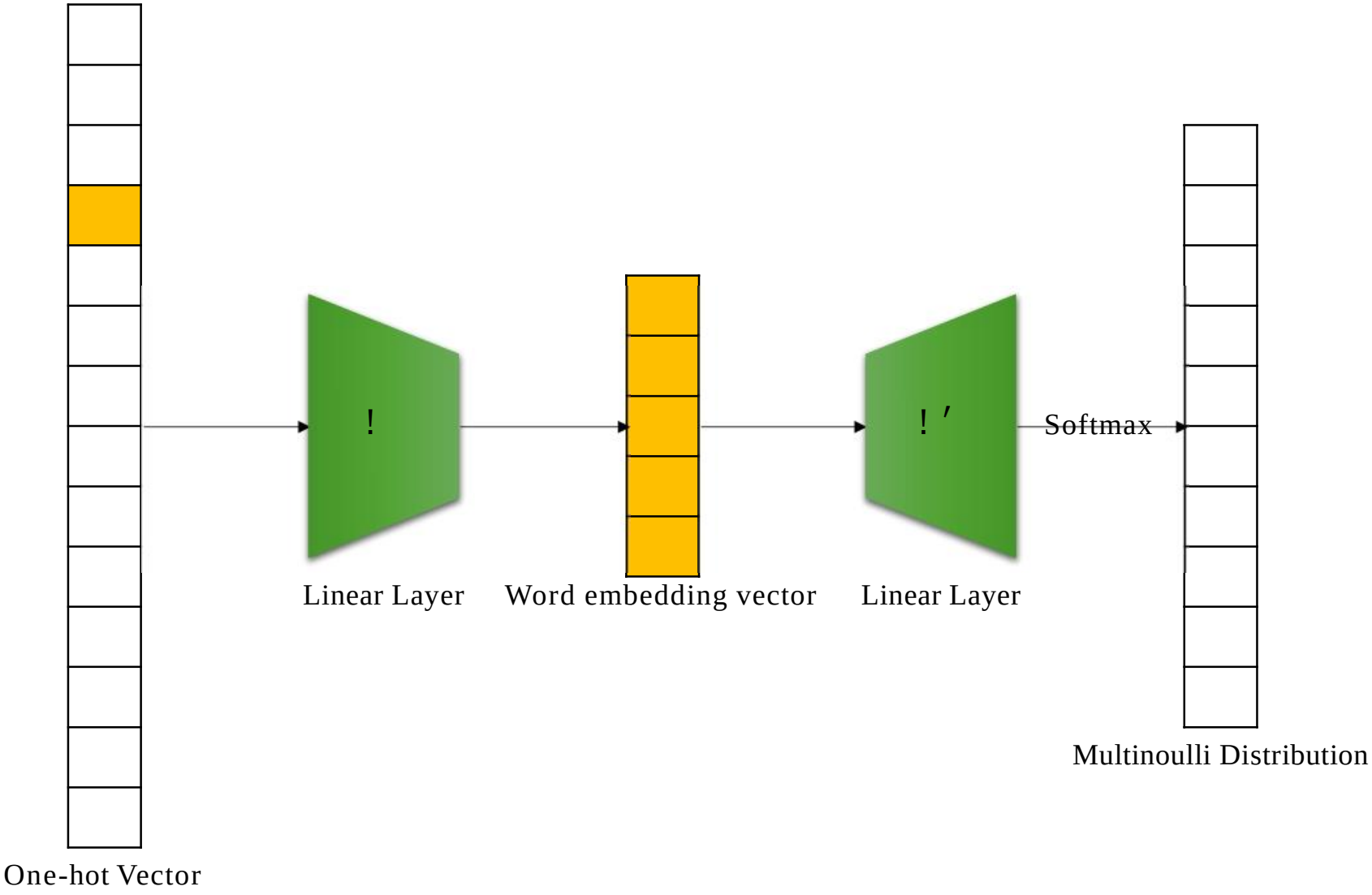
Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

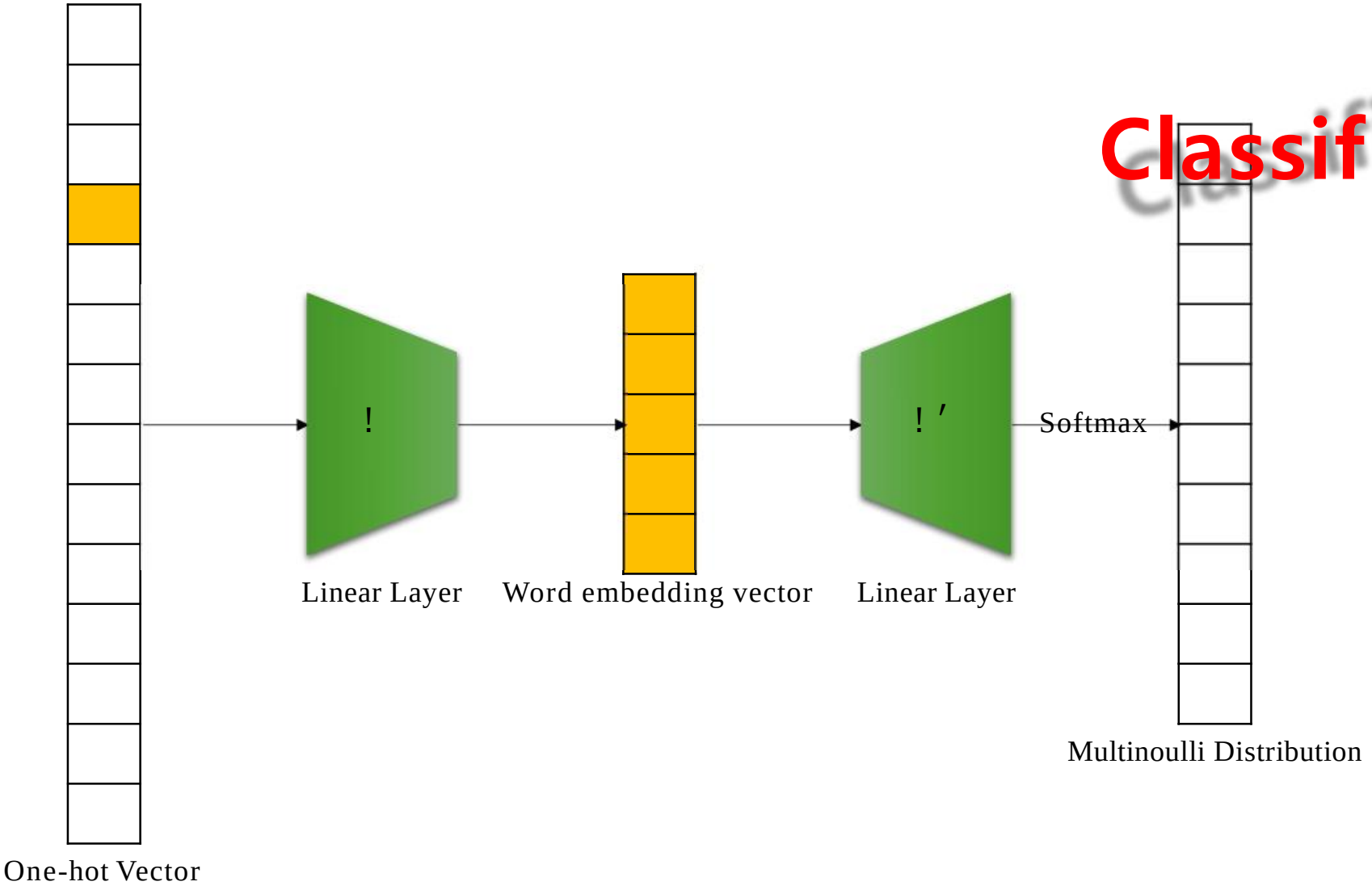
Skip-gram Example (|W|=5)

예식장	은	용궁	예식장	주례	는	문어	아저씨	.	
피아노	는	오징어	예물	은	조개껍데기	.			

Skip-gram


!
!′
Softmax
One-hot Vector
Linear Layer
Word embedding vector
Linear Layer
Multinoulli Distribution

Skip-gram


Classification
!
!′
Softmax
Linear Layer
Word embedding vector
Linear Layer
One-hot Vector
Multinoulli Distribution

Skip-gram

- 장점(at that time):
  - 쉽다.
  - 빠르다.
  - 비교적 정확한 벡터를 구할 수 있다.

- 단점(currently):
  - 헌데 느리다.
  - 출현 빈도가 적은 단어일 경우 벡터가 정확하지 않다.

Glove

GloVe

- Global Vectors for Word Representation
  [Pennington et al.,2014]

- 단어 x와 윈도우 내에 함께 출현한 단어들의 출현 빈도를 맞추도록 훈련

GloVe

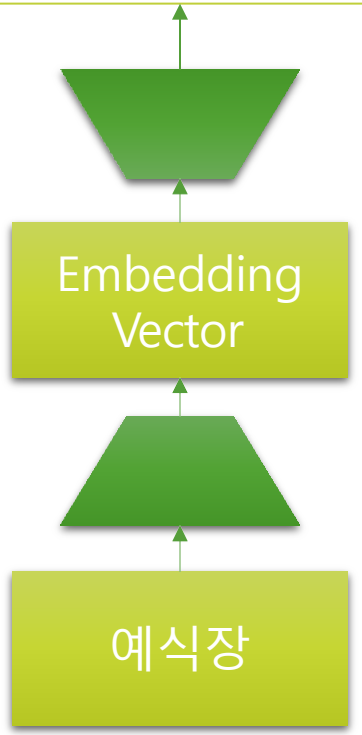
- Global Vectors for Word Representation
  [Pennington et al.,2014]

- 단어 x와 윈도우 내에 함께 출현한 단어들의 출현 빈도를 맞추도록 훈련

Regression

GloVe Example

오징어	예물	조개	피아노	은	는	결혼식	웨딩	주례	문어
5	17	2	23	31	27	41	29	34	1



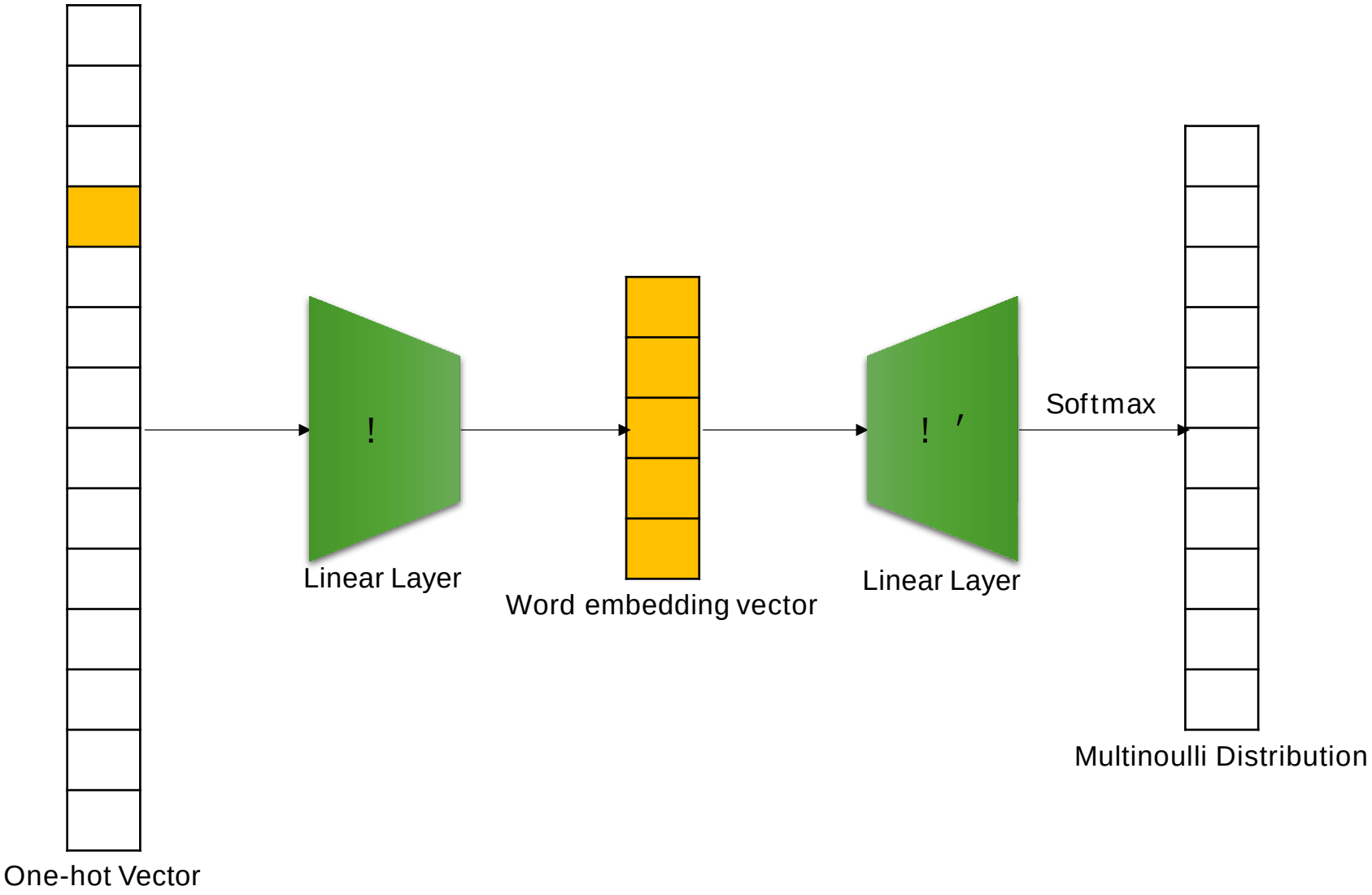
GloVe

- 출현 빈도가 적은 단어에 대해서는 loss의 기여도를 낮춤
  - 따라서 출현 빈도가 적은 단어에 대해 부정확해지는 단점을 보완

- 장점:
  - 더 빠르다
    - 전체 코퍼스에 대해 각 단어 별 co-occurrence를 구한 후, regression을 수행
  - 출현 빈도가 적은 단어도 벡터를 비교적 정확하게 잘 구할 수 있다.

FastText

Review: Skip-gram



FastText: Upgrade Version of Skip-gram

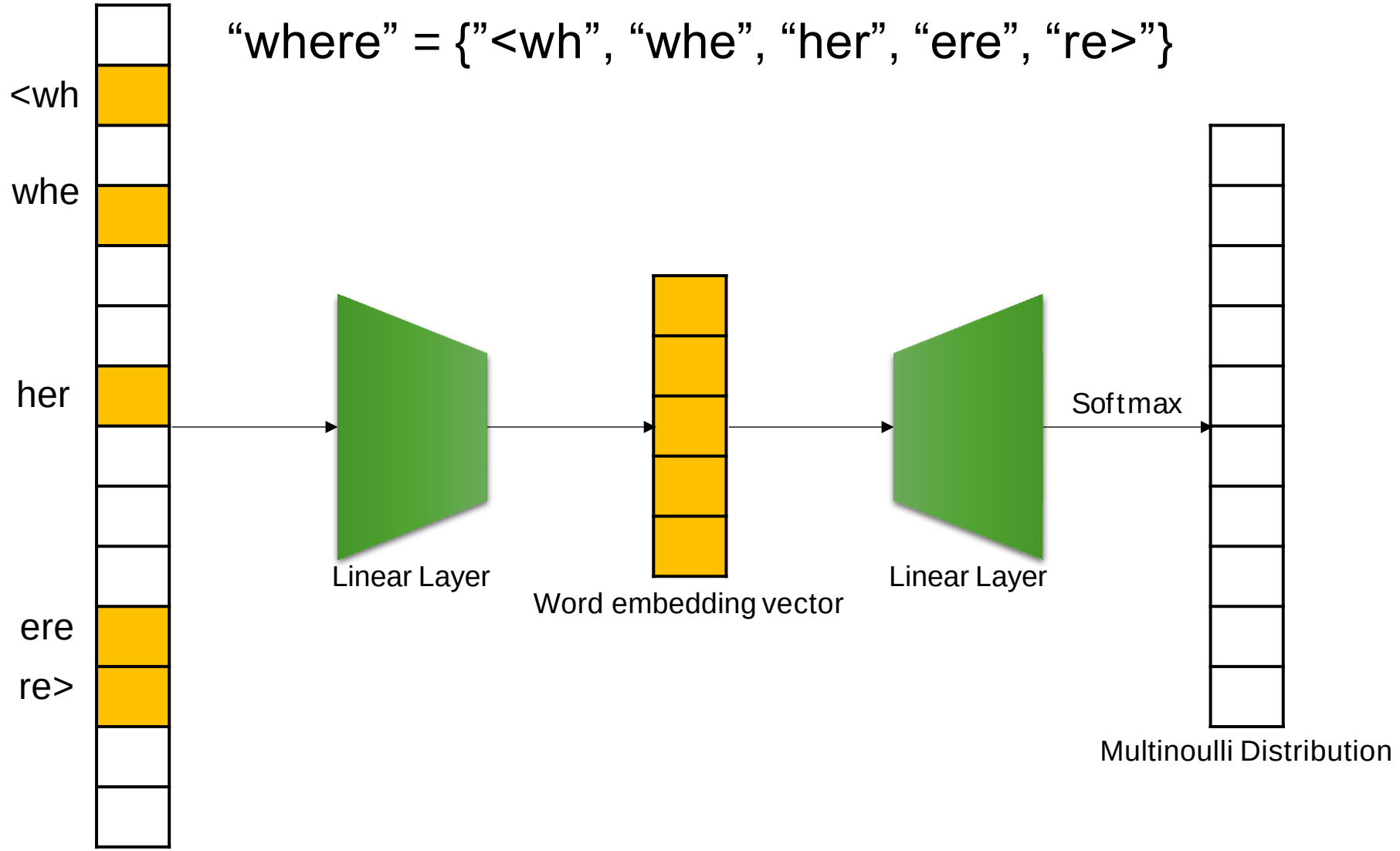
- Enriching Word Vectors with Subword Information [Bojanowski and Grave et al., 2016]

- Motivation:
  - 기존의 word2vec은 저빈도 단어에 대한 학습과 OoV에 대한 대처가 어려웠음

- FastText는 학습시,
  1) 단어를 subword로 나누고,
  2) Skip-gram을 활용하여, 각 subword에 대한 embedding vector에 주변 단어의 context vector를 곱하여 더한다.
  3) 이 값이 최대가 되도록 학습을 수행한다.

- 최종적으로 각 subword에 대한 embedding vector의 합이 word embedding vector가 된다.

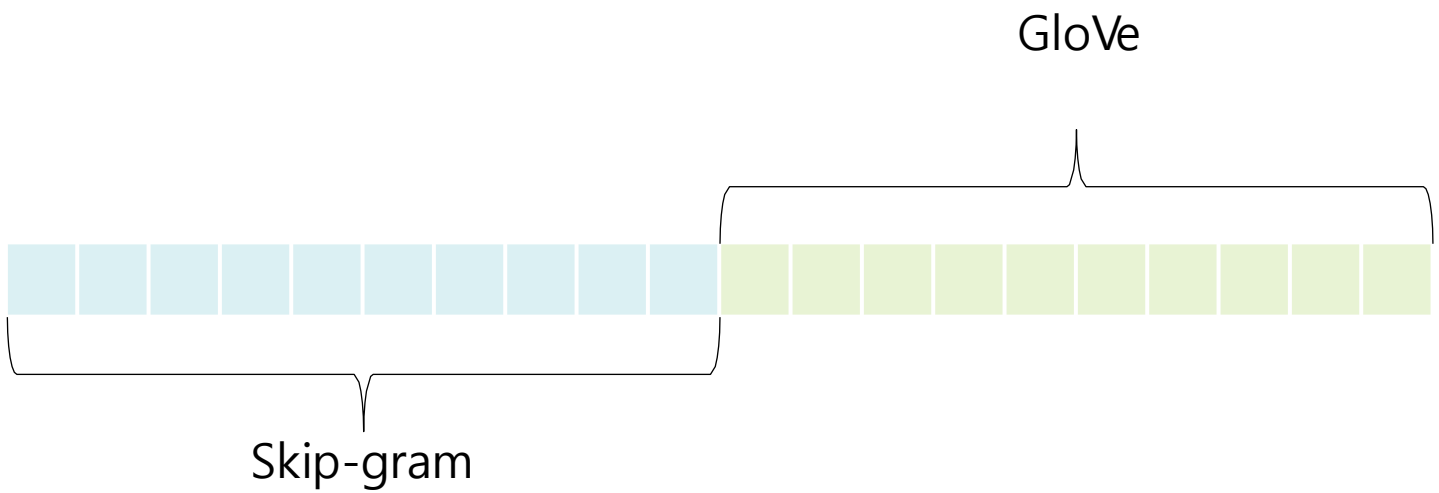
Example: where



Conclusion: Word Embedding

- 딱히 어떤 알고리즘이 더 뛰어나다고는 할 수 없다.
  - 구현이 쉽고 빠른 오픈소스를 사용하는 것이 낫다.

- 두 개의 다른 알고리즘 결과물을 concat하여 사용하기도



Open-source

- Gensim
  - https://radimrehurek.com/gensim/install.html
- GloVe
  - https://github.com/stanfordnlp/GloVe
- FastText
  - https://github.com/facebookresearch/fastText/
  - http://bit.ly/fasttext_win

수고하셨습니다.