< word2vec >

2013년 구글은 'word2vec' 이라는 강력한 도구를 발표한다. 'Skip-gram(스킵그램)'과 'Continuous bag of words(CBOW)'의 두 가지 모델을 포함한다. 이 모델들을 통해 특정한 단어를 계산 가능한 숫자로 바꿀 수 있고 서로 다른 두 단어의 유사성과 유사점의 정도를 잘 표현할 수 있다.

발표 후 word2vec은 자연언어처리에 널리 사용되고 있고, 그것의 원래 모델들과 훈련 방법들은 많은 단어 내장 모델들과 알고리즘들을 깨우친다.

1. Skip-gram(스킵그램)

: 한 단어에 집중하여, 주위에 어떤 단어가 나타날지 예측하는 데 사용한다.

ex) " The boy adores that girl. "

모든 두 단어 사이에 공백이 있기 때문에 "the", "boy", "adores", "that", "girl"과 같은 배경 단어들을 쉽게 알 수 있다. "adores"가 중심 단어이고, 양 옆의 단어의 개수를 2개로 똑같이 설정한다.

Skip-gram의 주요 견해는 바탕 단어가 중심 단어로부터 두 단어로 떨어져 있는 경우, 주어진 중심 단어를 토대로 각 바탕 단어의 조건부 확률이다.

- 수학적 언어

사전 색인 D 집합의 크기를 $|D|$라고 가정하고, $D=\{1,2,\dots,|D|\}$로 표시된다. T 길이의 문자 순서가 주어지고, t번째 단어를 $w^{(t)}$라고 나타낸다. 양 옆 단어의 개수가 m개로 동일할 때, Skip-gram은 임의의 중심 단어를 토대로, 중심 단어로부터 m개의 단어로 떨어져 있는 각 바탕 단어의 모든 조건부 확률의 총 합계를 최대화하려 한다.

$$\prod_{t=1}^{T} \prod_{-m \le j \le m, j \ne 0, 1 \le t+j \le |T|} P(w^{(t+j)} | w^{(t)})$$

그래서, 우도 함수는

$$\sum_{t=1}^{T} \sum_{-m \le j \le m, j \ne 0, 1 \le t+j \le |T|} \log P(w^{(t+j)} | w^{(t)})$$

위의 우도 함수를 최대화하여 다음 손실 함수를 최소화 할 수 있다.

$$-\frac{1}{T} \sum_{t=1}^{T} \sum_{-m \le j \le m, j \ne 0, 1 \le t+j \le |T|} \log P(w^{(t+j)} | w^{(t)})$$

중심 단어와 배경 단어의 벡터를 각각 **v**와 **u**로 표시한다. 즉, 색인 i를 가진 단어에 대해 v_i와 u_i는 각각이 중심 단어와 배경 단어일때의 벡터이다. 그리고 우리가 훈련시키고자 하는 모델의 매개변수는 모든 단어의 두 가지 종류의 벡터이다.

모델 매개변수를 손실 함수로 구현하기 위해, 주어진 중심 단어 하에서 배경 단어의 조건부

확률을 모델 매개변수와 함께 표현해야 한다. 중심 단어가 주어졌을 때, 각 배경 단어 생성이 상호독립적이라고 가정하자. 이 때 중심 단어 w_c와 배경 단어 w_b에 대해, b와 c는 사전에서 그것들의 색인이다. 그렇게 하면, 주어진 중심 단어 w_c하에서 배경 단어 w_b의 생성 가능성은 softmax 함수로 정의될 수 있다.

$$P(w_b|w_c) = \frac{\exp(u_b^T v_c)}{\sum_{i\in D}\exp(u_i^T v_c)}$$

도출하면, 위 조건부 확률의 변화도를 알 수 있다.

$$\frac{\partial \log P(w_b|w_c)}{\partial v_c} = u_b - \sum_{j\in D}\frac{\exp(u_i^T v_c)}{\sum_{i\in D}\exp(u_i^T v_c)}u_j$$

즉, 다시말해

$$\frac{\partial \log P(w_b|w_c)}{\partial v_c} = u_b - \sum_{j\in D}P(w_j|w_c)u_j$$

그리고 나서, 우리는 이것을 Gradient Descent나 Stochastic Gradient Descent을 반복하여 해결할 수 있다. 중심 단어와 배경 단어가 있고 손실 함수가 최소에 다다를 때 단어 벡터 v_i와 u_i, $i = 1,2,\ldots,|D|$일 때 모든 각 단어들을 알 수 있다.

단어 연속의 길이 T가 매우 길 때 대략적인 해결책을 알아내기 위해, 각 시대에서의 이러한 반복에 대한 손실을 계산하고자 다소 짧은 반복을 무작위로 추출할 수 있다.
자연어 처리 적용에서 각 단어의 단어 벡터로서 Skip-gram의 중심 단어를 사용할 것이다.

2. Continuous Bag of Words

: 문자순으로 중심 단어와 그 주위에 있는 배경 단어들을 예측한다.

ex) “ The boy adores that girl. ”
“the”, “boy”, “adores”, “that”, “girl”과 같은 배경 단어들을 알 수 있다. “adores”는 중심 단어이고, 양 옆의 단어의 개수를 2개로 똑같이 설정한다.

CBOW의 주요 견해는 바탕 단어가 중심 단어로부터 두 단어로 떨어져 있는 경우, 모든 바탕 단어들을 토대로 주어진 중심 단어를 생성할 조건부 확률이다.

- 수학적 언어
사전 색인 D 집합의 크기를 $|D|$라고 가정하고, $D = \{1,2,\ldots,|D|\}$로 표시된다. T 길이의 문자 순서가 주어지고, t번째 단어를 $w^{(t)}$라고 나타낸다. 양 옆 단어의 개수가 m개로 동일할 때 CBOW는 중심 단어로부터 m개의 단어로 떨어져 있는 모든 바탕 언어들을 토대로, 임의로 주어진 중심 단어 생성의 조건부 확률 총 합계를 최대화하려 한다.

$$\prod_{t=1}^{T} P(w^{(t)}|w^{(t-m)},...,w^{(t-1)},w^{(t+1)},...,w^{(t+m)}) \quad \cdots\cdots\cdots\cdots (7)$$

여기서 m은 중심 단어 양 옆 단어의 개수이고, $(t-m+j)\in[1,|T|],j\in[0,2m]$임을 보장해야 한다.

그래서, 우도 함수는

$$\sum_{t=1}^{T}\log P(w^{(t)}|w^{(t-m)},...,w^{(t-1)},w^{(t+1)},...,w^{(t+m)}) \quad \cdots\cdots\cdots\cdots (8)$$

위의 우도 함수를 최대화하여 다음 손실 함수를 최소화 할 수 있다.

$$-\sum_{t=1}^{T}\log P(w^{(t)}|w^{(t-m)},...,w^{(t-1)},w^{(t+1)},...,w^{(t+m)}) \quad \cdots\cdots\cdots\cdots (9)$$

우리는 여전히 Skip-gram 모델을 논의할 때 이 표기법을 사용한다. 중심 단어 v_c와 그것의 배경 단어들 $w_{b0},w_{b1},...,w_{b\cdot 2m}$은 softmax 함수에 의해 다음과 같이 정의될 수 있다.

$$P(w_c|w_{b0},w_{b1},...,w_{b\cdot 2m})=\frac{\exp(\frac{v_c^T(u_{b0}+u_{b1}+...+u_{b\cdot 2m})}{2m})}{\sum_{i\in D}\exp(\frac{v_i^T(u_{b0}+u_{b1}+...+u_{b\cdot 2m})}{2m})} \quad \cdots\cdots\cdots (10)$$

도출하면, 위 조건부 확률의 변화도를 알 수 있다.

$$\frac{\partial \log P(w_c|w_{b0},w_{b1},...,w_{b\cdot 2m})}{\partial v_{bi}} \quad \cdots\cdots\cdots (11)$$

$$=\frac{1}{2m}(v_c-\sum_{j\in D}\frac{\exp(\frac{v_c^T(u_{b0}+u_{b1}+...+u_{b\cdot 2m})}{2m})}{\sum_{i\in D}\exp(\frac{v_i^T(u_{b0}+u_{b1}+...+u_{b\cdot 2m})}{2m})}v_j)$$

즉, 다시말해

$$\frac{\partial \log P(w_c|w_{b0},w_{b1},...,w_{b\cdot 2m})}{\partial v_{bi}} \quad \cdots\cdots\cdots\cdots (12)$$

$$=\frac{1}{2m}(v_c-\sum_{j\in D}P(w_c|w_{b0},w_{b1},...,w_{b\cdot 2m})\cdot v_j)$$

이것을 Gradient Descent나 Stochastic Gradient Descent을 반복하여 해결할 수 있다. 중심 단어와 배경 단어가 있고, 손실 함수가 최소에 다다를 때 단어 벡터 v_i와 모든 각 단어

들의 $u_i(i = 1,2,\ldots,|D|)$를 알 수 있다.

단어 연속의 길이 T가 매우 길 때 대략적인 해결책을 알아내기 위해, 각 시대에서의 이러한 반복에 대한 손실을 계산하고자 다소 짧은 반복을 무작위로 추출할 수 있다.

자연어 처리 적용에서 각 단어의 단어 벡터로서 CBOW의 바탕 단어 벡터를 사용할 것이다.