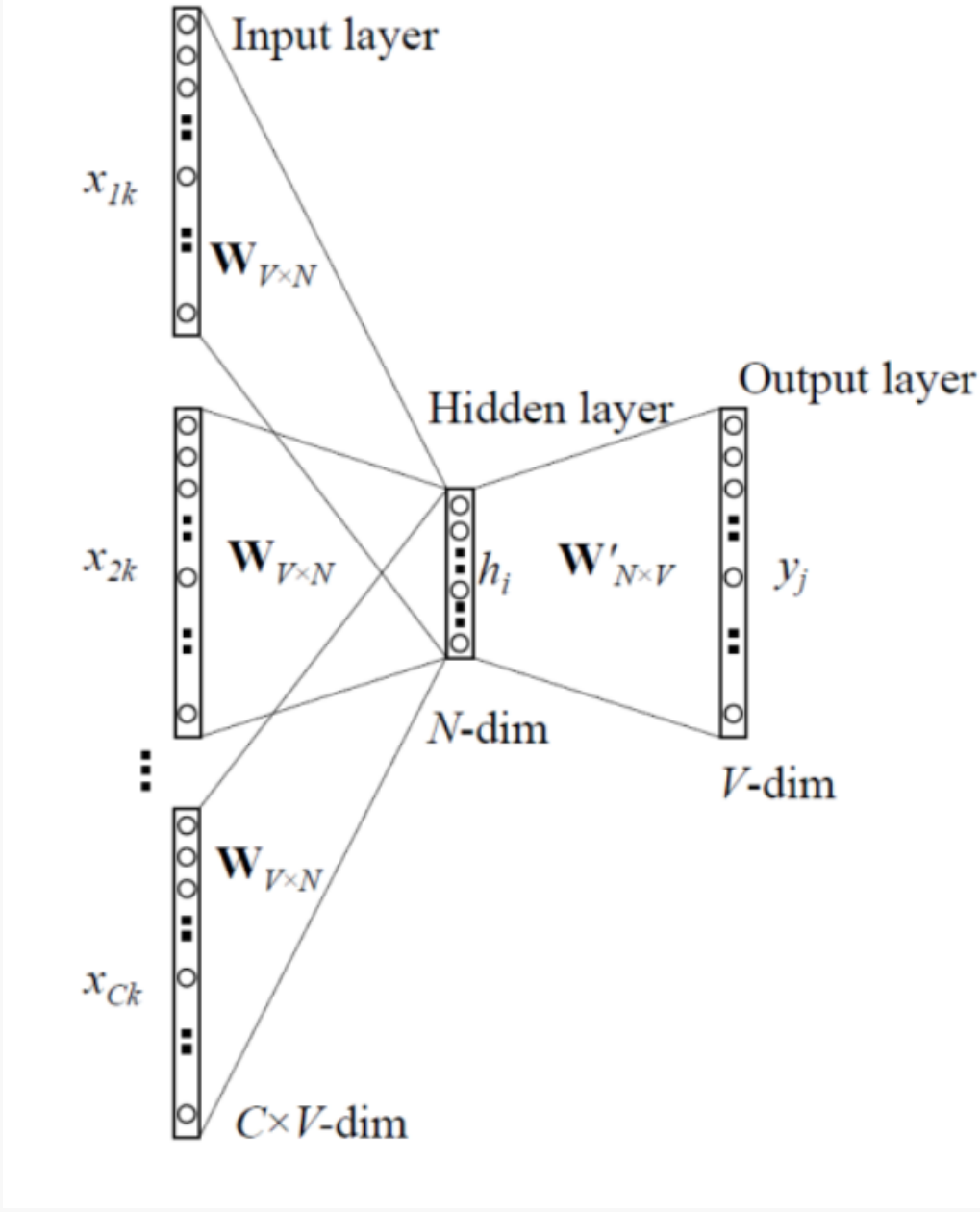
2장. 단어 수준 임베딩 : Word2Vec

CBOW, *Skip-gram*, *negative sampling*

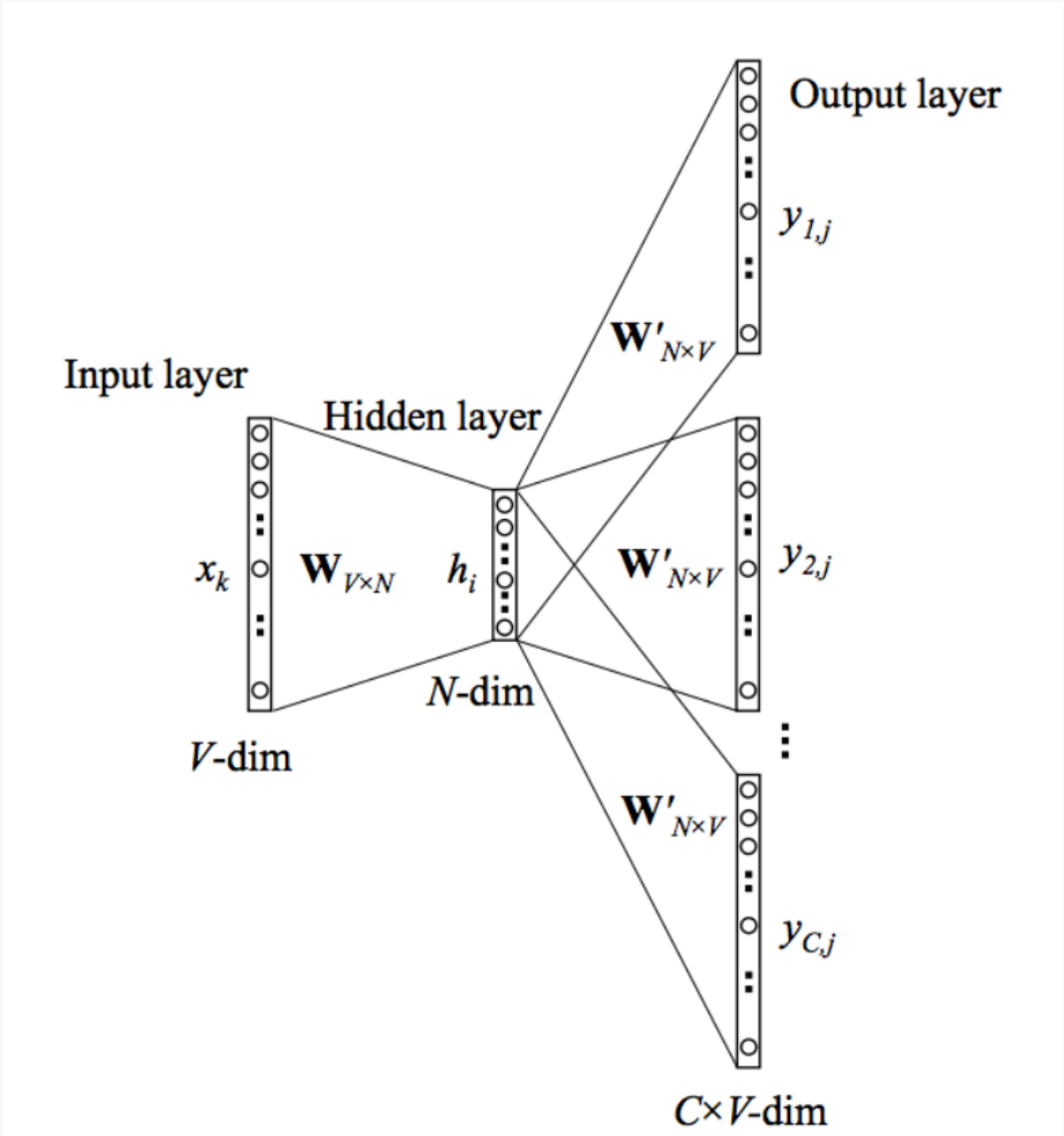
1. 모델 기본 구조

- CBOW : 주변에 있는 문맥 단어 $^{context\_word}$ 들을 가지고 타깃 단어^{target_word} 하나를 맞추는 과정에서 학습
    - 학습(입력, 출력) 데이터 : [문맥 단어 4개, 타깃 단어]인 한 쌍
- Skip-gram : 타깃 단어를 가지고 주변 문백 단어가 무엇일지 예측하는 과정에서 학습
    - 학습데이터 : [타깃 단어, 타깃 직전 두 번째 단어], [타깃 단어, 타깃 직전 단어], [타깃 단어, 타깃 직전 다음 단어], [타깃 단어, 타깃 다음 두 번째 단어]인 4쌍
    - Skip-gram이 같은 말뭉치로도 더 많은 학습 데이터를 확보할 수 있어 임베딩 품질이 CBOW보다 좋은 경향이 있다.
- CBOW 모델 구성도



CBOW 모델 구성도

- Skip-gram 모델 구성도



Skip-gram 모델 구성도

2. 학습데이터 구축 (focus on Skip-gram)

- 한국어 위키백과에 언급된 빨래

  `...개울가 {에서 속옷 빨래 를 하는} 남녀 ...`

- Skip-gram 모델의 학습 데이터 구축 : positive sample

target word	context word
빨래	에서
빨래	속옷
빨래	를
빨래	하는

positive sample : 타깃 단어{target word}와 그 주변에 실제로 등장한 문맥 단어(context word) 쌍

- Skip-gram 모델의 학습 데이터 구축 : negative sample

target word	target word	context word	context word
빨래	빨래	책상	커피
빨래	빨래	안녕	떡
빨래	빨래	자동차	사과
빨래	빨래	숫자	노트북

negative sample : 타깃 단어와 그 주변에 등장하지 않은 단어 (말뭉치 전체에서 랜덤 추출) 쌍.

위의 빨래를 target word로 한 sample들의 window size는 2 (positive sample을 만들 때 target word 앞뒤 2개씩만 고려)

- Negative Sampling 을 이용한 학습 방식
  - 기존 학습 방식 : 기존에 skip-gram 모델은 타깃 단어가 주어졌을 때 문맥단어가 무엇일지 맞추는 과정으로부터 학습 $\Rightarrow$ 어휘 집합에 속한 단어 수 만큼의 softmax 함수를 거쳐야하기 때문에 계산량이 큼
  - Negative sampling : **타깃단어와 문백 단어 쌍이 주어졌을 때 해당 쌍이 positive sample(+)인지 negative sampling(-)인지 이진분류하는 과정에서 학습** $\Rightarrow$ 1개의 positive sample과 k개의 negative sample만 계산(=정확히는 매 스텝마다 차원 수가 2인 시그모이드를 k+1회 계산)하면 되기 때문에 계산량이 기존 학습 방법의 계산량보다 훨씬 작다. (위 예제는 1개의 positive sample과 2(=k)개의 negative sample을 계산)
  - **작은 데이터에서는 k를 5~20, 큰 말뭉치에서는 k를 2~5로 하는 것이 성능이 좋다고 함**
- Negative sample을 뽑는 방법
  - Negative sampling 확률 : **말뭉치에 자주 등장하지 않는 희귀한 단어가 네거티브 샘플로 조금 더 잘 뽑힐 수 있도록 설계.**

$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=0}^{n} f(w_j)^{3/4}}$$

    - $f(w_i)$: 해당 단어가 말뭉치에서 차지하는 비율(해당 단어 빈도/어휘 집합 크기)
    - 예를 들어 말뭉치에 있는 단어가 용과 미르 둘 뿐이고 그 구성 비율은 각각 0.99, 0.01일 때 용과 미르과 negative sample로 뽑힐 확률은 아래의 수식과 같다. 원래대로라면 용은 0.99의 확률로 negative sample이 되었겠지만 0.97로 살짝 **낮아짐**. 미르는 0.01에서 0.03으로 살짝 **높아짐**.

$$P(\text{용}) = \frac{0.99^{3/4}}{0.99^{0.75} + 0.01^{0.75}} = 0.97$$

$$P(\text{미르}) = \frac{0.01^{3/4}}{0.99^{0.75} + 0.01^{0.75}} = 0.03$$

  - subsampling 확률 : **자주 등장하는 단어는 학습에서 제외하는 방법**

    skip-gram모델은 말뭉치로부터 엄청나게 많은 학습 데이터 쌍을 만들어 낼 수 있기 때문에 고빈도 단어의 경우 등장 횟수 만큼 모두 학습시키는 것이 비효율적이라고 본 것.

$$P_{subsampling}(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

    $f(w_i)$: w_i의 빈도, t : 하이퍼파라메터 (기존 논문에선 t는 10^{-5}로 설정)

만일 $f(w_i)$가 0.01로 나타나는 빈도 높은 단어(ex : 조사 `은/는`)는 위 식으로 계산한 $P_{subsampling}(w_i)$가 0.9684나 돼서 해당 단어가 가질 수 있는 100번의 학습 기회 가운데 96번 정도는 학습에서 **제외**됨. 반대로 등장 비율이 적어 $P_{subsampling}(w_i)$가 0에 가깝다면 해당 단어가 나올 때마다 빼놓지 않고 학습을 시키는 구조. 즉, **서브 샘플링은 학습량을 효과적으로 줄여 계산량을 감소**시키는 전략.

3. 모델 학습

- Skip-gram 모델은 입력 (타깃 단어, 주변 단어 쌍)이 주어졌을 때 positive sample(+)인지 아닌지를 예측하는 과정에서 학습

- t, c가 positive sample(=t 주변에 c가 존재)일 확률

$$P(+|t,c) = \frac{1}{1+exp(-U_tV_c)}$$

  입력이 실제로 positive sample이라면 **위 수식에 정의된 조건부확률 최대화**해야 함 $\rightarrow$
  $exp(-U_tV_c) \downarrow \quad \rightarrow U_t, V_c$의 내적값 $\uparrow$

  U : 타깃 단어에 대응되는 행렬, U_t: 타깃 단어에 해당하는 U의 행 벡터

  V : 문맥 단어에 대응되는 행렬, V_c : 문맥 단어에 해당하는 V의 열 벡터

- t, c가 negative sample일 확률

$$P(-|t,c) = 1 - P(+|t,c) = \frac{exp(-U_tV_c)}{1+exp(-U_tV_c)}$$

  입력이 실제로 negative sample이라면 **위 수식에 정의된 조건부확률 최대화**해야 함 $\rightarrow$
  $exp(-U_tV_c) \uparrow$
  $\rightarrow U_t, V_c$의 내적값 $\downarrow$

- Skip - gram 모델의 로그우도 함수

$$L(\theta) = logP(+|t_p, c_p) + \sum_{i=1}^{k} logP(-|t_{n_i}, c_{n_i})$$

  모델 파라미터인 θ를 한 번 업데이트할 때 1개 쌍의 positive sample과 k개 negative sample이 학습됨

  skip-gram은 위의 로그우도 함수를 최대화하는 과정에서 학습 $\rightarrow$ **말뭉치의 분포 정보를 단어 임베딩에 함축**

  모델 학습이 완료되면 **(1) U만 사용**하거나 **(2) U, V를 이어 붙여 단어임베딩**으로 사용

출처 : 한국어 임베딩 (책)