Word 2 Vec

- 주변에 같은 단어가 나타나는 단어일 수록 비슷한 벡터값을 가져야 한다.

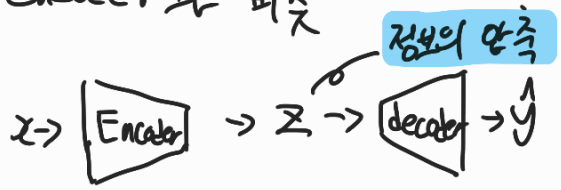
단어 → 모델 → fix된 임베딩 벡터

문장 문맥에 따라 결정 X → 코퍼스 —학습→ 모델

- CBOW, Skip-gram

Skip gram
- 주변단어 예측 → 정보의 압축(임베딩) 가능
- Non-linear activation func 없음
- Auto-encoder와 비슷

|W| = window_size



window_size

중심단어 기반 전체

○△ 중심 단어 X☆

5

|W| = 5

중심기반 단어 양옆

○△ 중심단어 X☆

2 2

|W| = 2

단어1, 단어2, 중심단어, 단어3, 단어4

classification

중심단어 —예측→ 단어 1, 2, 3, 4 → 정보의 압축 발생

↳ 강의에서는 하나씩 예측함

one-hot vector → Linear Layer → word embedding vector → Linear & softmax → ŷ

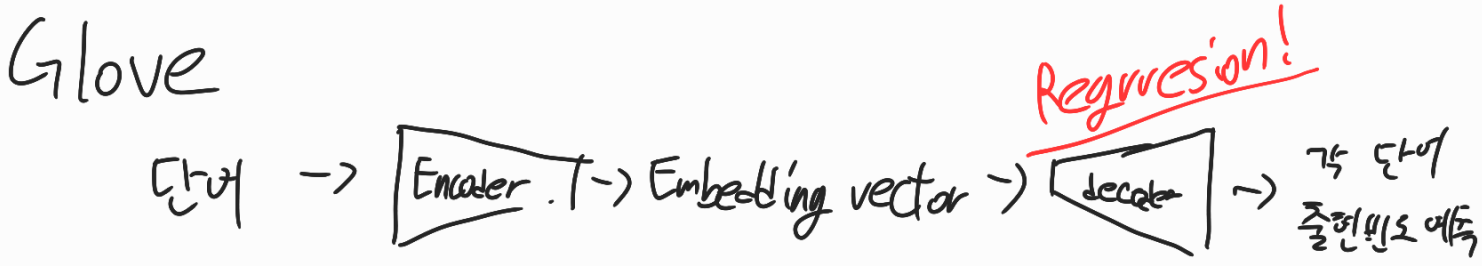
장점 (before)

- 쉽다
- 빠르다
- 비교적 정확한 벡터

단점 (현재)

- 느리다
- 출현빈도↓ 정확도↓

Glove



- 출현빈도가 적은 단어에 대해서는 loss 가중도를 낮춤
  ↳ 출현빈도가 적은 단어에 대해 부정확해지는 단점을 보완

- 장점
  - Skip-gram에 비해 빠르다
    Glove: 전체 코퍼스에 대해서 즉 단어별 co-occurrence를 구한 후 regression 수행
  - 출현빈도가 적은 단어에 대해서도 Good!

Fast text

facebook!

skip-gram의 upgrade version

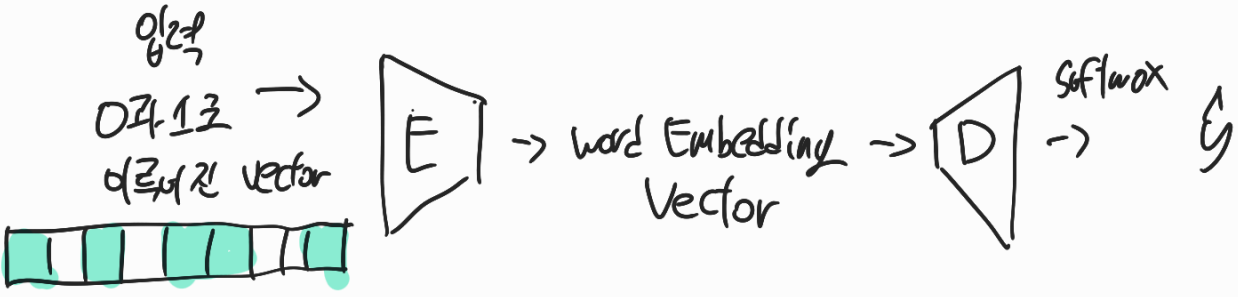
기존 skip-gram -> 저빈도 단어에 대한 학습 & OOV에 대한 대처 어려움

fast text

1) 단어를 subword로 분절
2) Skip-gram 학습 (subword Embedding vector × 주변 context vector) (단어)
3) 주변 단어에 대한 확률 최대 학습

분절: subword에 대한 Embedding vector의 합이 word Embedding vector

입력 vector -> ex) where = {<wh, whe, her, ere, re>}



Embedding vector summary

- 어떤 알고리즘이 확실히 우세하지는 않다.
  - 구현 쉽고 빠른 오픈소스 사용이 best
- 2개의 다른 알고리즘(ex. glove fasttext) concat 하기도

Open-source

- Gensim
- Glove
- fasttext

Skip-gram

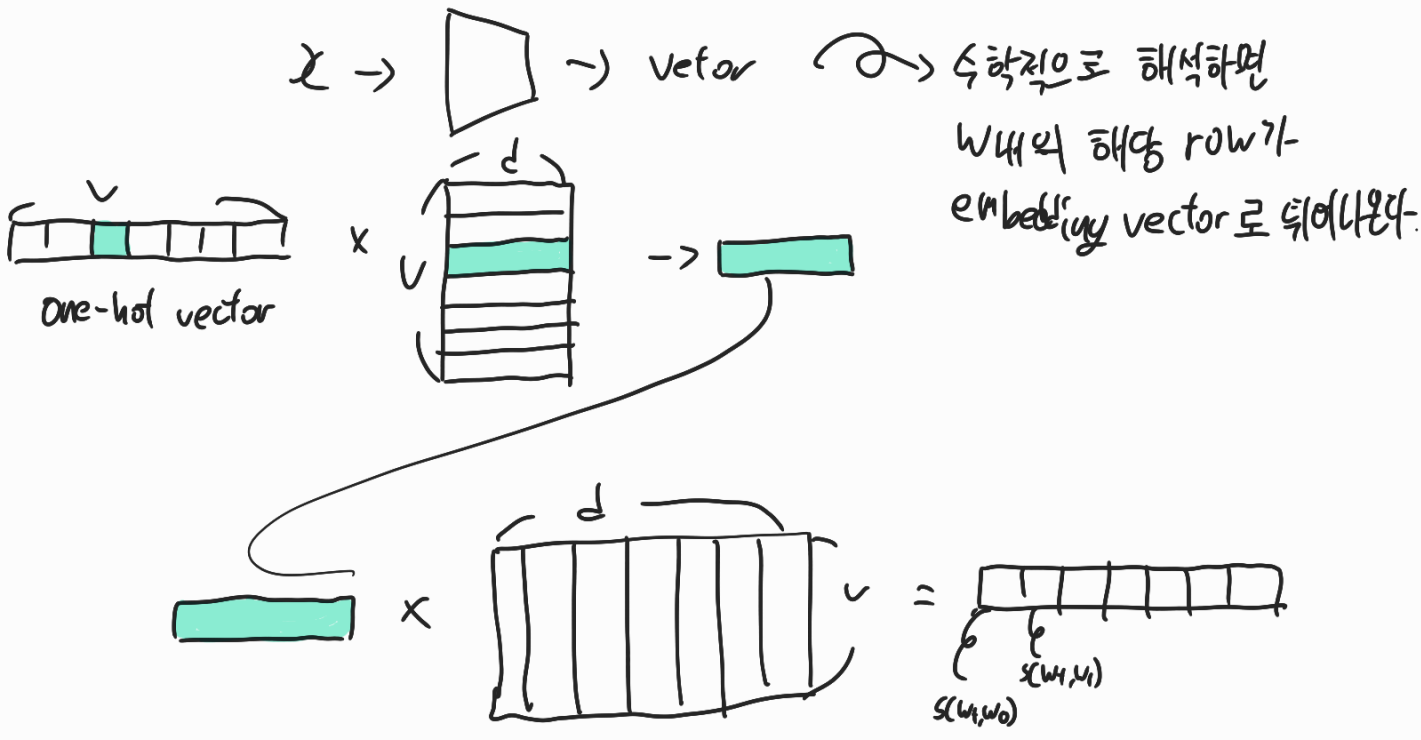
$$\sum_{t=1}^{T} \sum_{c \in C_t} \log p(w_c | w_t)$$

Maximize

- 주위 단어
- w_t가 주어졌을 때 w_c의 확률
- 현재의 단어
- t에서 윈도우 사이즈 만큼의 주위 단어

$$P(w_c | w_t) = \frac{e^{s(w_t, w_c)}}{\sum_{i=1}^{|V|} e^{s(w_t, w_i)}} \quad \text{where } s(w, w') = u_w^T v_{w'}$$

softmax



Negative Sampling (Softmax 대체용)

$$\log\left(1 + e^{-s(w_t, w_c)}\right) + \sum_{n \in N} \log\left(1 + e^{s(w_t, w_n)}\right), \quad \text{Minimize}$$

- $\log\left(1 + e^{-s(w_t, w_c)}\right)$: 정답
- $\sum_{n \in N} \log\left(1 + e^{s(w_t, w_n)}\right)$: 오답

정답

Glove

★ regression

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmin}} \sum_{x \in X} f(x) \times \| w'_x w_x - \log C_x \|_2^2$$

- $f(x)$: penalty
- $\| w'_x w_x - \log C_x \|_2^2$: MSE
- w'_x: decoder, w_x: encoder
- C_x: co-occurences with x vector (in other word y)

$$f(x) = \begin{cases} (\text{count}(x)/\text{thres})^{\alpha} & \text{if count}(x) < \text{thres} \\ 1 & \text{otherwise} \end{cases}$$

적게 나타나는 값은 loss를 줄이며 적은 단어에 대한 과적합을 줄임

Fast Text

input : sum of sub-word one-hot vector

$$\sum_{t=1}^{T} \sum_{c \in C_t} \log p(w_c | w_t), \quad P(w_c | w_t) = \frac{e^{s(w_t, w_c)}}{\sum_{j=1}^{|V|} e^{s(w_t, w_j)}}$$

$$s(w, w') = \sum_{y \in g_w} z_y^T v_{w'}$$

where g_w is a set of subword n-gram

z_y =