Distributed Representation Models

✓ Word2Vec : 방법론

IDEA Distributional Hypothesis : 주변단어가 비슷하면, 중심 단어의 의미가 비슷하다.

- Skip-gram

  자네가 무언가를 간절히 원할 때 온 우주는 자네의 소망이 실현되도록 도와준다네

  <연금술사>, 파울로 코엘료

- Continuous Bag-of-words (CBOW)

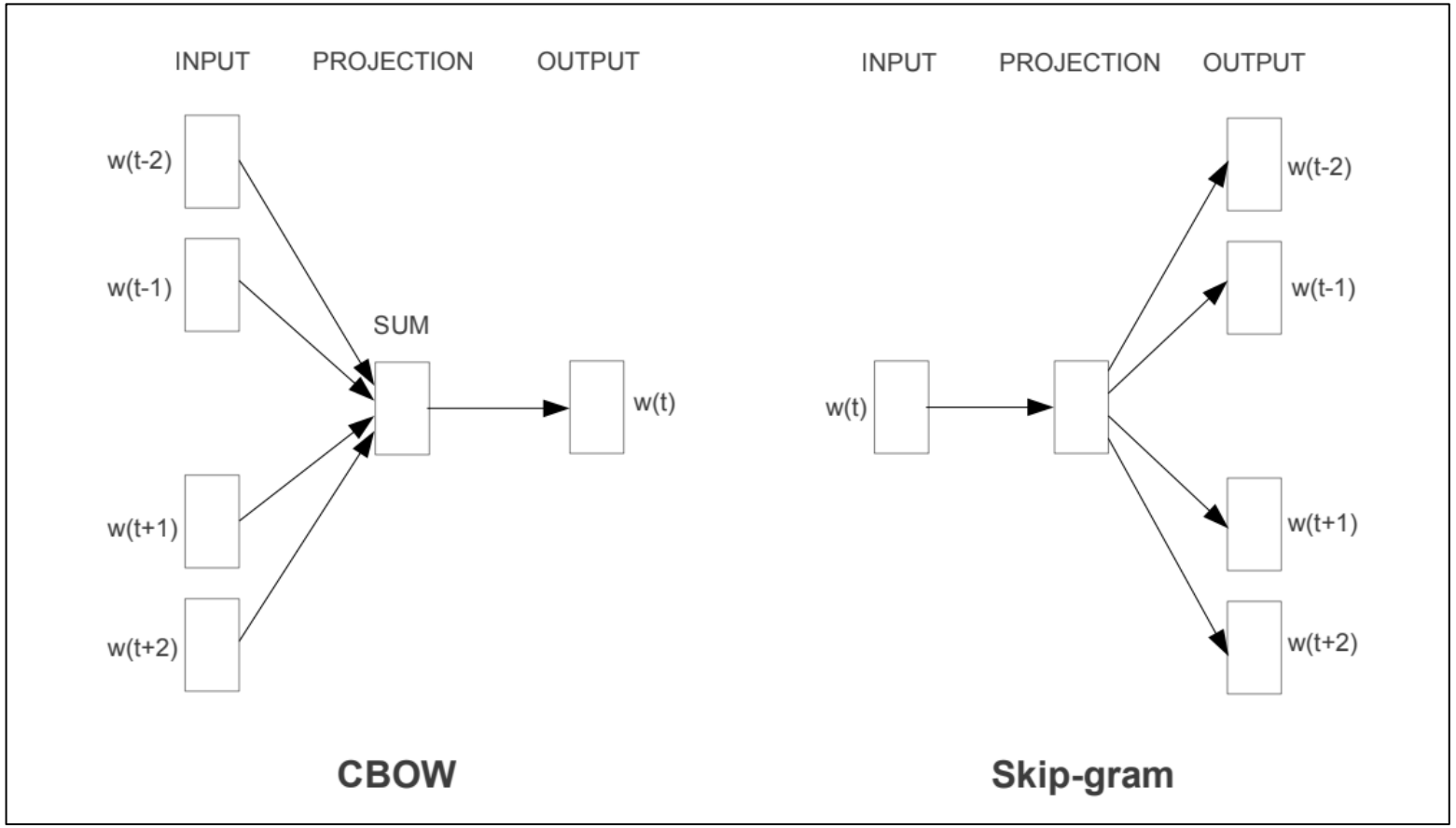
  자네가 무언가를 간절히 원할 때 온 우주는 자네의 소망이 실현되도록 도와준다네

  <연금술사>, 파울로 코엘료

Distributed Representation Models

✓ Word2Vec : Architecture

- Term
  - Center word : 중심 단어
  - Context word : 주변 단어
  - Window : 윈도우 크기만큼 주변단어로 설정
- Input size : (1, vocab_size)
- Projection layer
  - W size : (vocab_size, m)
  - W' size : (m, vocab_size)
- Output size : (1, vocab_size)

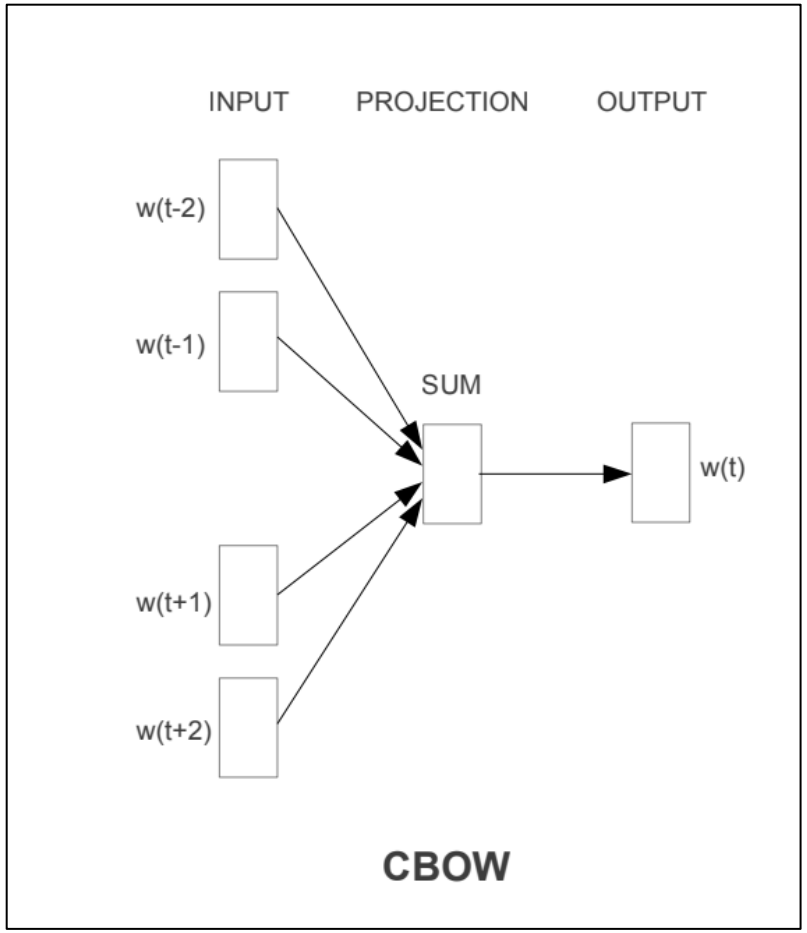


Efficient Estimation of Word Representations in Vector Space, Tomas et al. 2013

Distributed Representation Models

✓ Word2Vec : Continuous Bag-of-words (CBOW)

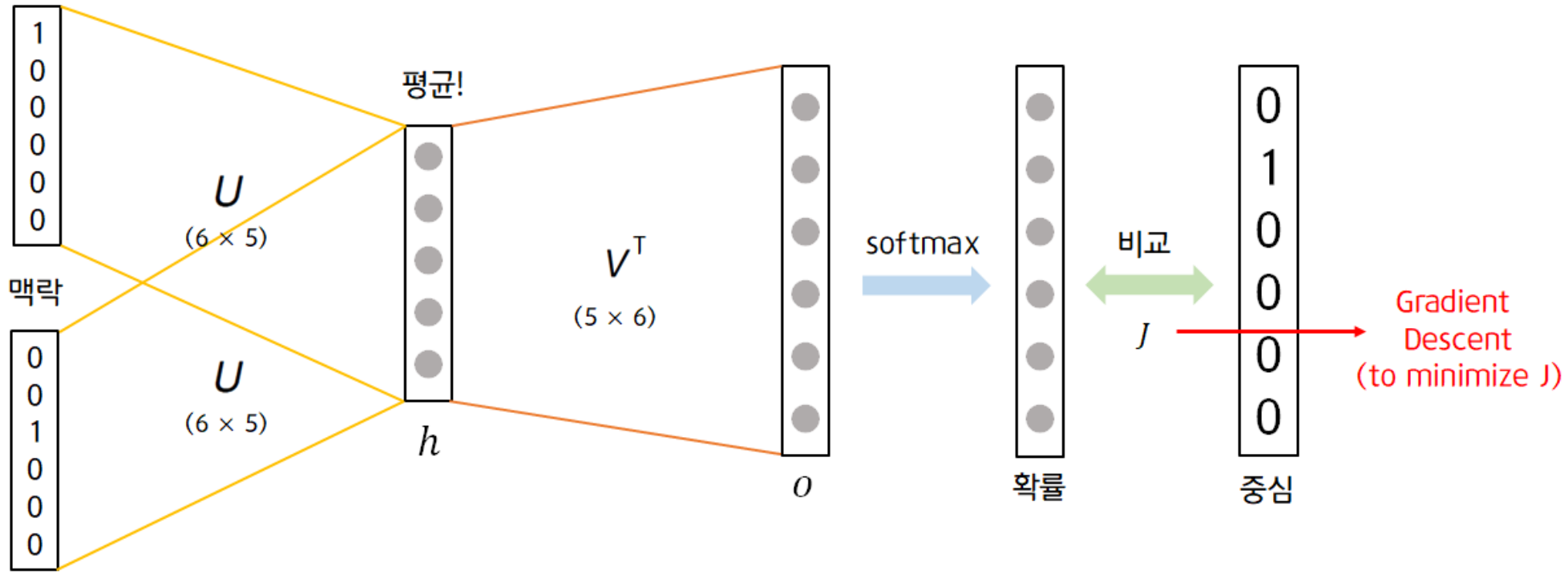
- Project Layer의 차원에서 M이 무엇인가요?
  - Word의 임베딩 차원입니다.
- Projection layer는 어떻게 사용되나요?
  - 모델이 학습되는 방법에 대해서 자세하게 알아봅시다 🤔
  - input은 one-hot vector로 구성됩니다. *(1,vocab_size)인 이유*
  - W를 통해서 m차원의 vector를 찾고 w(t-2), w(t-1), w(t+1), w(t+2)에 대해 각각 vector를 찾습니다.
  - 주변 단어 벡터들의 평균값을 구합니다. (sum과정)
  - W'을 통과하고 softmax를 거쳐 중심단어를 예측합니다.
  - **실제 정답 [0, 1, 0,⋯,0] , 예측된 값 [0.02, 0.8, 0.1, ⋯, 0.0]** 사이에 loss를 구해 W, W'을 업데이트 합니다.



Efficient Estimation of Word Representations in Vector Space, Tomas et al. 2013

Distributed Representation Models

✓ Word2Vec : Continuous Bag-of-words (CBOW)



https://velog.io/@tobigs-text1314/CS224n-Lecture-2-Word-Vectors-and-Word-Senses

Distributed Representation Models

✓ Word2Vec : Skip-gram

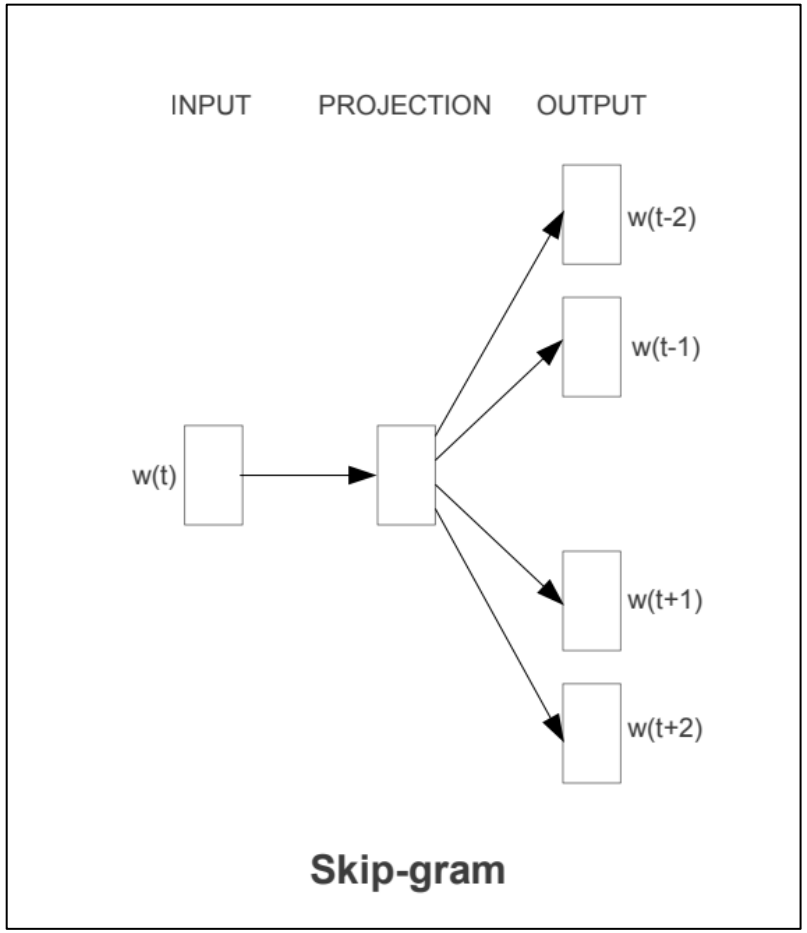
- 모델이 학습되는 방법에 대해서 자세하게 알아봅시다 !

1. Input : one-hot vector(1, vocab_size)
2. Projection layer W(vocab_size, m) : input word에 해당하는 (1,m) 벡터
3. Output layer : (1, vocab_size)로 주변단어 해당하는 단어를 예측
    - 주의할 점

      한꺼번에 w(t-2), w(t-1), w(t+1), w(t+2)를 예측하는 것이 아닙니다.

      각각에 대해서 예측하고 loss를 구합니다.

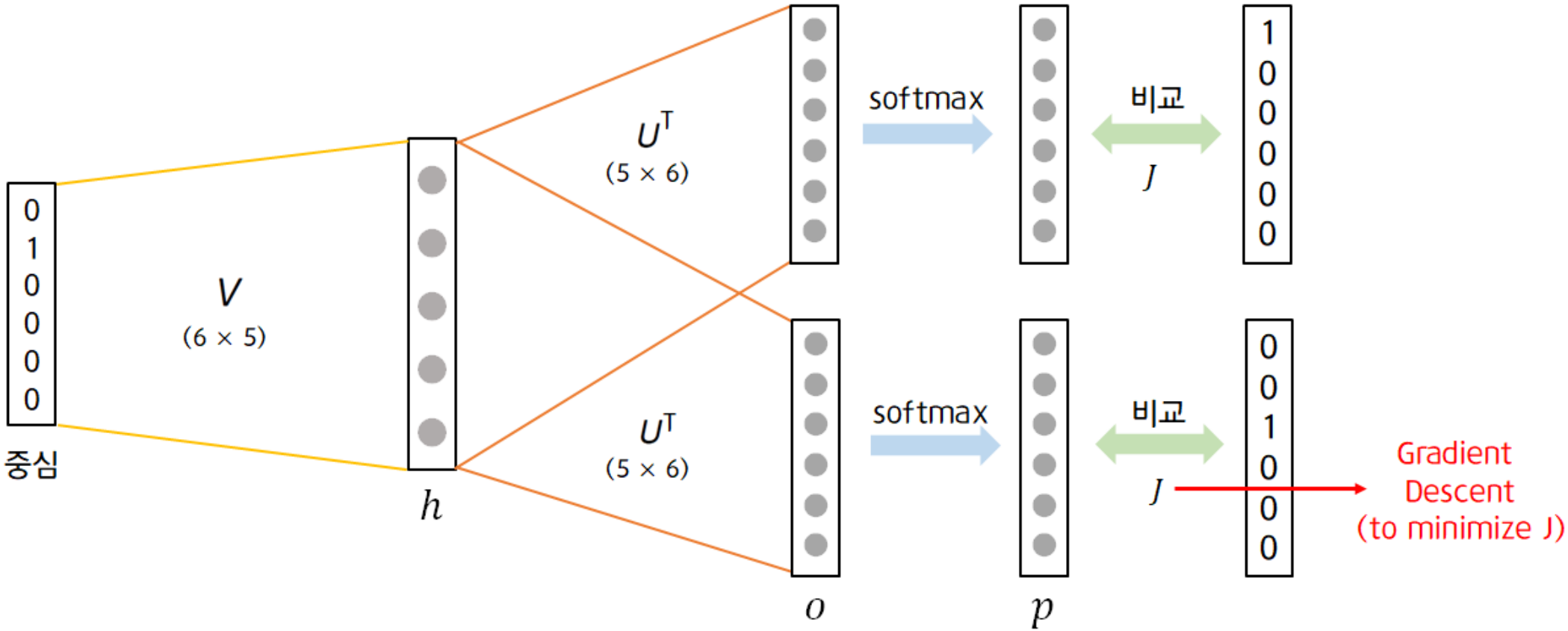
      그렇다면, loss를 구할 벡터쌍(예측값, 실제값)이 오른쪽 그림에서는 4개가 될 것입니다.



Efficient Estimation of Word Representations in Vector Space, Tomas et al. 2013

Distributed Representation Models

✓ **Word2Vec : Skip-gram**



https://velog.io/@tobigs-text1314/CS224n-Lecture-2-Word-Vectors-and-Word-Senses

Distributed Representation Models

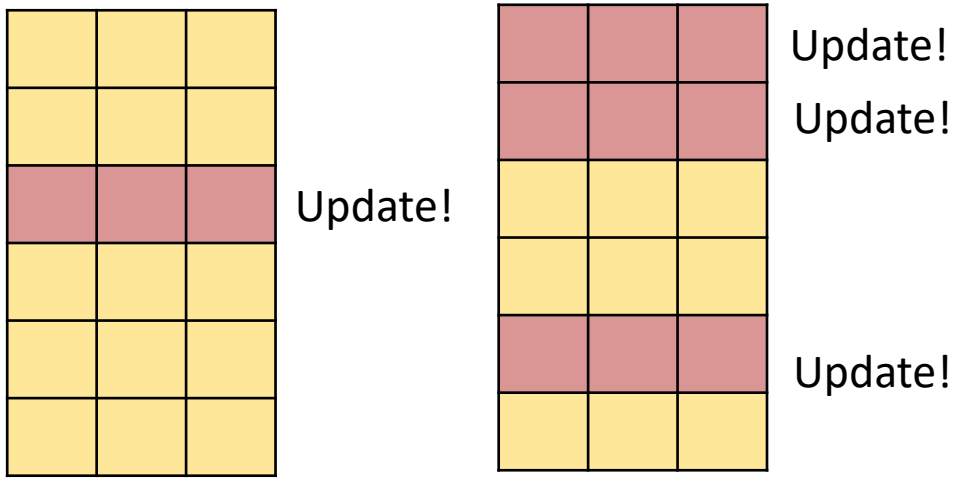
✓ Word2Vec : Objective

- 중심 단어에 대해 주변단어들은 독립
- 주변 단어가 주어졌을 때 중심 단어가 나올 확률을 최대화 하는 방향으로 학습됨.

$$p(o|c) = \frac{exp(u_o^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$$

Distributed Representation Models

- ✓ **Word2Vec**
- ✓ Output(1, vocab_size) = sparse vector 문제점
  - ✓ 0에 해당하는 위치에서는 더이상 update되지 않음. 따라서 불필요한 계산이 계속됨.
- ✓ 해결 : Negative Sampling
  - ✓ Vocab전체가 아닌 일부(5~20개)만 update하는 방법
  - ✓ center word와 context word embedding 레이어를 만든다.
  - ✓ center word와 window사이즈 내에 있는 단어는 레이블을 1, 그 외의 단어는 랜덤으로 뽑아서 0으로 레이블링한다.
  - ✓ Binary classification을 통해서 embedding layer를 업데이트 한다.



Distributed Representation Models

✓ CBOW vs Skip-gram

자네가 무언가를 간절히 원할 때 온 우주는 자네의 소망이 실현되도록 도와준다네

<연금술사>, 파울로 코엘료

CBOW : 1회 update

(우주는, 자네의, 실현되도록, 도와준다네) ⬅ 소망이

Skip-gram : 4회 update

소망이 ⬅ 우주는

소망이 ⬅ 자네의

소망이 ⬅ 실현되도록

소망이 ⬅ 도와준다네

Skip-gram > CBOW

Distributed Representation Models

✓ GloVe

✓ 카운트 기반 방법론

장점 : 코퍼스 전체에 대한 통계 정보를 반영

단점 : 의미적인 정보를 담지 못함.

✓ Word2vec : 예측 기반의 학습된 임베딩을 사용

장점 : 의미적 정보를 반영

단점 : 윈도우 내에서 학습되어 전체적인 정보를 반영하지 못함.

Glove = 카운트 기반 방법론 + word2vec

Distributed Representation Models

✓ **GloVe 함수(F)의 목적**

임베딩 된 중심단어와 주변 단어 벡터의 내적이 전체 코퍼스에서의 동시 등장 확률이 되도록 만드는 것

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

✓ **예시**

$$F(w_{delicious}, w_{boring}, w_{beer}) = \frac{P_{delicious,beer}}{P_{boring,beer}} = \frac{P(beer|delicious)}{P(beer|boring)} = \frac{1.9 \times 10^{-4}}{2.2 \times 10^{-5}} = 8.9$$

✓ **해석**

- delicious와 boring 주변에 beer가 등장할 확률
- **8.9** ➡️ (delicious, beer) 동시 등장 확률 > (boring, beer) 동시 등장 확률
- 동시 등장 확률 차이 : (delicious, beer) 내적값 – (boring, beer) 내적값

Distributed Representation Models

✓ GloVe 목적 함수의 변화 : word embedding을 받아서 관계를 보존해줄 수 있는 함수를 찾자

$$F(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

1) i번째 단어 벡터와 j번째 단어벡터의 차이 (목적함수 표현 변화)

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

2) i-j와 기준 k사이의 관계를 표현하기 위해 내적 사용 (목적함수 표현 변화)

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)}$$

3) 동시 등장 확률을 내적으로 표현 (목적함수를 구하는 식의 변화)

$$F(w_i^T \tilde{w}_k - w_j^T \tilde{w}_k) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)}$$

Distributed Representation Models

✓ GloVe 목적 함수의 변화 : word embedding을 받아서 관계를 보존해줄 수 있는 함수를 찾자

$$F(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)}$$

$$F(w_i^T \tilde{w}_k - w_j^T \tilde{w}_k) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)}$$

⭐ F의 조건

1) 대칭 $w_i \longleftrightarrow \tilde{w}_k$

2) 대칭으로서 가지는 특징 $X \longleftrightarrow X^T$

3) Homomorphism $F(X - Y) = \frac{F(X)}{F(Y)}$

F(A+B) = F(A)F(B)

$$F(x) = exp(x)$$

Distributed Representation Models

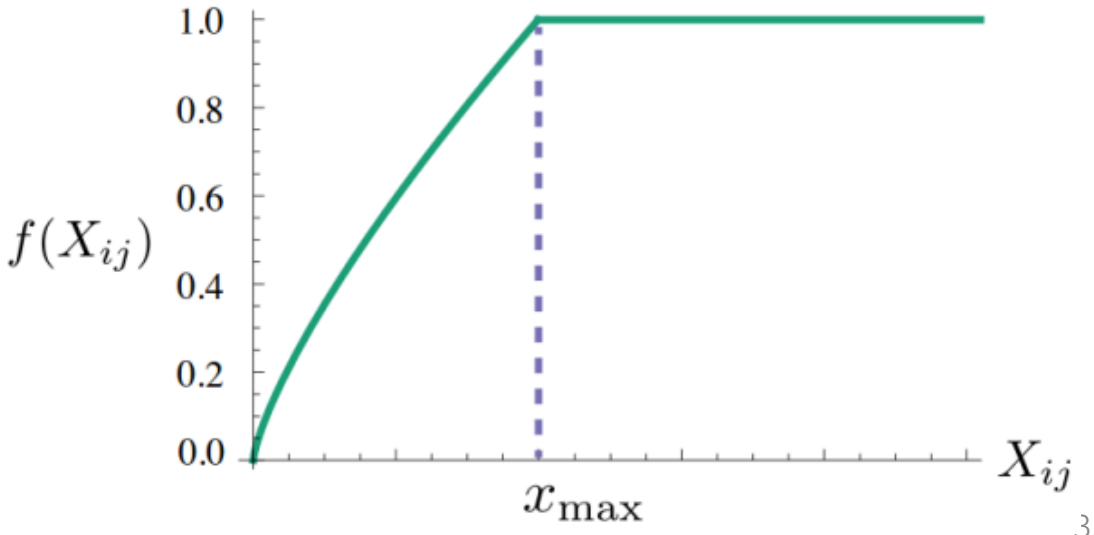
✓ GloVe 학습 과정

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

1. 전체 corpus에서 word co-occurrence 구하기
   - Shape = (vocab size, vocab size)
2. word_i와 word_j에 대한 임베딩 벡터의 내적 구하기
3. 내적과 co-occurrence의 log값 사이의 차이를 최소화하는 방향으로 학습

✓ Advanced

- 동시등장확률의 크기에 따라 가중치로 반영
- 동시등장확률이 지나치게 클 경우 가중치가 많이 반영되지 않도록 1로 조절함.



Distributed Representation Models

✓ **FastText**

✓ Word2Vec / Glove의 단점

- Out-of-Vocabulary : 학습 때 보지 못한 단어에 대해 대응이 불가함.
- Morphology : 형태소 변화에 대해서 대응하기 어려움.
  - 예) 자다 / 자는/ 잤다 / 자니

FastText : subword 사용 + word2vec

Distributed Representation Models

✓ FastText : subword

1. <natural>에서 만들 수 있는 모든 n-gram을 구하기 (보통 n=3~6)
    1. n=3 <na, nat, atu, tur, ura, ral, al>
    2. n=4 <nat, natu, atur, tura, ural, ral>
    3. n=5 <natu, natur, atura, tural, ural>
    4. n=6 <natur, natura, atural, tural>
    5. natural에 해당하는 임베딩

2. Natural의 임베딩 => word2vec,Glove보다 훨씬 vocab크기가 커짐.

    = N=3을 구성하는 조합에 해당하는 모든 임베딩

    + N=4을 구성하는 조합에 해당하는 모든 임베딩

    + N=5을 구성하는 조합에 해당하는 모든 임베딩

    + N=6을 구성하는 조합에 해당하는 모든 임베딩

    + natural에 해당하는 임베딩

Distributed Representation Models

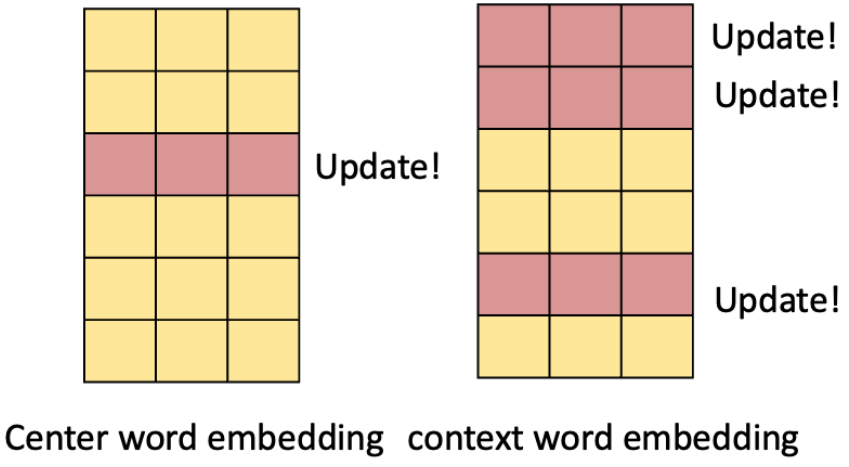
✓ FastText : 학습방식

✓ Skip-gram의 negative sampling을 사용

- Word2vec와 동일한 학습방법이지만 word embedding에 차이가 있는 것!

✓ **해결 : Negative Sampling**

- ✓ Vocab전체가 아닌 일부(5~20개)만 update하는 방법
- ✓ center word와 context word embedding 레이어를 만든다.
- ✓ Center word와 window사이즈 내에 있는 단어는 레이블을 1, 그 외의 단어는 랜덤으로 뽑아서 0으로 레이블링한다.
- ✓ Binary classification을 통해서 embedding layer를 업데이트 한다.



감사합니다