1 Question 1

While increasing the window size from 2 to 9, the density of the graph increases because the number of edges between nodes increase. with a reference to Fig. 1

```
The density of the graph is for w=  2 is  0.10606060606060606
The density of the graph is for w=  3 is  0.21212121212121213
The density of the graph is for w=  4 is  0.3181818181818182
The density of the graph is for w=  5 is  0.41666666666666663
The density of the graph is for w=  6 is  0.5227272727272727
The density of the graph is for w=  7 is  0.5833333333333334
The density of the graph is for w=  8 is  0.6287878787878788
The density of the graph is for w=  9 is  0.6666666666666666
```

Figure 1: This is a caption.

2 Question 2

The time complexity of k core decomposition:
$n = |V|$
m$= |E|$
neigh$_m$ = mean number of neighbors of the nodes.

Line 1 : Calculate the degrees of each node $O(max(m, n))$ which is less then n^2
Line2: while $O(n)$: Loop over all the vertices
Line 3: $O(n^2)$: Get the vertex with the lowest weight, but since the number of vertices will decrease by 1 at each iteration so the number of operation considering the loop will be

$$\sum_{i=1}^{n} i = O(n^2)$$

line 6 and 7 : $O({neigh_m}^2)$: To delete the edges of a node from an adjacency list we need to loop over all the edges. But the number of edges will decrease at each time but we can consider an upper bound m but in the average case scenario it is the mean of the number of neighbors of all the nodes squared.

So the total complexity is

$$O(n^2 + n * {neigh_m}^2)$$

Or an upper bounded complexity of

$$O(n^2 + n * m)$$

3 Question 3

The performance of k-core and weighted k-core algorithm compared to PageRank and TFIDF with getting top 0.33 of the ranked words i. Overall, PageRank and TFIDF have similar results, with a precision higher than the recall. It is the opposite for k-core, which tends to extract a main core with a lot of vertices since the k-core condition can be interpreted as a set of keywords that co-occur with at least k other keywords, so the recall would be higher as shown by Fig 2 .

4 Question 4

The main advantages of k-core is that is able to capture term dependence and term order via directed graph edges.

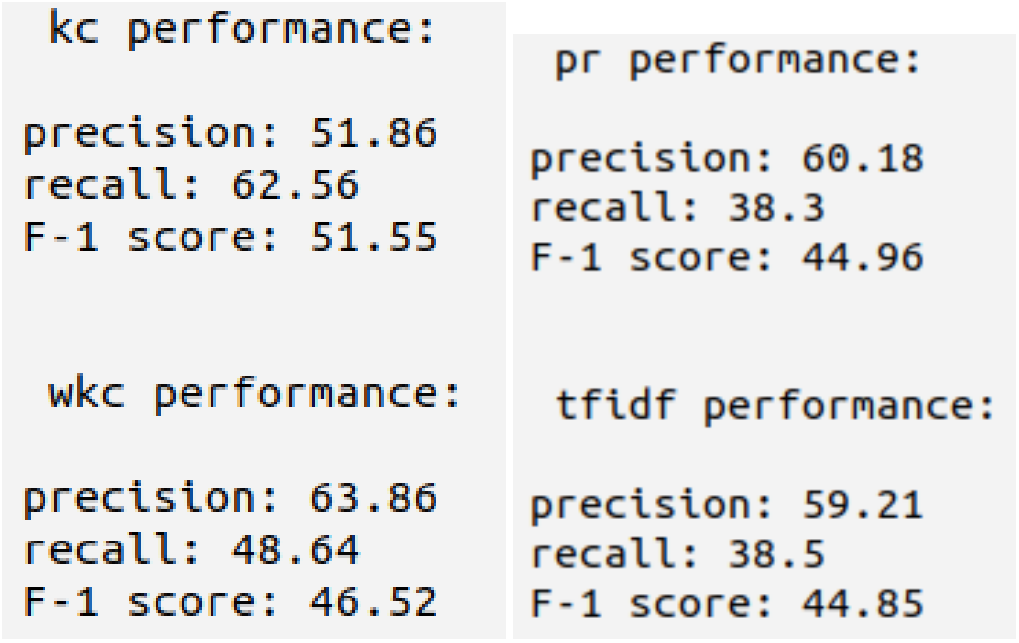

Figure 2: This is a caption.

Another advantage of weighted k-core is that it encodes the strength of the dependence as edge weights.

On the other hand K-core decomposition suffers from the following limitations: (1) k-core is good but not best in capturing cohesiveness; (2) retaining only the main core (or truss) is suboptimal, as one cannot expect all the gold standard keywords to be found within a unique subgraph actually, many valuable keywords live in lower levels of the hierarchy. [2]

Also one of the drawbacks of the k-core decomposition is that the nested k-core subgraphs do not satisfy a natural density property, simply defined as the ratio between the number of edges and nodes of the subgraph. In other words, the maximal k-core subgraph is not necessarily the densest subgraph of the graph. Since it is a greedy algorithm, we may have a node (word) with a high degree which is connected to low degree nodes. So when we remove iteratively low degree nodes, the high degree one will loose its importance[1]

5 Question 5

To improve k-core decomposition we can consider words from the other k-1 cores if they satisfy a threshold of density of the subgraphs. Or we can add backward edges to capture the dependence of words in bidirectional way.

References

[1] Apostolos Papadopoulos Michalis Vazirgiannis Fragkiskos Malliaros, Christos Giatsidis. The core decomposition of networks: Theory,algorithms and applications. 2019.

[2] Rousseau and Vazirgiannis. Main core retention on graph-of-words for single-document keyword extraction. 2015.