인공지능 과제 #2

**0. 개발환경**

- JAVA Eclipse Oxygen.3a Release(4.7.3 a)

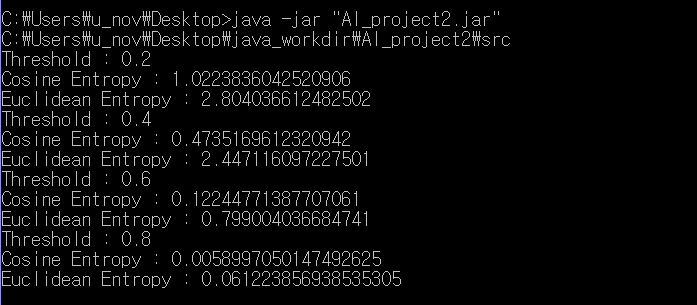
- JDK 1.8

- Command 실행 후 아래와 같이 입력

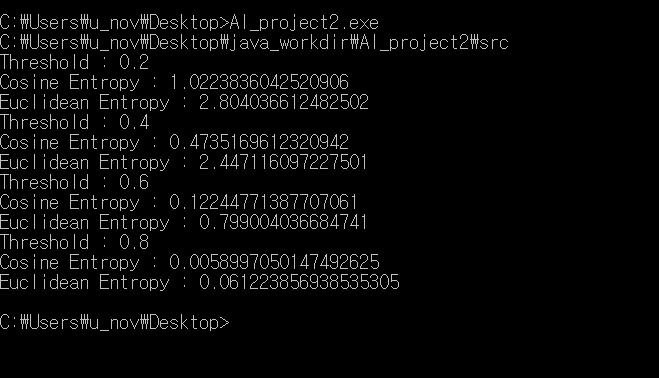
1) java -jar “AI\_project2.jar” (해당 jar 경로)

2) 다음으로 WordEmbedding.txt 와 WordTopic.txt 가 있는 path를 입력

(두 파일은 같은 경로에 위치)



- 혹은 .exe 파일을 실행 후 path 입력(JSmooth-0.9.9-7 사용)



- clustering number 첨부한 파일 역시 path에 생성

**1. 클러스터링 결과**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Consine Similarity | | Euclidean Similarity | |
|  | Entropy | Cluster 개수 | Entropy | Cluster 개수 |
| Threshold = 0.2 | 1.022383 | 82 | 2.804036 | 7 |
| Threshold = 0.4 | 0.473516 | 184 | 2.447116 | 39 |
| Threshold = 0.6 | 0.122447 | 286 | 0.799004 | 217 |
| Threshold = 0.8 | 0.005899 | 331 | 0.061223 | 327 |

Euclidean distance는 Cosine similarity와는 다르게 [0:2\*sqrt(300)] 의 범위를 가지므로 이를 다음의 min – max scaling을 통하여 scaling을 통해 Euclidean similarity를 구하였다.

**2. 클러스터링 평가**

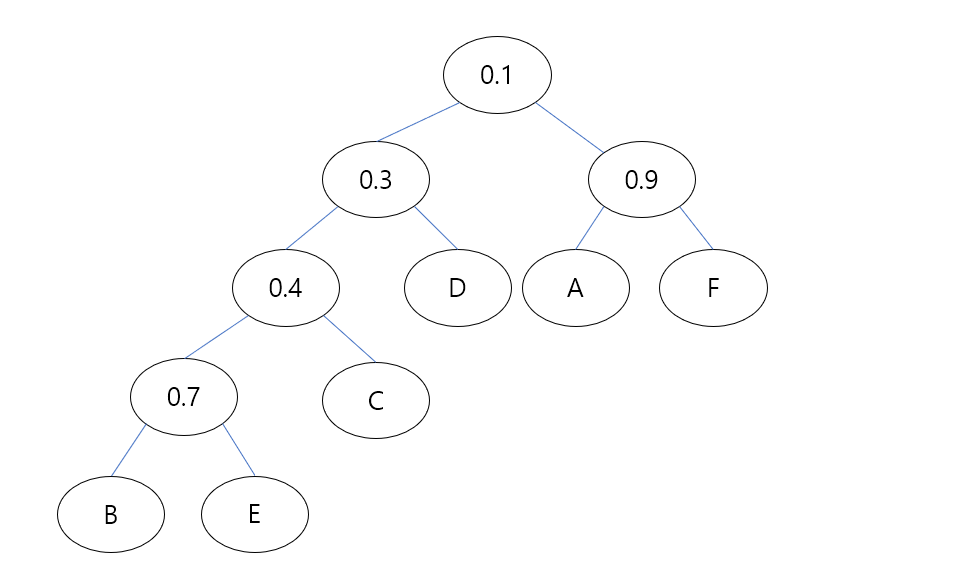
**Threshold에 따른 cluster 수와 entropy**

Threshold가 작을 때는 한 cluster에 많은 word가 clustering 되었으며, threshold 클수록 더 많은 cluster set을 생성하였으나 각 cluster에 더 적은 개수의 word가 포함이 되었다. 실제로 clustering 결과, cosine similarity 기준으로 threshold가 0.2 일 때는 82개, 0.4 일 때는 184개, 0.6일 때는 286개, 0.8일 때는 331개의 cluster가 생성되었으며, 단어가 총 338개임을 생각하면, 0.8의 경우는 거의 대부분의 cluster가 1개의 단어를 가지고 있음을 알 수 있다. Cluster에 한 개의 word만 있으면 자연히 하나의 class로 구성되어 있는 cluster 이므로, purity가 가장 좋을 것이고 Entropy가 0이 된다. 따라서, threshold가 커질수록 entropy는 0에 가까워 진다. 따라서, 적절한 크기의 clustering 을 하기 위해서는 threshold가 어느정도는 작아야 하며, purity를 위해서는 어느정도는 커야 한다.

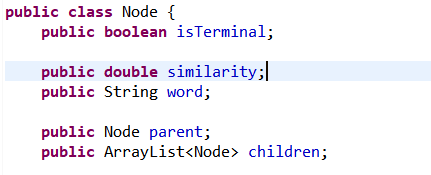
**Cosine Similarity와 Euclidean Similarity의 비교**

Cosine Similarity는 cluster 개수에서 확인할 수 있다시피 그 값이 고르게 분포되어 있으나, Euclidean similarity는 0.6과 0.8사이에 대부분이 집중되어 있음을 볼 수 있다. Threshold에 의해서 clustering이 고르게 나누기 위해서는 균등하게 분포 되어 있는 Cosine similarity가 clustering에 더 이득일 것이다.

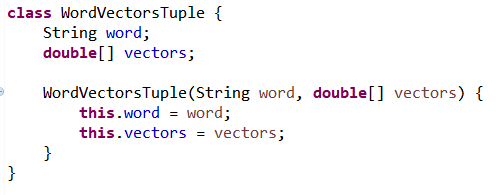
**3. 코드 분석**



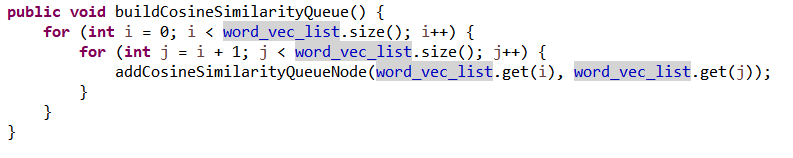
Complete Link Structure를 위와 같은 구조로 구성하였다. 각 node가 terminal일 때 word node를 나타내고 그 외에는 similarity node를 나타낸다.



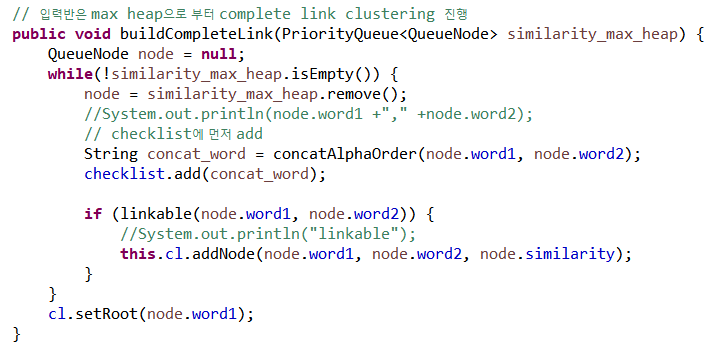
파일로 입력받은 word와 300차원 vector는 WordVectorTuple에 담아두었다.



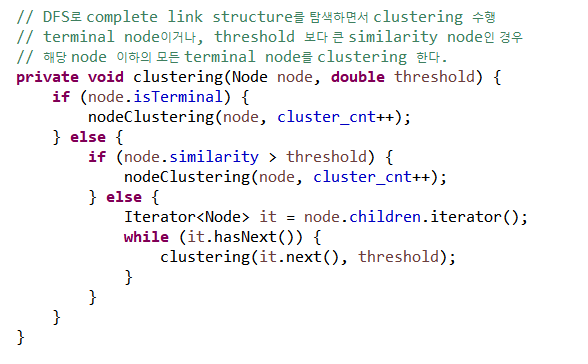
Word\_vec\_list는 WordVectorsTuple 타입을 담고있는 arraylist이며, 300개의 단어를 아래와 같이 조합하여 각 pair 쌍의 similarity를 구해서 similarity priority queue에 담았다.



구성된 similarity max heap을 통해 complete link structure를 구성한다. Checklist에 해당 단어 pair 쌍을 넣고, linkable 한지 확인한다. Linkable은 checklist를 확인하여 필요한 모든 pair 쌍이 있는지를 확인한다. 만약 linkable 하다면, complete link에 node를 추가한다.



주어진 threshold를 가지고 clustering을 진행한다. Complete link structure를 탐색하면서, terminal이거나, threshold 보다 큰 similarity node 를 만날 경우, 해당 node 이하의 모든 node들을 clustering 한다.



Key로 단어와 value로 topic으로 구성된 hashmap 을 이용하여 cluster의 entropy를 계산한다. 모든 cluster를 iterate 하면서 각각 cluster의 entropy를 구하고, weighted sum을 구한다.

