Glove

1. Introduction

단어 벡터를 학습하기 위한 주요 모델은 1) LSA와 같은 global matrix factorization과 2) skip-gram과 같은 local context window methods가 있다. 이 두 모델은 큰 결점을 가지고 있다. LSA와 같은 방법은 통계적인 정보를 잘 보존하는 반면에 단어 유사도에서는 상대적으로 좋지 못한 성능을 보인다. skip-gram과 같은 방법은 유사도를 보고자 할 때에는 좋지만 분리된 global co-occurrence counts 대신 local context windows에 학습하기 때문에 코퍼스의 통계적 정보를 잘 활용하지 못한다. Glove는 두 모델의 단점을 모두 극복한다. 코퍼스의 통계적 정보를 잘 활용하면서, global word-word co-occurrence counts을 기반으로 하는 WLS 모델을 제시한다.

2. The GloVe Model

먼저 notation을 정의하자.

- X: 단어-단어 공기어(co-occurrence)를 나타내는 행렬이다. X_{ij}는 단어 j가 단어 i의 맥락에서 나오는 횟수이다.
- $X_i = \sum_k X_{ik}$: 단어 i의 맥락에서 나타나는 모든 단어의 횟수
- $P_{ij} = P(j \mid i) = X_{ij}/X_i$: 단어 j가 단어 i의 맥락에서 나타날 확률

공기어 확률에서 어떤 측면의 의미를 뽑을 수 있는지 예시를 통해서 살펴보겠다. 예를 들어, 열역학과 관련된 단어들에 관심이 있다고 할 때, $i = ice,\ j = steam$이라고 해보자. 이 두 단어의 관계는 다양한 단어 k에 대해서, ratio of co-occurrence probabilities을 살펴보면 알 수 있다. 아래 표를 보자.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

예를 들어, steam이 아니라 ice에 관련있는 $k = solid$ 단어에 대해서 P_{ik}/P_{jk}가 클 것으로 기대된다. 왜냐하면 ice와 관련되어 있는 solid에 대해, $P_{ik} = P_{ice,\,solid}$가 $P_{jk} = P_{steam,\,solid}$보다 클 것이기 때문이다. 위 표에서 그 값은 8.9로 그 값이 상대적으로 크다.
이와는 반대로, ice가 아니라 steam과 관련있는 $k = gas$ 단어에 대해서, P_{ik}/P_{jk}가 작을 것으로 기대된다. 왜냐하면 steam가 관련있는 gas에 대해서 $P_{jk} = P_{steam,\,gas}$가 $P_{ik} = P_{ice,\,gas}$보다 클 것이기 때문이다. 위 표에서 그 값은 8.5×10^{-2}로 그 값이 상대적으로 작다.
만약 비율이 아닌, 확률 자체만 본다면 어떨까? $P_{ice,\,solid} = 1.9 \times 10^{-4}$는 solid와 ice가 연관되어 있는지 쉽에 와닿지 않는다. 하지만 $P_{ice,\,solid}/P_{steam,\,solid}$가 크다는 것은 분자가 크다는 것이므로, solid에 대해 steam보다는 ice가 더 연관되어 있을 것이라고 유추할 수 있다.
또 다른 경우로, ice나 steam과 모두 관련되어 있는 water에 대해서는 분자, 분모가 똑같이 클 것이므로 1에 가까울 것이다. 유사하게 ice, steam과 모두 관련 없는 fasion에 대해서는 분자, 분모가 똑같이 작을 것이므로 비율이 1에 가까울 것이다. 즉, 위의 예시와는 다르게 water나 fasion이 ice, steam과 연관된 관계의 방향성이 동일하다는 것이다.
결론적으로 두 단어를 살펴보는raw probabilities보다, 세 단어의 관계를 나타내는 비율이 관련된 단어(solid, gas)와 관련되지 않은 단어(water, fasion)을 구분하는데 더 낫고, 관련된 단어 중에서도, 작은지 큰지에 따라서 어떻게 관련 되어 있는지 잘 판단할 수 있다.
비율 P_{ik}/P_{jk}가 i, j, k에 의존한다는 것을 이용하여 일반적인 형태를 아래와 같이 쓸 수 있다.

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}} \tag{1}$$

여기서 $w \in \mathbb{R}^d$는 단어 벡터이고 $\tilde{w} \in \mathbb{R}^d$는 별개의 context 단어 벡터이다. (1)에서 우변은 코퍼스로부터 얻을 수 있고 F가 명시되어 있지 않은 모수이다. F의 후보는 많이 있겠지만, 조건을 추가하며 범위를 좁혀보자.
우리는 P_{ik}/P_{jk}에 들어있는 정보를 word vector space에 인코딩 하는 F를 원한다. 그런데 vector space는 linear 구조이므로, 이를 표현하는 가장 자연스러운 방법은 벡터 뺄셈을 하는 것이다. 이러한 목적을 가지고, 함수 F를 아래와 같이 제한할 수 있다.

$$F(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}} \tag{2}$$

(2)에서 F의 argument는 벡터이고 우변은 scalar임에 주목하자. 물론 input으로 벡터를 받는 neural network도 있지만, 여기서는 간단한 linear 구조인 vector space를 생각하므로, 이러한 복잡한 구조는

피하는 것이 좋다. 따라서 벡터의 내적을 arguments로 취한다.

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}} \tag{3}$$

마지막으로, 단어-단어 공기어 행렬에서 단어와 context 단어는 임의로 정할 수 있으므로 이 둘의 역할을 바꿔도 무방하다. 이러한 일치성을 위해서 w와 $\tilde{w}$뿐만 아니라 X와 X^T를 서로 바꿀 수 있어야 하므로 모델은 이러한 relabeling에 invariant해야 한다. 대칭성은 두 가지 step에 의해서 만족될 수 있다.

- $Homomorphism:\ F(X-Y) = \frac{F(X)}{F(Y)}$ 조건인데, 이를 (3)에 적용해보면 $F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)}$으로 바꿀 수 있다. 여기서 $F(w_i^T \tilde{w}_k) = P_{ik} = \frac{X_{ik}}{X_i}$이다.
- 위 조건을 만족하는 함수 F는 exp 함수이다. 즉,

$$w_i^T \tilde{w}_k = log\left(P_{ik}\right) = log\left(X_{ik}\right) - log(X_i) \tag{4}$$

- (4)의 관계식은 $log(X_i)$만 아니라면, exchange 특징을 가진다. 하지만이 항은 k와 관련이 없으므로 w_i에 대한 bias인 b_i에 합칠 수 있다. 마지막으로 $\tilde{w}_k$에 대한 추가적인 bias인 $\tilde{b}_k$를 더한다면, 대칭을 구현할 수 있다.

$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = log\left(X_{ik}\right) \tag{5}$$

  numerical stability을 위해서 $log\left(X_{ik}\right)$가 아니라 $log\left(X_{ik}+1\right)$로 두는 경우가 많다.

(5)에서 미지수는 좌변이고 우변의 $log\left(X_{ik}\right)$는 이미 알고 있는 값이다. 따라서 우변과의 차이를 최소로 하는(Least Squares 개념) 좌변의 값이 d차원 벡터 공간에 적절히 임베딩된 단어 벡터들일 것이다.
이 모델에 주요한 단점은 가끔 일어나거나 아예 일어나지 않은 것에 대해서도 동일한 가중치를 준다는 것이다. 따라서 이러한 문제를 해결하기 위해 가중치를 달리하는 weighted least squares regression을 제안하여 아래와 같은 손실 함수를 정의한다.

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - log\left(X_{ij}\right)\right)^2 \tag{6}$$

여기서 V는 단어의 크기이다. weighting 함수는 아래의 특징을 만족해야 한다.

1. $f(0) = 0$
2. $f(x)$는 non-decreasing이어야 한다. 만약 감소함수라면, 적게 나타나는 공기어에 대해 over-weight을 주기 때문이다.
3. $f(x)$는 큰 x 값에 대해서는 비교적 작은 값을 주어, 너무 많이 동시에 나타나는 공기어는 over-weight되지 않게 한다. 이유는 은,는 등의 공기에는 가중치를 높게 주는 것은 올바르지 못하기 때문이다.

이러한 조건을 만족하는 많은 함수들 중에서 논문의 저자가 잘 작동하는 함수를 아래와 같이 발견하였다.

$$f(x) = \begin{cases} (x/x_{max})^{\alpha} \, if \; x < x_{max} & if \; x < x_{max} \\ 1 & o.w. \end{cases} \tag{7}$$

GloVe는 임베딩 벡터를 만들기 위해서 공기어 행렬 X를 만드는 것이 학습의 시작이다. 단어 개수가 1만개 정도 되는 코퍼스라면 10000×10000 차원의 행렬을 만들어야 한다는 뜻이다. 이후 (6)의 objective function을 최소화 하기 위해서, matrix factorization을 수행하는 것이다. 따라서 GloVe는 계산 복잡성이 꽤나 큼을 유추할 수 있다.

Reference

1. GloVe: Global Vectors forWord Representation, By Jeffrey Pennington, Richard Socher, Christopher D. Manning

2. https://ratsgo.github.io/from%20frequency%20to%20semantics/2017/04/09/glove/