MinHash

Sungjoo Ha

August 11th, 2017

Jaccard Distance

$$
\begin{aligned}
J(A,B) &= \frac{|A \cap B|}{|A \cup B|} \\
d_J(A,B) &= 1 - J(A,B)
\end{aligned}
$$

- 두 집합 사이의 거리를 측정하는 한 가지 방법
- 집합이 크고 비교해야 하는 집합이 많으면 수행시간이 오래 걸릴 수 있음

MinHash

- Hashing을 랜덤 샘플링 기법으로 활용하여 Jaccard distance 근사에 활용
- 해시함수 h를 A의 모든 원소에 적용 후 해시값이 가장 작은 원소 $h_m(A)$를 추출
- 동일 과정을 집합 B에 적용해서 해시값이 가장 작은 원소$h_m(B)$를 추출
- $P[h_m(A) = h_m(B)] = J(A, B)$

Many Hash Functions

- $h_m(A) = h_m(B)$를 체크하는 것은 unbiased estimator 이지만 0/1의 두 값만 나오므로 분산이 큼
- 해시함수를 k 개 사용해서 $h_m^k(A) = h_m^k(B)$ 를 만족하는 개수를 k 로 나누면 $J(A, B)$에 대한 좋은 추정이 됨

Single Hash Function

- 여러 해시함수를 사용하는 것은 비쌀 수 있음
- $h_k(A)$를 집합 A 중 해시값이 가장 작은 k개의 원소라 하면
- $X = h_k(h_k(A) \cup h_k(B)) = h_k(A \cup B)$
- 즉, X는 $A \cup B$에서의 랜덤 샘플이라 할 수 있음
- $Y = X \cap h_k(A) \cap h_k(B)$는 X에 속하는 원소 중 $A \cap B$에도 속하는 원소가 됨
- 그러므로 $\frac{|Y|}{k}$ 가 $J(A, B)$의 unbiased estimator가 됨

Recap

- 해시값이 가장 작은 원소를 선택하는 것은 랜덤 샘플링의 역할
- 해시값이 변하지 않는다는 사실을 활용해 집합의 signature를 만들 수 있음
- 앞선 논의에 따라 signature만 가지고도 두 집합 사이의 거리 측정 가능
- 집합마다 미리 signature를 만들어두고 이를 활용하여 두 집합의 Jaccard distance를 빠르게 계산

Application

- 문서 사이의 유사도 측정 (locality sensitive hashing)
- 두 벡터 사이의 거리 계산 가속