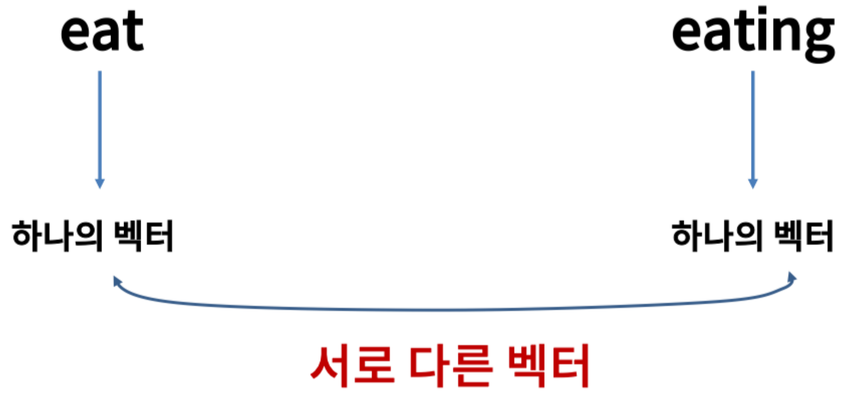
FastText

Word2Vec의 개량 알고리즘으로 Subword를 고려한 알고리즘

Word2Vec은 하나의 단어에 고유한 벡터를 할당하므로 단어의 형태학적 특징을 반영할 수 없다.

훈련 데이터에서 eat는 충분히 많이 등장해서, 학습이 충분히 잘 되었지만
eating은 잘 등장하지 않아서 제대로 된 임베딩 값을 얻지 못한다고 해보자.

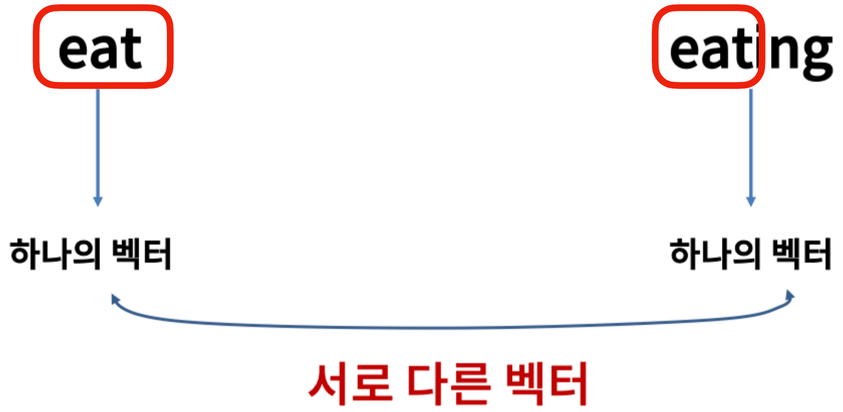


FastText

Word2Vec의 개량 알고리즘으로 Subword를 고려한 알고리즘

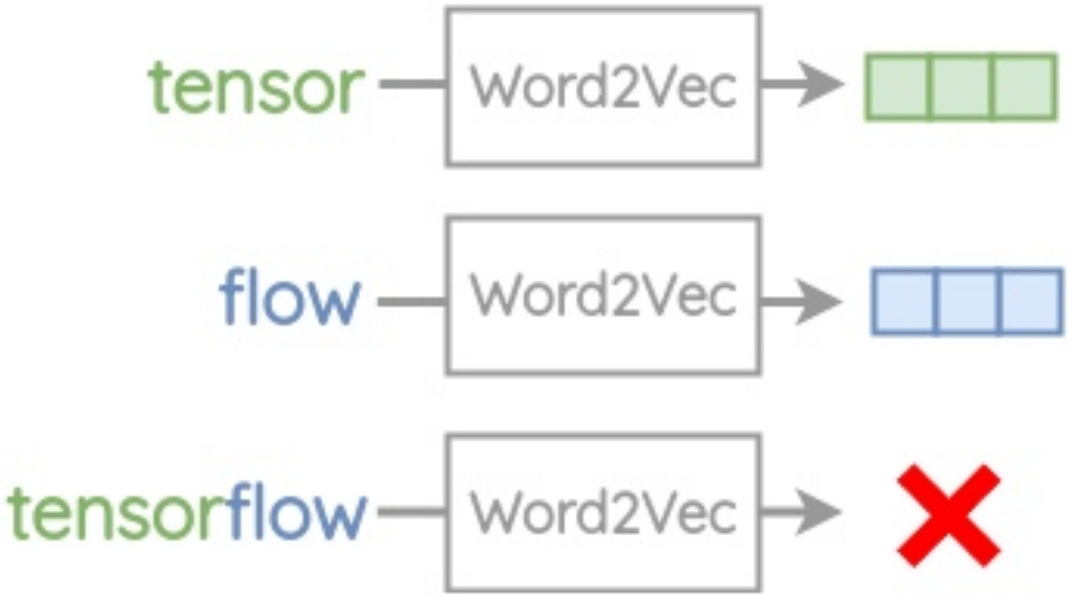
Word2Vec은 하나의 단어에 고유한 벡터를 할당하므로 단어의 형태학적 특징을 반영할 수 없다.

공통적인 내부 단어를 갖고 있는데, 이를 활용할 수는 없을까?



Word2Vec의 대표적인 문제점 두 가지

1. OOV(Out-of-Vocabulary) 문제



Word2Vec의 Vocabulary에
"tensor"와 "flow"가 있더라도,
"tensorflow"라는 단어가
Vocabulary에 없다면,
"tensorflow"의 벡터값을 얻을 수 없다.

Word2Vec의 대표적인 문제점 두 가지

2. 형태학적 특징을 반영할 수 없다.

다음의 단어들은 eat라는 동일한 어근을 가진다.
하지만 Word2Vec에서는
각 단어는 각 벡터의 값을 가질뿐이다.
이들을 고려할 수는 없을까?

FastText

FastText는 단어를 Character 단위의 n-gram으로 간주한다.
n을 몇으로 하느냐에 따라서 단어가 얼마나 분리되는지가 결정된다.
단어 'eating'을 예를 들어보자.

1. 단어 eating에 시작과 끝을 의미하는 <와 >를 추가한다.

FastText

FastText는 단어를 Character 단위의 n-gram으로 간주한다.
n을 몇으로 하느냐에 따라서 단어가 얼마나 분리되는지가 결정된다.
단어 'eating'을 예를 들어보자.

2. n-gram을 기반으로 단어를 분리한다. (Ex) n=3)

* n-gram은 단어를 묶는 경우의 수를 의미한다. n=3인 경우, 단어를 3개씩 묶는다.

FastText

FastText는 단어를 Character 단위의 n-gram으로 간주한다.
n을 몇으로 하느냐에 따라서 단어가 얼마나 분리되는지가 결정된다.
단어 'eating'을 예를 들어보자.

3. 주로 n은 범위로 설정해준다. Ex) n : 3~ 6

Word	Length(n)	Character n-grams
eating	3	<ea, eat, ati, tin, ing, ng>
eating	4	<eat, eati, atin, ting, ing>
eating	5	<eati, eatin, ating, ting>
eating	6	<eatin, eating, ating>

FastText

훈련 데이터를 N-gram의 셋으로 구성하였다면, 훈련 방법 자체는 SGNS와 동일하다.

단, Word가 아니라 subwords들이 최종 학습 목표이며

이들의 합을 Word의 vector로 간주한다.

FastText

FastText의 훈련 과정을 이해해보자.

여기서는 SGNS(Skip-gram with Negative Sampleing)을 사용한다.

우리의 목표는 중심 단어 eating과 주변 단어 am과 food로부터 SGNS를 학습하는 것이다.

FastText

앞서 언급하였듯이 단어 eating은

아래와 같이 n-gram들의 합으로 나타낸다.

FastText

중심 단어 eating을 기준으로 주변 단어 am과 food로 데이터셋을 구축한다.

Negative Sampling이므로 실제 주변 단어가 아닌 단어들도 필요하다.

FastText

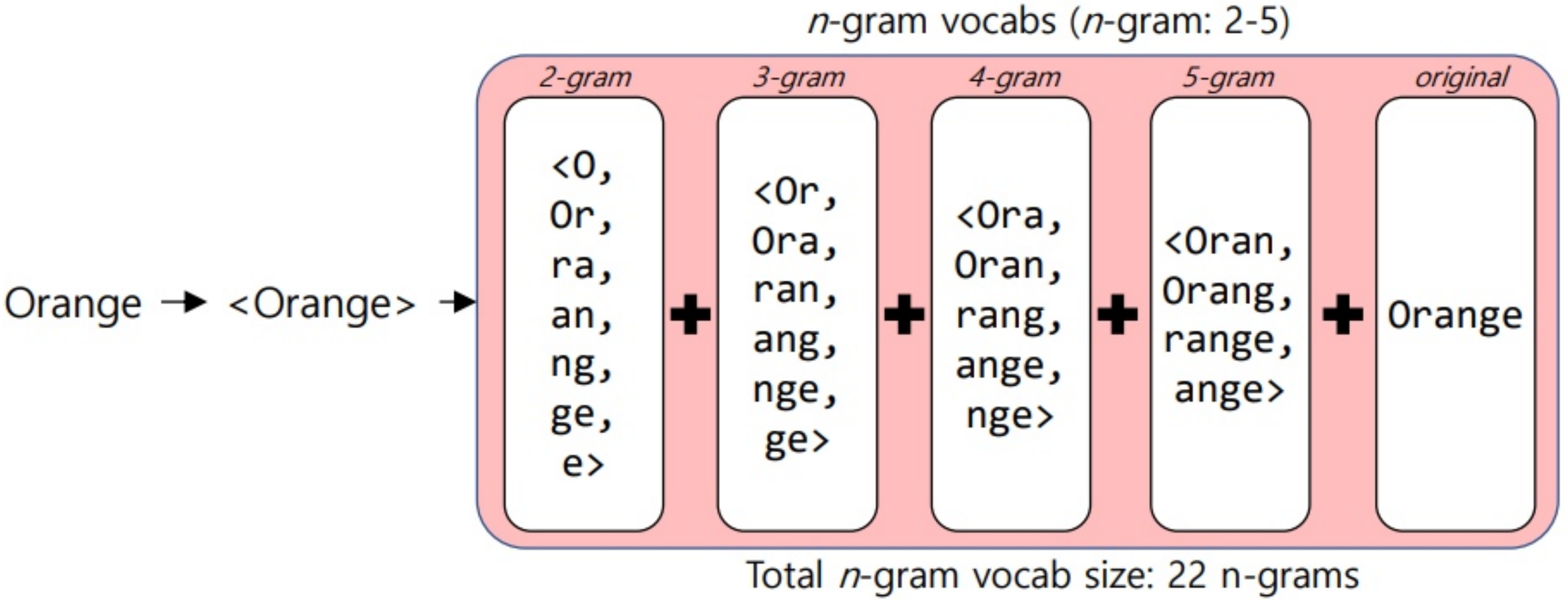
SGNS 자체는 동일하다.

eating과 am 그리고 eating과 food의 내적값에 시그모이드 함수를 지난 값은 10 되도록 학습.

eating과 paris 그리고 eating과 earth의 내적값에 시그모이드 함수를 지난 값은 00이 되도록 학습.

FastText

예를 들어 단어 Orange에 대해서 FastText를 학습했다고 해보자. n의 범위는 2-5로 한다.

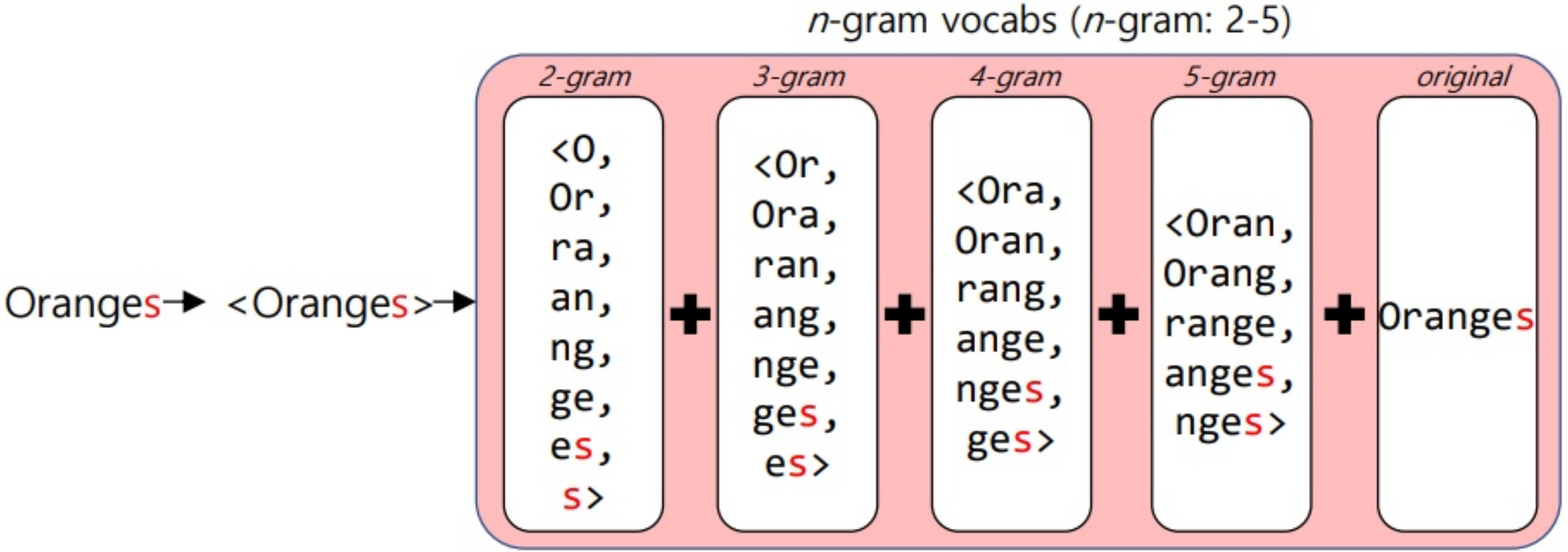


FastText

예를 들어 단어 Orange에 대해서 FastText를 학습했다고 해보자. n의 범위는 2-5로 한다.

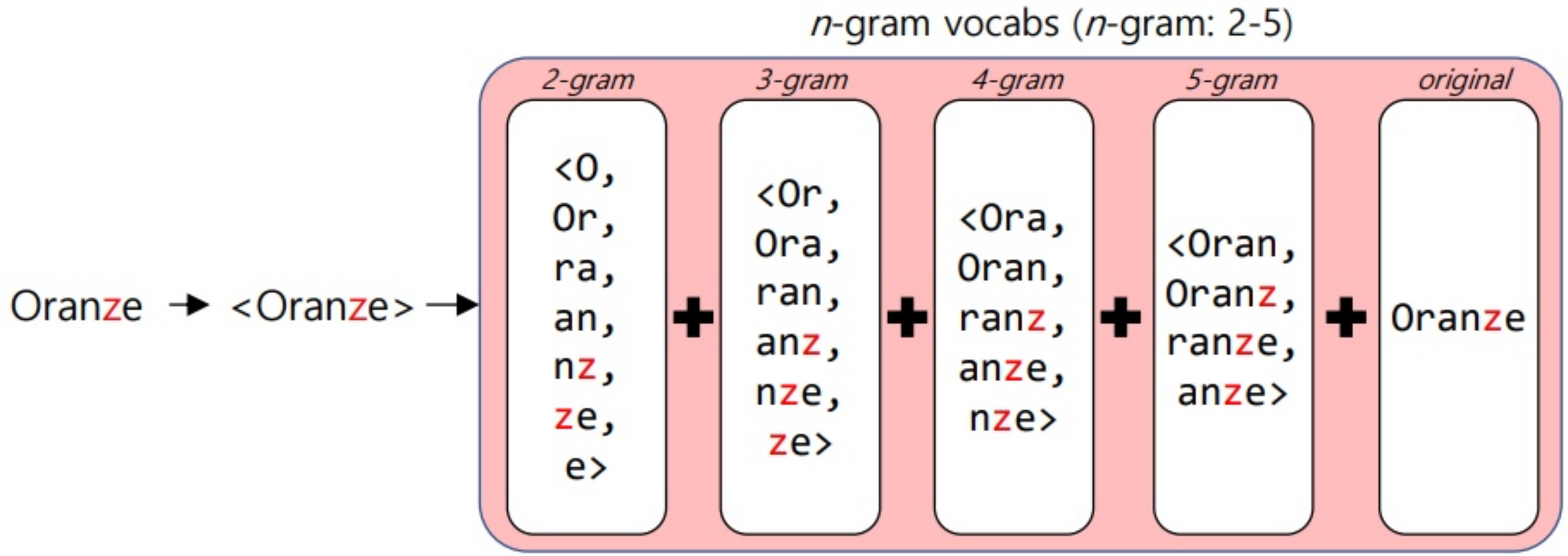
그 후 Oranges라는 OOV 또는 희귀 단어가 등장했다고 해보자.

Orange의 n-gram 벡터들을 이용하여 Oranges의 벡터값을 얻는다.



FastText

예를 들어 단어 Orange에 대해서 FastText를 학습했다고 해보자. n의 범위는 2-5로 한다.



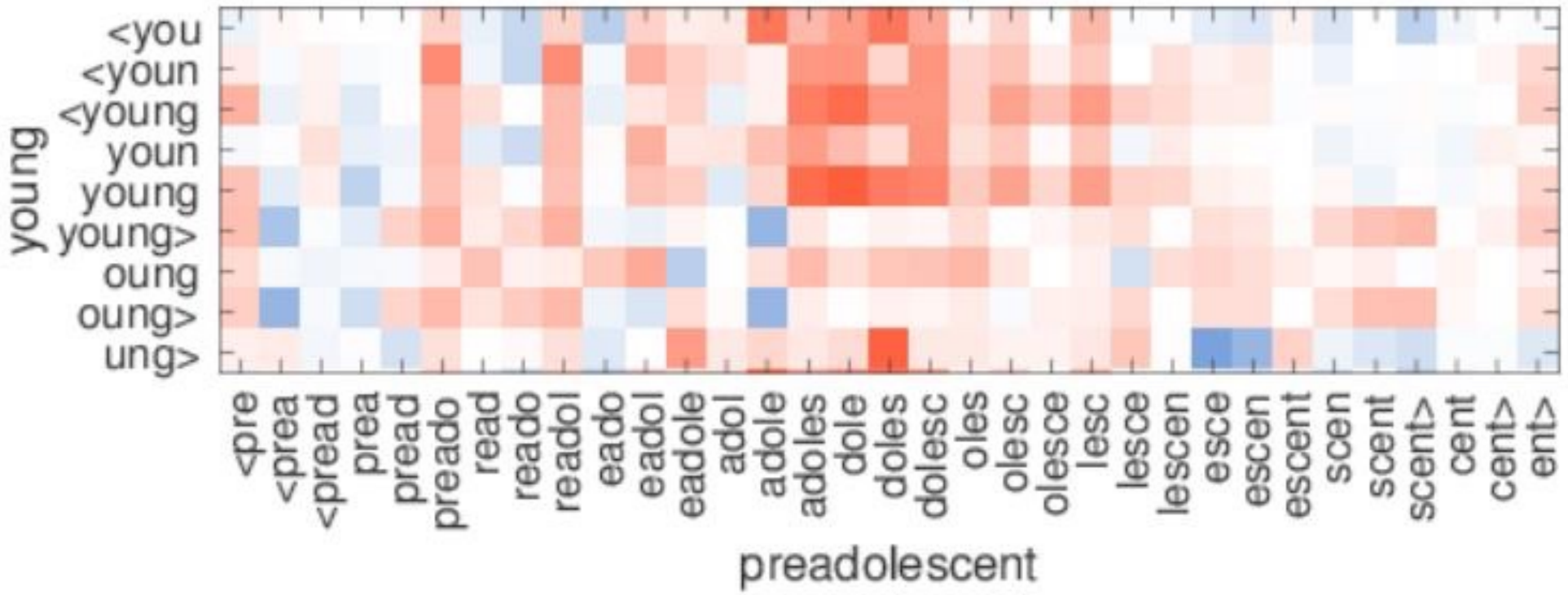
FastText

Q. 단어에 <,>를 왜 해주는 것인가요?

단어 양끝에 <,>를 해주지 않으면 실제로 독립적인 단어와 특정 단어의 n-gram인 경우를 구분하기 어렵습니다.
가령, **where**의 n-gram 중 하나인 **her**도 존재하지만 독립적인 단어 her 또한 Vocabulary에 존재할 수 있어요.
이 때, 독립적인 단어 her는 <her>가 되므로서 w**her**e 내의 her와 구분할 수 있습니다.

Subword Similarity

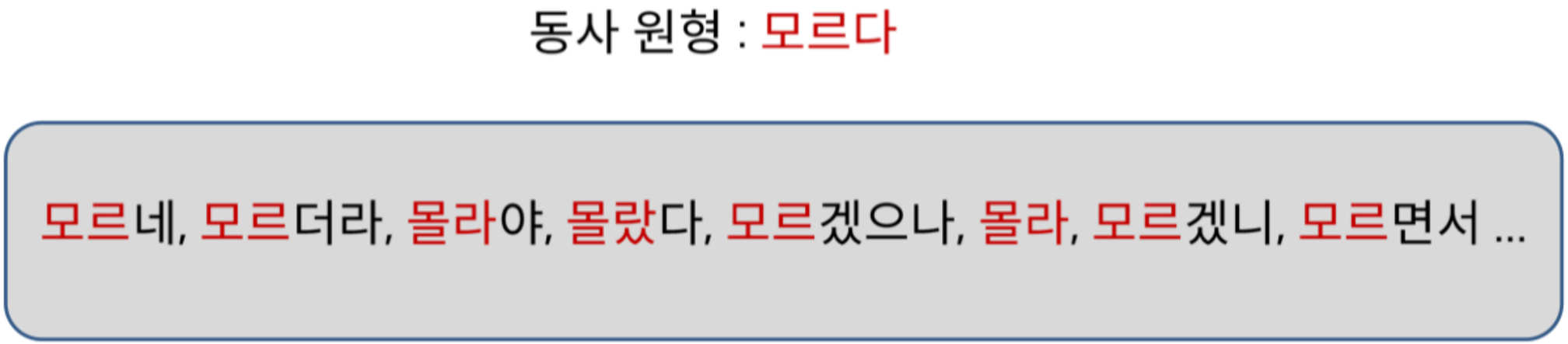
충분히 잘 학습된 FastText는 전체 Word가 아니라 Subword들의 유사도를 반영함을 확인할 수 있다



한국어에서의 FastText

한국어는 다양한 용언 형태를 가진다.

Word2Vec의 경우 다양한 용언 표현들이 서로 독립된 단어로 표현된다.



한국어에서의 FastText

한국어는 다양한 용언 형태를 가진다.

Word2Vec의 경우 다양한 용언 표현들이 서로 독립된 단어로 표현된다.

동사 원형 : 모르다

모르네, 모르더라, 몰라야, 몰랐다, 모르겠으나, 몰라, 모르겠니, 모르면서 ...

한국어의 경우에는 이를 대응하기 위해 FastText의 n-gram 단위를 자모 단위로 하기도 한다.

예시) ㅁㅗㄹ

Pre-trained Embedding Vs. Embedding layer

사전 훈련된 음절 단위 FastText를 이용해서 네이버 영화 리뷰 분류에 대한 실험 결과

- FastText(자모 단위): 네이버 영화 리뷰 + 위키피디아 : 86.90%
- FastText(음절 단위): 네이버 영화 리뷰 + 위키피디아 : 87.07%
- FastText(음절 단위): 네이버 영화 리뷰 + 위키피디아 + 나무 위키 : 87.15%
- Embedding layer: 네이버 영화 리뷰 : 84.05%

성능

	Accuracy
fastText	87.07%
Keras Embedding	84.05%

생각보다 차이가 나는군요.

성능 차이는 아무래도,

- fastText는 네이버 훈련 데이터 외에 한국어 Wikipedia 데이터도 사용했기 때문에 훨씬 커버리지가 큽니다.
- fastText의 알고리즘 특성상 OOV에 대해서 성능이 더 좋습니다. 네이버 코퍼스가 구어체가 많기 때문에 이 점도 영향을 주었을 것 같고요.

	Accuracy
한국어 위키피디아 + 네이버 영화	87.07%
한국어 위키피디아 + 네이버 영화 + 나무위키	87.15%

https://jins-sw.tistory.com/8

GloVe

Document Term Matrix와 마찬가지로 고차원을 가지는 행렬 표현 방법.

Window 크기에 따라서 행렬의 값이 결정된다는 특징을 가지고 있다.

- I like deep learning.
- I like NLP.
- I enjoy flying.

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

- 다음의 3개의 예시 문장에 대해서

윈도우 크기가 1일 때를 보여준다.

- 일반적으로는 윈도우 크기는 5-10이다.
- 동시 등장 행렬은 기본적으로 대칭 행렬.

GloVe

Document Term Matrix와 마찬가지로 고차원을 가지는 행렬 표현 방법.

Window 크기에 따라서 행렬의 값이 결정된다는 특징을 가지고 있다.

- I like deep learning.
- I like NLP.
- I enjoy flying.

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

동시 등장 행렬 그 자체를 모델의 입력으로 사용하면?

- 다음의 3개의 예시 문장에 대해서

윈도우 크기가 1일 때를 보여준다.

- 일반적으로는 윈도우 크기는 5-10이다.
- 동시 등장 행렬은 기본적으로 대칭 행렬.

GloVe

Document Term Matrix와 마찬가지로 고차원을 가지는 행렬 표현 방법.

Window 크기에 따라서 행렬의 값이 결정된다는 특징을 가지고 있다.

- I like deep learning.
- I like NLP.
- I enjoy flying.

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

동시 등장 행렬 그 자체를 모델의 입력으로 사용하면?

- 행과 열이 단어 집합(Vocabulary size)의 크기.
- DTM, OHE와 마찬가지로 저장 공간 낭비.
- 행렬이 매우 희소하다는 특징을 가진다.

GloVe

Document Term Matrix와 마찬가지로 고차원을 가지는 행렬 표현 방법.

Window 크기에 따라서 행렬의 값이 결정된다는 특징을 가지고 있다.

- I like deep learning.
- I like NLP.
- I enjoy flying.

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

동시 등장 행렬 그 자체를 모델의 입력으로 사용하면?

GloVe

Word2Veco 전체 코퍼스의 통계 정보를 충분히 활용하지 않는다는 점을 지적.

윈도우 기반의 동시 등장 행렬(Window Based Co-occurrence Matrix)를 사용한다.

Word2vec과 마찬가지로 Dense한 Vector를 학습.

윈도우 기반 동시 등장 행렬로부터 **동시 등장 확률**을 정의한다.

	x = solid	x = gas	x = water	x = fashion
$P(x\|\text{ice})$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(x\|\text{steam})$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$	8.9	8.5×10^{-2}	1.36	0.96

GloVe

Word2Veco 전체 코퍼스의 통계 정보를 충분히 활용하지 않는다는 점을 지적.

윈도우 기반의 동시 등장 행렬(Window Based Co-occurrence Matrix)를 사용한다.

Word2vec과 마찬가지로 Dense한 Vector를 학습.

윈도우 기반 동시 등장 행렬로부터 **동시 등장 확률**을 정의한다.

	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

GloVe

GloVe는 중심 단어 벡터와 주변 단어 벡터의 내적(dot product)이 동시 등장 확률이 되도록 학습한다.

$$dot\ product(w_i\ \tilde{w}_k) \approx\ P(k \mid i) = P_{ik}$$

내적 : 두 벡터의 곱을 의미하며 단어의 유사도를 계산하는 방법이므로
내적값이 높도록 한다는 것은 단어의 유사도가 높도록 한다는 의미다.

GloVe

GloVe는 중심 단어 벡터와 주변 단어 벡터의 내적(dot product)이 동시 등장 확률이 되도록 학습한다.

$$dot\ product(w_i\ \tilde{w}_k) \approx\ P(k \mid i) = P_{ik}$$

더 정확히는 다음을 만족하도록 학습한다.

$$dot\ product(w_i\ \tilde{w}_k) \approx\ log\ P(k \mid i) = log\ P_{ik}$$

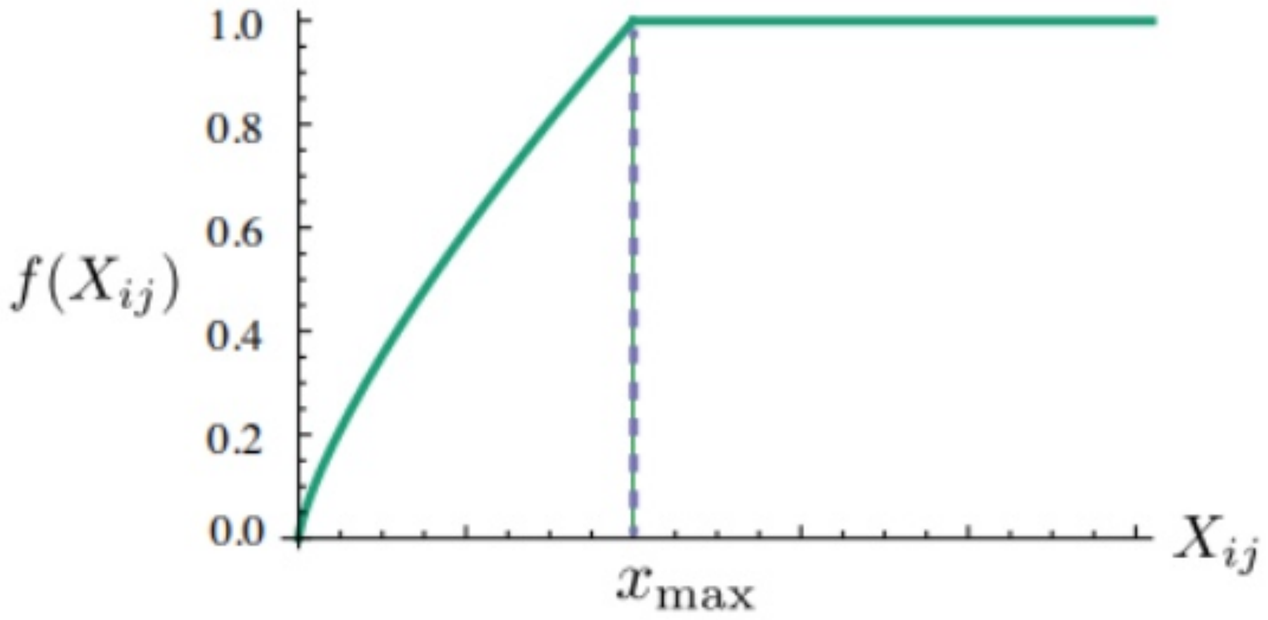
두 단어의 내적이 동시 등장 빈도의 로그값의 차이가 최소화 되도록 학습한다.

GloVe

- 두 단어의 내적이 동시 등장 빈도의 로그값의 차이가 최소화 되도록 학습한다.

$$Loss\ function = \sum_{m,n=1}^{V} f(X_{mn})(w_m^T \tilde{w}_n + b_m + \tilde{b_n} - logX_{mn})^2$$

$$f(x) = min(1,\ (x/x_{max})^{3/4})$$



Word Embedding의 한계

동형어, 다의어에 대해서는 제대로 훈련되지 않음.

단순히 주변 단어만을 고려하므로 문맥을 고려한 의미를 담고있지는 않음.