Arnaud Massenet
arnaudyoh@gmail.com

Lab session # 1
ALTEGRAD 2019

02/12/19

Question 1

We computed the density of a the graph of word with window sizes ranging from 0 to 20. Here are the results below:

Window size	Density	Window size	Density
2	0.106	12	0.742
3	0.212	13	0.758
4	0.318	14	0.773
5	0.417	15	0.788
6	0.523	16	0.795
7	0.583	17	0.803
8	0.629	18	0.803
9	0.667	19	0.803
10	0.697	20	0.803
11	0.720		

Table 1: Density of Graph relative to window size

We observe from the above table that the density increases with the window size. Note that it looks like the increase is logarithmic as the higher the window size the slower the density increases.

Question 2

Don't assume worst case but rather average case.
In Algorithm 1 we consider $n = |V|$ as the number of nodes and k the average number of neighbours.
We have a while loop, that loops $O(|V|)$ times with a few operations inside of it:

- At line 3 we get the node with the lowest value out of p. This requires time complexity $O(|V|)$.
- Line 4 is not very relevant complexity-wise (it is $O(1)$).
- Line 5 has a complexity of order $O(|V|)$. We are looking through all entries of the column of the node in the adjacency matrix.
- Line 6 and line 7 are the deletion of a node and every edge that contains that node. We will have to look in all existing edges, as such our complexity is $O(|V| + |E|)$.
- Line 8 and 9 is a for-loop that performs a constant time operation k times where k is the number of a neighbours of the current node. Thus, the time complexity of the for loop is $O(|V|)$.

As a result the worst case complexity of our algorithm is $O(|V|^2 + |V||E|)$.

Question 3

We have tested four methods: TF-IDF, K-Core (window wize 4), Weighted-K-Core (window wize 4) and PageRank. We show the results in the table 2:

Method	Precision	Recall	F1-Score
TF-IDF	59.21	38.5	44.85
PageRank	60.18	38.3	44.96
K-Cores	51.86	62.56	51.55
Weighted-K-Cores	63.86	48.64	46.52

Table 2: Scores out of 100. The higher the better. For both K-Cores method we used a window size of 4

Overall, we can quickly observe that the TF-IDF method has a very similar performance to PageRank on precision, recall and F1-score. Weighted-K-Cores outperforms them on the recall and performs slightly better on the two other scores. The K-Cores method has the worst precision but significantly better recall score than PageRank and TF-IDF and the best F1-Score.
Overall, K-Cores is best at minimizing the amount of false negatives while Weighted-K-Cores is better at minimizing false positives. Note however that K-Cores has the best F1-score and thus, we could argue that it has the ”best” balance between precision and recall.

Question 4

As mentionned above and as supported by Table 2, the weighted method has an overall better ability to avoid false positive but it is worse to avoid false negative compared to the vanilla K-Cores method. In a practical settings, we will better use the vanilla K-Cores if we want to avoid false negatives (eg. we want to avoid people thinking they are disease-free when they are not) while we would prefer the weighted version when we want to avoid false positives (eg. we have limited supply of medication and we want to give them only to the patients that need them).

Question 5

We can improve the performance of the K-Cores methods by playing with the window size. In the table 2 above we reported the results for a window size of 4. We noticed that the F1-score increases a bit when we increase the window size to 10. Other than playing with the hyperparameters, it would be also interesting to see if we could modify the definition of the weighting for the weighted-K-Cores to see if the algorithm’s performance could be further improved.