

# Week 2

☰ 태그	
☰ 열	

## Lecture 2 – Word Vectors and Word Senses

### Word2Vec

- Word2Vec maximizes objective function by putting similar words nearby in space
- 벡터를 빼고 더하는 과정 등을 통해 단어에서의 의미 요소를 빼고 더할 수 있다.
  - Ex) king - man + woman = queen
- 2D space로 변환하는 과정에서 잃어버리는 맥락이나 정보가 있을 수 있다.

### Optimization: Gradient Descent

- Minimizing cost function
- Gradient Descent is an algorithm to minimize cost function

#### Gradient Descent

- Update equation (in matrix notation):

$$\theta^{new} = \theta^{old} - \alpha \nabla_{\theta} J(\theta)$$

$\alpha$  = step size or learning rate

- Update equation (for a single parameter):

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial}{\partial \theta_j^{old}} J(\theta)$$

- alpha 값이 너무 크면 한 번에 많이 하강하게 되므로 정확하지 않은 결과가 나올 수 있다. 따라서 조금씩 하강할 수 있도록 적절한 값을 찾는 것이 중요하다.
- 기존의 cost function은 corpus 내의 모든 window에 대해 계산하기 때문에 계산량이 너무 많다.
  - 해결방법: Stochastic gradient descent (SGD)
  - Repeatedly sample windows, and update after each one

## Stochastic gradients with word vectors

- Iteratively take gradients at each such window
- But in each window, only have at most  $2m + 1$  words, so parameter update is sparse
- Therefore the goal is to update the word vectors that actually appear

## Word2Vec: More details

Two model variants

1. Skip-grams (SG)
    - : Predict context words given center words
  2. Continuous Bag of Words (CBOW)
    - : Predict center word from context words
- The skip-gram model with negative sampling
    - 기존의 방식에서 쓰인 normalization factor는 모든 단어에 대해 계산을 진행하므로 너무 느리다

$$\bullet P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

- 이를 해결하기 위해 negative sampling을 이용한다
- 분모에서 실제로 관찰된 단어에 대해 하나의 **binary logistic regression**을 진행하고, 실제로 관찰된 단어에 높은 확률을 부여한다.

- 그 후 다른 여러 개의 단어를 랜덤하게 샘플링하는데 이를 **negative sample** 이라고 한다.
- negative sample들은 실제로 관찰된 단어들이 아니므로 가능한 한 낮은 확률을 부여한다.

### The skip-gram model with negative sampling (HW2)

- Notation more similar to class and HW2:

$$J_{neg-sample}(\mathbf{o}, \mathbf{v}_c, \mathbf{U}) = -\log(\sigma(\mathbf{u}_o^\top \mathbf{v}_c)) - \sum_{k=1}^K \log(\sigma(-\mathbf{u}_k^\top \mathbf{v}_c))$$

- We take  $k$  negative samples (using word probabilities)
- Maximize probability that real outside word appears, minimize prob. that random words appear around center word
- $P(w) = U(w)^{3/4} / Z$ ,  
the unigram distribution  $U(w)$  raised to the 3/4 power  
(We provide this function in the starter code).
- The power makes less frequent words be sampled more often

### Co-occurrence matrix (vector)

- 사이즈가 너무 커서 비효율적.
- 해결 방법
  - Dimensionality reduction

### Count based vs. Direct prediction

- LSA, HAL (Lund & Burgess),
- COALS, Hellinger-PCA (Rohde et al, Lebrete & Collobert)

- Fast training
- Efficient usage of statistics
- Primarily used to capture word similarity
- Disproportionate importance given to large counts

- Skip-gram/CBOW (Mikolov et al)
- NNLM, HLBL, RNN (Bengio et al; Collobert & Weston; Huang et al; Mnih & Hinton)

- Scales with corpus size
- Inefficient usage of statistics
- Generate improved performance on other tasks
- Can capture complex patterns beyond word similarity

- 두 가지 방법의 장점을 결합한 방법인 GloVe도 있다.