
Protein Localization

Protein Localization Prediction Using Protein Interaction Networks

Matthijs Klaassen^{1,*}

Abstract

In this paper we investigate 4 different methods to predict protein sub cellular location using protein interaction networks. We present a very simple algorithm as a benchmark to which the other, more complex, methods can be measured. This algorithm is based on the simple observation that to interact proteins must be physically close to one another, so there is some relation between interactions and locations. We use node2vec to find feature vector representations of the networks and see if information about protein locations is recapitulated. SimRank is used to obtain a similarity measure between proteins in the network and the similarity is used as weights in an extended form of the first algorithm. Finally we use the matrix exponential to get a general measure of distance between proteins and locations. We find that, compared to the first simple algorithm, more complex methods fail to make adequate predictions. Analyzing the algorithms reveals that this might be a result of the fact that the networks are strongly connected. As a result the simple localized algorithm performs much better than the other algorithms that take the complete graph into account.

Contact: matthijsklaassen1995@gmail.com

1 Introduction

Proteins are the molecular entities responsible for cell functioning. Proteins mostly act within signalling pathways to send messages from membrane to nucleus, or to exert certain actions on other proteins (activation, phosphorylation, etc.). The compartment of the cell they locate to is critical to their action and knowledge about their location can aid in the identification of drug targets Hung and Link[7]. Therefore it is worthwhile to study methods that give insight in protein sub cellular locations.

The traditional way of experimentally determining a protein's sub cellular locations requires a lot of human labour. For this reason, automated computer algorithms that could accurately detect protein sub cellular locations would be useful. For instance, Xiaohua Xu[13] and Emanuelsson[3] showed it is possible to classify proteins (find their sub cellular location) by using a random walk algorithm on a network built by looking at amino acid sequences of proteins. Others have tried to automate the localization of proteins by using neural networks on microscopy images of cells in which proteins have been marked using immunofluorescence [12][9].

Another possibility for automating the process is to use protein interaction networks (PIN). There are free online databases that contain extensive PIN's [1]. In these networks proteins are represented as nodes and edges between them represent an interaction between two proteins. The idea behind using these PIN's for localization prediction follows from a simple hypothesis: If a protein has an interaction with another protein then they must be located in the same location of the cell at least at the moment of interaction. Thus it follows that a proteins location can be predicted, up to a certain level, using information about it's interactions.

The general problem can be described as follows: 'Given a set of proteins, their sub cellular location, and a set of interactions between those proteins, can we predict the sub cellular location of any given protein.'

We investigate four methods for prediction:

- f-neighbour
- Node2Vec
- weighted-f-neighbour (SimRank)
- k-walks

The first method is intentionally simple, its main purpose is to provide a benchmark to which the other, more complex, methods can be tested. The second method, node2vec, is not a method to predict protein location in itself, but having a feature vector representation of a graph would enable us to apply a variety of existing prediction algorithms, in this paper we only investigate if a useful feature vector representation for the purpose of protein location prediction can be obtained. Node2vec was created by Grover and Leskovec[6] and used on the very same PIN used in this paper (Homo Sapiens) its goal, however, was to predict protein interaction and not protein location. The third method tries to improve on the first method by making use of a similarity measure, called SimRank, between nodes, which in principal should make predictions more accurate. Finally, k-walks gives us a measure of distance between a location and a protein. We show how that measure might be used to predict protein localization.

The aim of this paper is to investigate the potential of advanced methods (node2vec, SimRank, and k-walks) for the purpose of protein location prediction.

2 Materials & Methods

Data

We use data from the BIOGRID database [1]. BIOGRID offers data for different organisms, we focus solely on the Homo Sapiens and Saccharomyces Cerevisiae networks because these were the largest both in number of proteins and number of interactions. The network contains both physical (raw) interactions and genetic interactions we only consider physical interactions in this paper. The UniProt [2] database was cross referenced to find the location of most proteins in those two sets. Proteins of which no location is known were taken out as well as locations that host less than 500 proteins. Table 1 gives an overview of the data sets and the

Table 1. The data sets used and the number of interactions and proteins present in both.

Dataset	Interactions	Proteins
Homo Sapiens	381 353	18 249
Saccharomyces Cerevisiae	599 271	5 512

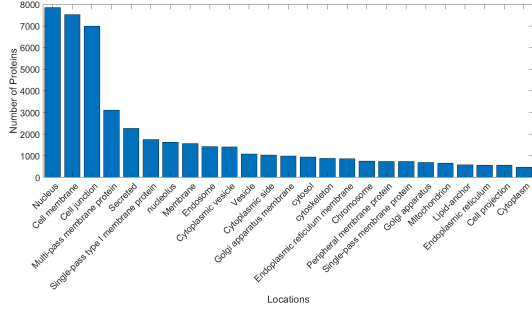


Fig. 1. Number of proteins by locations for the Homo Sapiens data set. Locations with less than 500 proteins filtered out. The distribution is unbalanced, the top 3 locations each contain almost half of all proteins

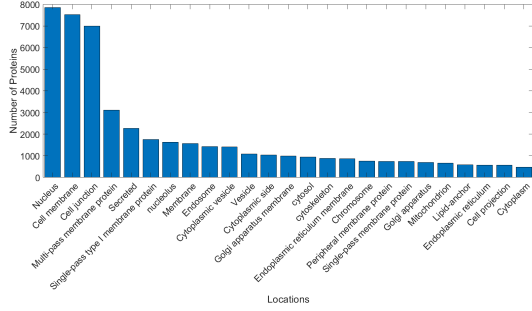


Fig. 2. Number of proteins by locations for the Saccharomyces data set. Locations with less than 500 proteins filtered out. The distribution is unbalanced, the top 3 locations each contain almost half of all proteins.

number of interactions and proteins present in both after removing genetic interactions and proteins with an unknown location.

There are more than 500 locations in the UniProt database, but only 25 with more than 500 proteins present. Figures 1 and 2 show these locations and the number of proteins localized to those locations, note that a protein might be located in 1 or more locations and in such cases it is counted once per location. When predicting protein sub cellular location researchers often consider less than 25 locations, see for instance Kraus et al.[8], Xiaohua Xu[13], and Pärnamaa and Parts[11].

Evaluation

To evaluate the performance of the methods we use the F1 metric. Precision and recall are defined as follows:

$$\text{precision} = \frac{|\{\text{actual proteins}\} \cap \{\text{predicted proteins}\}|}{|\{\text{predicted proteins}\}|} \quad (1)$$

$$\text{recall} = \frac{|\{\text{actual proteins}\} \cap \{\text{predicted proteins}\}|}{|\{\text{actual proteins}\}|} \quad (2)$$

$$F1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} \quad (3)$$

Where the set *actual proteins* refers to the set of proteins known to localize to a specific location and the set *predicted proteins* refers to the set of predicted proteins. Precision is the proportion of predicted proteins that are predicted correctly, it is comparable to accuracy. Recall is the proportion of actual proteins that have been predicted. If precision and recall would be applied to the art of clay pigeon shooting, precision would signify the proportion of shots taken that have hit their mark and recall would signify the proportion of clay pigeons thrown that have been hit. The important distinction being that if you only take one shot and hit a clay pigeon your precision is 1, but if in total 10 clay pigeons were thrown your recall is 0.1. F1 combines the two measures into a single metric that reflects the overall performance, in the clay pigeon example the f1 would be 0.182. The performance is evaluated on a per location basis so that the difference between common and less common locations can be observed. Imagine that, in the clay pigeon example, 99 blue pigeons and 1 red pigeon are thrown. A shooter, making no distinction in clay pigeon color, might show good performance (f1 score) for the blue pigeons (their sheer number making them hard to miss) but have a very low f1 (0) for the red pigeon. In this case the f1 score over both colors would still be high but would not reflect the low performance for the red pigeon.

Methods

f-neighbour

We restate the hypothesis from the introduction: If protein A interacts with protein B, then they must be located in the same part of the cell, at least at some specific moment when these two proteins interact. This assumption forms the basis of the first method: f-neighbour. Algorithm 1 describes the function of the method on some graph $G < V, E >$ where the vectors represent proteins and the edges represent interactions between them. There is also a Map M that contains, for every protein, the set of known locations.

We are aware it is technically only necessary to iterate over a protein's neighbours once, instead of one iteration for every location, but this will only yield a marginal performance gain and the run-time of the algorithm is very short already.

Algorithm 1: f-neighbour

```

1 Let  $G < V, E >$  be a graph;
2 Let  $P$  be a subset of  $V$  we want to predict;
3 Let  $M$  be a list of locations;
4 Let  $f$  be a real number between 0 and 1;
5 for  $p \in P$  do
6   for  $l \in M$  do
7      $i \leftarrow 0$ ;
8     for  $n \in p.\text{neighbours}$  do
9       if  $n.\text{presentAt}(l)$  then
10         $i \leftarrow i + 1$ ;
11      end
12    end
13    if  $i/p.\text{neighbours.size} \geq f$  then
14      assign label  $l$  to  $p$ ;
15    end
16  end
17 end
```

node2vec

In node2vec we take a graph as input and returns for every node in the graph a feature vector representation. The aim of the algorithm is to compute a vector representation that faithfully recapitulates the graph topology. The creators of node2vec Grover and Leskovec[6] suggest that the effectiveness of the algorithm plateaus around a feature vector size of 100, so we used that as a crude indication of what feature vector dimensions to use. We generated feature vectors of dimensions 16, 32, 64 and 128. All other parameters were kept at their default values. To get a measure of how well node2vec is learning the topology of the graph, specifically for the purpose of protein location prediction, we analyze the resulting feature vectors in two ways:

- Jaccard Similarity vs Euclidean Distance: we plot the jaccard similarity between the locations sets of two proteins against the euclidean distance of the feature vectors. An inverse relationship (i.e. higher similarity as distance gets smaller) would indicate that information about protein locations is captured in the feature vectors.
- In- vs out-group euclidean distance: we make a box plot of the distribution of euclidean distances between proteins that localize to some particular location (in group) and a box plot of the distribution of euclidean distances between those proteins and proteins not in that specific location (out group). If the distribution of euclidean distances in the in group is smaller than the distribution between euclidean distances of the out group then that would indicate some information about protein locations is captured in the feature vectors.

weighted-f-neighbour

In an attempt to increase the performance of the f-neighbour algorithm we use SimRank [5]. SimRank is an algorithm to measure the similarity between 2 nodes, a and b, and is defined as:

$$s(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b)) \quad (4)$$

Where C is a constant between 0 and 1, and I(a) and I(b) refer to the set of in-neighbours of nodes a and b respectively. Equation 4 is recursive but can be approximated by iteratively calculating the similarity between every pair using the similarity matrix from the previous iteration (the initial similarity matrix is a matrix with 1 along the diagonal and all other entries 0). The creators of SimRank suggest a decay factor of 0.8 and 5 iterations, and that is what is used here.

We use the similarity to make a small adaption to algorithm 1. Rather than simply counting the number of neighbours located in a particular location, we sum the weights of those neighbours and then divide by the total weight of all neighbours. Algorithm 2 shows the algorithm we end up with. The idea here is that more similar neighbours contribute more to the prediction then less similar neighbours and this should, in theory, increase performance.

k-walks

A question we can ask ourselves about graphs is: how 'far' are any 2 nodes from one another? Obviously two neighbouring nodes are closer to one another than two nodes that do not share an edge. One way to formalize this is to make a binary adjacency matrix of size n-by-n, where n is the number of nodes in the graph. Any entry [i,j] of this matrix will contain a 1 if there is an edge from node i to node j or a 0 otherwise, let's call this matrix A. Of course A only contains information about direct neighbours and no general measure of distance between any two nodes. Now let us take matrix A and raise it to power 2 to get A². The entry [i,j] in A² now contains the number of paths of length 2 going from node i to node j. This concept can be generalized by using the matrix exponential. Estrada

Algorithm 2: weighted-f-neighbour

```

1 Let  $G < V, E >$  be a graph;
2 Let  $P$  be a subset of  $V$  we want to predict;
3 Let  $M$  be a list of locations;
4 Let  $f$  be a real number between 0 and 1;
5 for  $p \in P$  do
6   for  $l \in M$  do
7      $i \leftarrow 0$ ;
8      $s \leftarrow 0$ ;
9     for  $n \in p.neighbours$  do
10      if  $n.presentAt(l)$  then
11         $i \leftarrow i + sim(p, n)$ ;
12      end
13       $s \leftarrow s + sim(p, n)$ ;
14    end
15    if  $i/s \geq f$  then
16      assign label  $l$  to  $p$ ;
17    end
18  end
19 end

```

et al. [4] use the term *communicability* to refer to the value $(exp(A))_{ij}$, this value reflects the general connectivity between nodes i and j and is formally defined as:

$$exp(A) = \sum_{k=0}^{\infty} \frac{1}{k!} A^k \quad (5)$$

Nodes with a lot of paths between them (shorter paths weigh more heavily) will have a high communicability.

There are multiple ways to find the exponential of a matrix:

1. Using the eigendecomposition of A: $exp(A) = Ue^DU^{-1}$. Where D is the diagonal matrix of A whose diagonal entries are the eigenvalues of A and U is formed from the corresponding eigenvalues of A.
2. Since the series converges we can approximate it by computing the first k values of the series, provided k is sufficiently large.
3. Using the Padé approximant, Padé[10].

The concept of communicability can be applied to our problem of protein location prediction in the following way:

- Add locations as nodes to the PIN.
- Add edges between locations and proteins known to localize to those locations.
- Calculate the matrix exponential of the adjacency matrix of this graph.
- Use the *communicability* $(exp(A))_{ij}$ between location i and protein j as indication of how likely protein j is to localize to location i.

To make prediction easier, communicability values are normalized i.e. they are converted to a value between 0 and 1. For this we use min-max normalization defined as follows:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (6)$$

Analogously with the fraction in the f-neighbour algorithm 1, we choose some value for the communicability and assign proteins to locations only when their communicability exceeds that value.

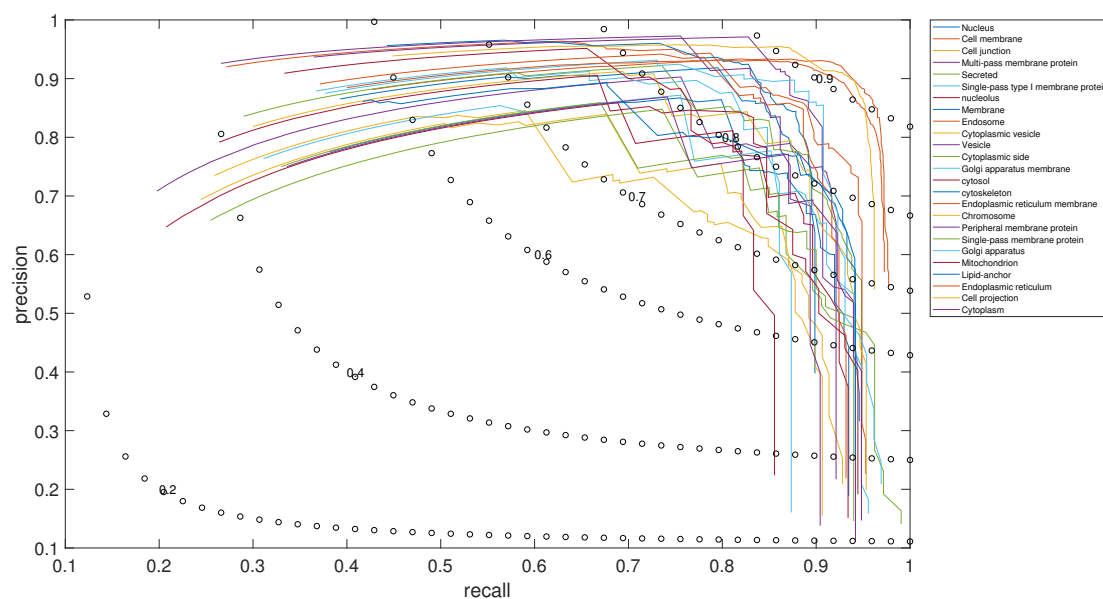


Fig. 3. The precision as a function of the recall for the Homo Sapiens data set. The dotted lines represent the f1 iso-curves. We see that all locations score above 0.7 and most above 0.8.

Table 2. The best f1-score for both data sets and corresponding fraction, precision and recall

data set	fraction	average f1	average precision	average recall
Homo Sapiens	0.34	0.81	0.77	0.83
Saccharomyces Cerevisiae	0.30	0.89	0.94	0.85

3 Results

f-neighbour

We ran Algorithm 1 (f-neighbour) for values of f between 0.01 and 1 with a step size of 0.01, a prediction was made for every value of f and the precision and recall for that prediction were stored. We built the list of P proteins we want to predict by randomly sampling them out of the protein interaction network, 15% was sampled. The locations of these proteins were assumed to be unknown. In figure 3 the precision is plotted as a function of the recall for the Homo Sapiens data set. Some locations have better performance than others. This is most likely due to the unequal representation of locations in the data set as shown in Figure 1 and figure 2. Locations that are better represented in the data tend to be more predictable. One should keep in mind however that for the top locations assigning the location to any arbitrary protein would give a correct prediction around 80% of the time. A much better indication of the effectiveness of the f-neighbour algorithm is therefore the performance of the less represented locations.

In Figure 4, we show the precision as a function of the recall for the Saccharomyces data set using Algorithm 1. We see very similar results as with the Homo Sapiens data set from Figure 3, having almost the same relative score differences between locations and a slightly higher overall score. Table 2 shows the highest f1-score for both data sets. The higher f1-score of the Saccharomyces data set compared to the Homo Sapiens data set seems to mostly be a result of the higher average precision of the former, the recall being approximately equal.

One notable observation is a 'jump' in f1, precision, and recall over all locations at fractions of 0.25, 0.33, 0.50, and 0.66. Both data sets seem to suffer the same effect at the same fraction values. This anomaly comes from the distribution of interactions per protein within each data set. Take for instance a protein with 2 neighbours. There are 3 possible outcomes of the f-neighbour algorithm for this protein:

1. none of its neighbours localize to a particular location.
2. exactly one of its neighbours localizes to a particular location.
3. both of its neighbours localize to a particular location.

For all values of f between 0.5 and 1 the protein will only be assigned to the location in the case of outcome 3. For values of f smaller or equal to 0.5 the protein will be assigned the location in the case of outcome 2 or 3. Visually this would manifest itself as a sudden change in f1-score at the 0.5 mark. It is for this reason that we observe a change in f1-score at values of f that correspond to common fractions.

node2vec

Figure 5, 6, 7, and 8 were created using feature vectors of dimension 128 using the Homo Sapiens data set, other feature vector dimensions and the Saccharomyces data set showed similar results and have been left out.

Figure 5 shows the relationship between jaccard similarity of protein locations sets (i.e. the set of locations a protein localizes to) and the euclidean distance between protein feature vector representations. Most distances lie between 1.5 and 5.0 with a few outliers. Most proteins have a similarity below 0.4. The same pattern of common fractions visible in figures 3 and 4 is also visible here, for instance around 0.5 and 0.75 we see thick lines due to the fact a lot of proteins share half or 3/4 of their locations. The hypothesized inverse correlation between similarity and distance is not present. Only a very slight negative correlation of -0.0708 can be seen, which would indicate no useful representation was created by node2vec.

Likewise when we look at the in- vs out-group similarity in Figures 6, 7, and 8, we don't see a significant distinction between in group and out

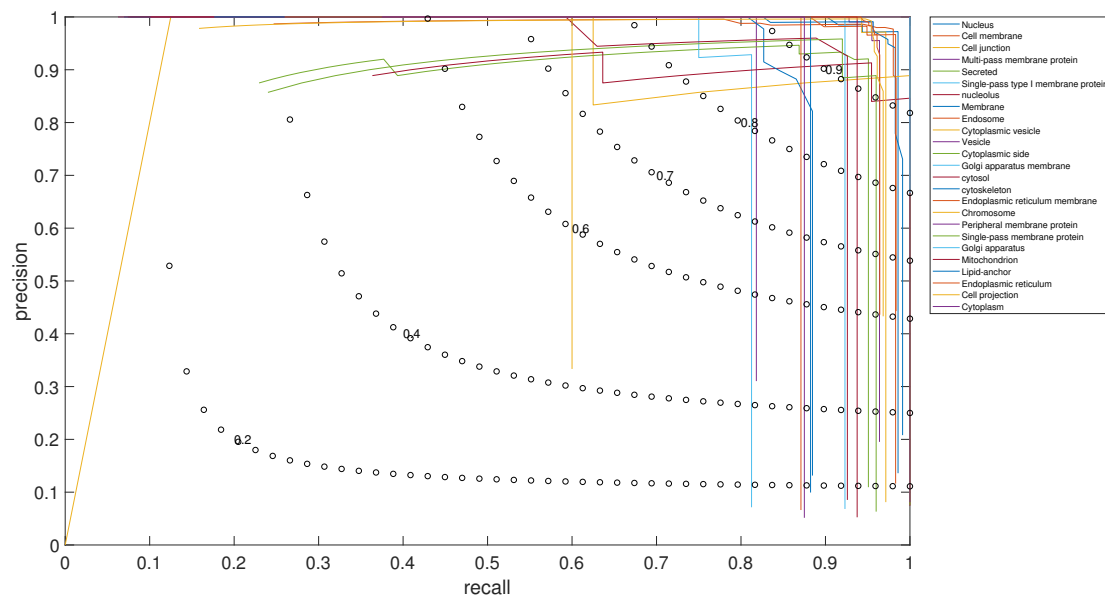


Fig. 4. The precision as a function of the recall for the Saccharomyces data set using Algorithm 1. The dotted lines represent the f1 iso-curves. We see that most locations score above 0.9.

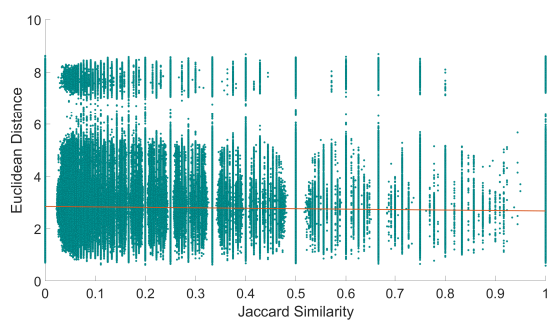


Fig. 5. The jaccard similarity of proteins location sets vs the euclidean distance between feature vector representations of proteins. Based on the Homo Sapiens data set. It shows a slight negative Pearson correlation coefficient of -0.0708, but not the strong negative correlation necessary for prediction.

group distances for the nucleus, cytoplasm, or secreted locations. This too indicates that the feature vectors do not contain use full information about protein locations.

Based on the results of these two metrics we conclude that the relationship between proteins and their locations was not captured by node2vec and subsequently attempts to use the feature vectors for protein location predictions were left out in favor of other more promising methods.

weighted-f-neighbour

Figure 9 shows the result of the weighted-f-neighbour algorithm for the Homo Sapiens data set. Using the similarity measure from SimRank ($C=0.8$, $k=5$) as weights. We see that there is almost no change in recall (over different values of f) within locations. From the definition of recall we find that this must mean that as f increases we are not assigning more proteins to locations (not even wrongly). This is caused by the fact that the similarity between nodes is typically very low, such that even two direct neighbours typically have a similarity of 0, resulting in the fraction

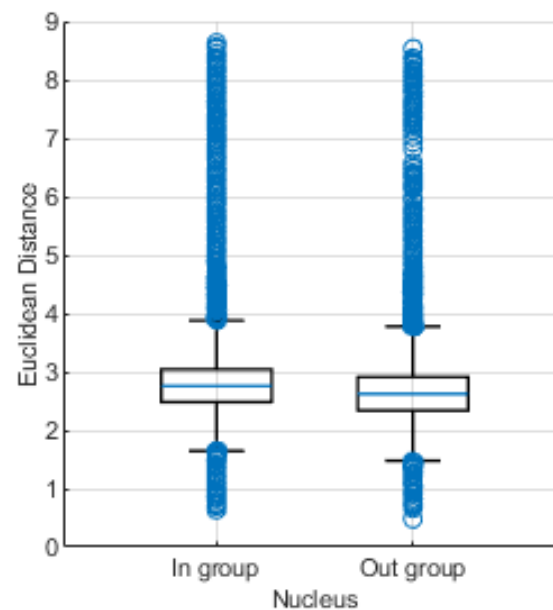


Fig. 6. Box plot of the pairwise euclidean distance of protein feature vectors within the nucleus (in group) and pairwise euclidean distance between proteins in the nucleus and proteins not in the nucleus (out group). We expect to see smaller distance between proteins both in the nucleus (in group) than the distance between pairs with one protein in the nucleus and one not in the nucleus (out group)

$\frac{\text{similarity neighbours at location}}{\text{similarity neighbours total}}$ being very small (< 0.01) or even 0 and thus the protein is not assigned to the location even for the smallest value of f (0.01). This is unexpected because we would expect most neighbours to have at least some similarity. Weights did not improve the performance of the Saccharomyces data set either, these results are not shown.

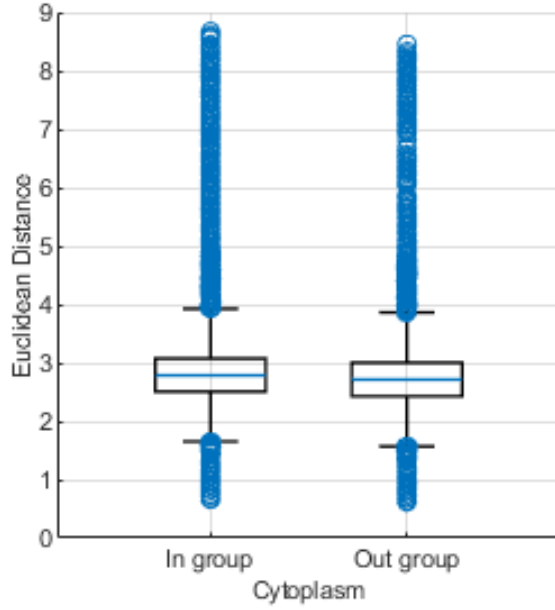


Fig. 7. Box plot of the pairwise euclidean distance of protein feature vectors within the cytoplasm (in group) and pairwise euclidean distance between proteins in the cytoplasm and proteins not in the cytoplasm (out group). We expect to see smaller distance between proteins both in the cytoplasm (in group) than the distance between pairs with one protein in the cytoplasm and one not in the cytoplasm (out group).

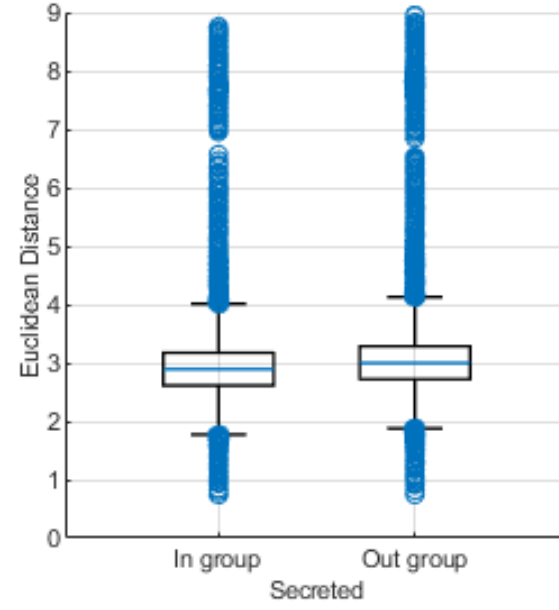


Fig. 8. Box plot of the pairwise euclidean distance of protein feature vectors in the secreted and pairwise euclidean distance between proteins in the secreted group and proteins not in the secreted. We expect to see smaller distance between proteins both in the secreted (in group) than the distance between pairs with one protein in the secreted and one not in the secreted (out group).

k-walks

To approximate the matrix exponential for the adjacency matrix of both the Homo Sapiens and Saccharomyces data set, the second method was used with $k = 10$. Calculating this approximation is a lot faster and requires less memory than the Pade approximant and the outcomes are nearly identical (the average difference over all entries of the output matrix being 0.0002). Figure 10 shows the results for the Homo Sapiens data set. We see a clear distinction between the more common locations (Nucleus, Endoplasmic Reticulum) and the less common locations. Figure 11 shows the same distinction between common and uncommon locations for the Saccharomyces data set as seen in figure 10. Taking a closer look at figures 10 and 11 we see that for most locations we have a very low precision, this indicates a lot of proteins are wrongly predicted to be at a certain location. The low precision might be a result of the tight distribution of distance (defined as $\exp(A)_{ij}$) values between the nodes as can be seen in figure 12 and 13. Closer examination of the overall protein interaction networks reveals a possible cause of this tight distribution. The networks are strongly connected; In the case of the Homo Sapiens network the average number of interactions per protein is 153.2. Then the average number of proteins reachable in 2 steps is 153.2^2 which is already more than the total number of proteins in the network. Taking into account that the locations we consider have a minimum of 500 proteins localized to them (and thus in the case of the k-walks algorithm have an edge with them) it stands to reason that almost all proteins are connected to all 25 locations in 1 or 2 steps.

Overview

Table 3 gives an overview of the top average performance over all locations for both data sets. We can immediately see that the Algorithm 1 f-neighbour is the best performing method for all locations except the Endoplasmic Reticulum in the Sarrachymces data set.

Locations	Homo Sapiens F1			Sarrachomyces F1	
	f-neighbour	w-f-neighbour	k-walks	f-neighbour	k-walks
Endoplasmic reticulum	0.83	0.24	0.60	0.98	0.99
Nucleus	0.92	0.36	0.54	0.97	0.58
Single-pass membrane protein	0.73	0.44	0.22	0.94	0.09
cytosol	0.83	0.22	0.35	0.90	-
Peripheral membrane protein	0.84	0.35	0.08	0.94	0.27
Lipid-anchor	0.84	0.36	0.16	0.87	0.01
Chromosome	0.81	0.32	0.09	0.93	0.03
Cell membrane	0.85	0.43	0.73	0.90	0.13
cytoskeleton	0.87	0.32	0.20	0.93	0.56
Golgi apparatus	0.84	0.26	0.32	0.97	0.05
Mitochondrion	0.89	0.42	0.34	0.97	0.05
Endosome	0.79	0.23	0.39	1.00	0.04
Vesicle	0.81	0.27	0.11	0.96	0.09
Single-pass membrane protein	0.81	0.25	0.18	0.94	0.11
Multi-pass membrane protein	0.87	0.36	0.32	0.97	0.00
Endoplasmic reticulum membrane	0.87	0.25	0.17	0.97	0.11
Cell projection	0.78	0.40	0.08	0.75	0.19
nucleolus	0.87	0.41	0.42	1.00	0.13
Cytoplasmic side	0.79	0.34	0.07	0.94	0.17
Cell junction	0.89	0.41	0.21	1.00	0.07
Golgi apparatus membrane	0.83	0.22	0.13	0.97	0.13
Secreted	0.82	0.53	0.17	0.94	0.26
Cytoplasm	0.91	0.34	0.21	0.97	0.03
Membrane	0.92	0.37	0.24	0.97	0.20
Cytoplasmic vesicle	0.82	0.25	0.77	0.93	-
Average	0.81	0.33	0.21	0.89	0.58

Table 3. The top scores by location and method. For every method the top location score is in bold. The best result over all methods in terms of average performance is also in bold.

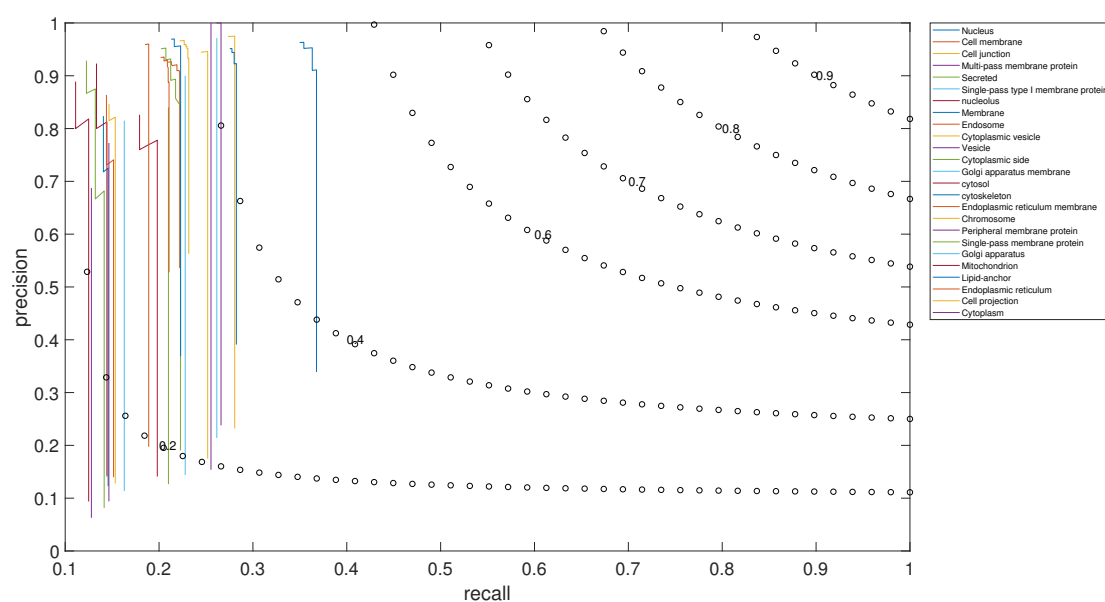


Fig. 9. Precision as a function of recall using the weighted f-neighbour algorithm. Fraction values between 0 and 1 with steps of 0.01. Using the Homo Sapiens data set. The dotted lines represent the f1 iso-curves.

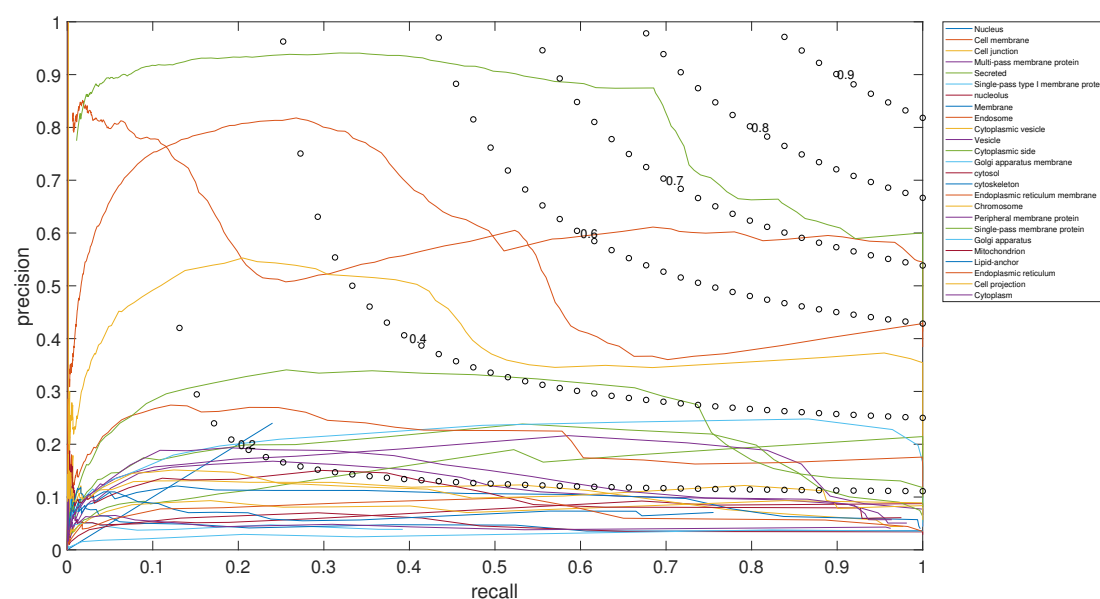


Fig. 10. Precision as a function of the recall. Using k=10 on the Homo Sapiens data set and communicability values between 0 and 0.25. The dotted lines represent the f1 iso-curves.

4 Discussion

We found that, compared to a simple graph based algorithm, more complex methods are less effective at protein location prediction. The feature vector representation acquired through the node2vec algorithm did not show convincing signs of having captured information about protein locations. Using SimRank similarity as weights did not improve the performance of the f-neighbour algorithm and the results strongly imply no actual similarity between nodes was found by SimRank.

In the same way the communicability measure from the k-walks algorithm

was not helpful as an indicator of protein sub cellular location.

The success of the f-neighbour is not surprising, the initial hypothesis, that if proteins interact with one another then they must also share a location, is logically sound. We kept the method intentionally simple because we wanted to use it for comparison, but it would be interesting to investigate its utility more and perhaps improve on it. For instance the relationship between the predictive power of the algorithm and the number of interactions is an interesting question. It stands to reason that

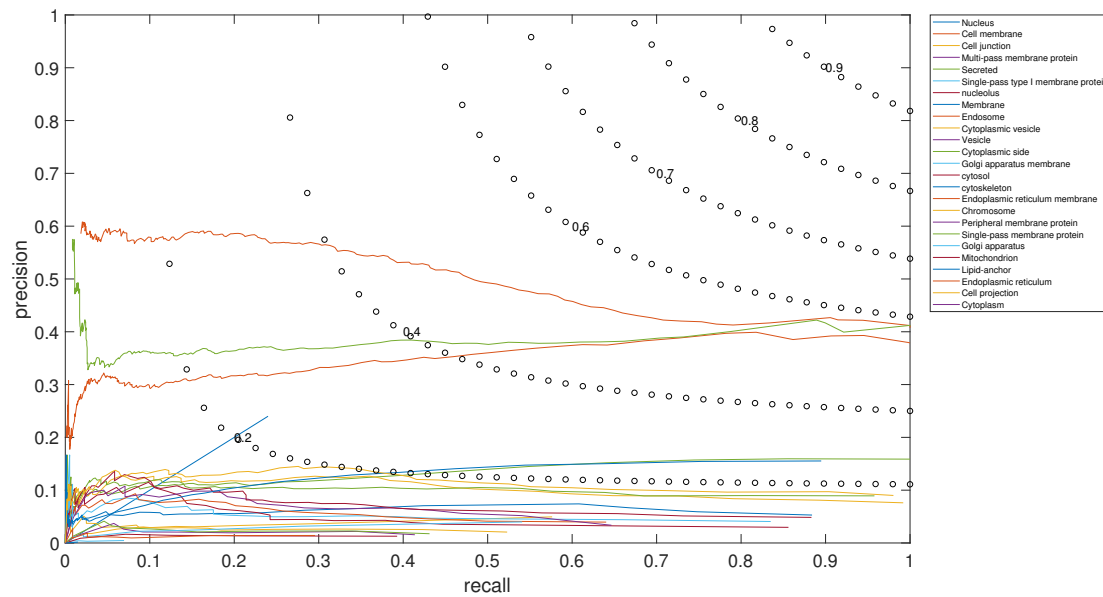


Fig. 11. Precision as a function of the recall. Using $k=10$ on the *Saccharomyces* data set and communicability values between 0 and 0.25. The dotted lines represent the f1 iso-curves.

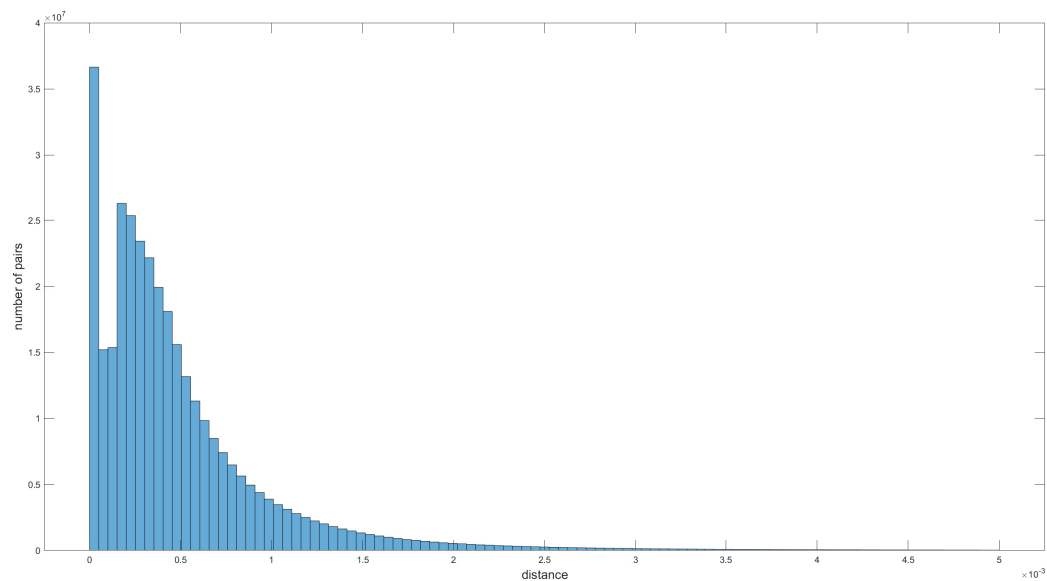


Fig. 12. The distribution of distances between nodes in the *Homo Sapiens* data set. Note that most nodes are very close together.

location prediction for a protein with only 2 known interactions will not be effective using the f-neighbour algorithm. More generally we could try to find a relationship between the 'confidence' of the f-neighbour algorithm and the degree of proteins.

More surprising than the high performance of the f-neighbour algorithm is the failure of the other methods to effectively predict, or assist in, protein localization. We know from Grover and Leskovec[6] that node2vec can be used to predict protein *interactions*, however our results indicate that the relation between the *location* sets of the proteins is not recapitulated by node2vec. The number of interactions per protein is a strong factor in

the low performance of both the SimRank and k-walks algorithms. From the formula of SimRank we see immediately why a strongly connected network might not give use full results. In the first term we divide by the number of incoming connections of node a times the number of incoming connections of node b , with an average of 153.2 interactions per protein this will make the similarity approach zero very rapidly. In the same sense, as already explained, the connectivity between proteins in the network is not a use full measure since most proteins are connected to most other proteins in 2 steps or less.

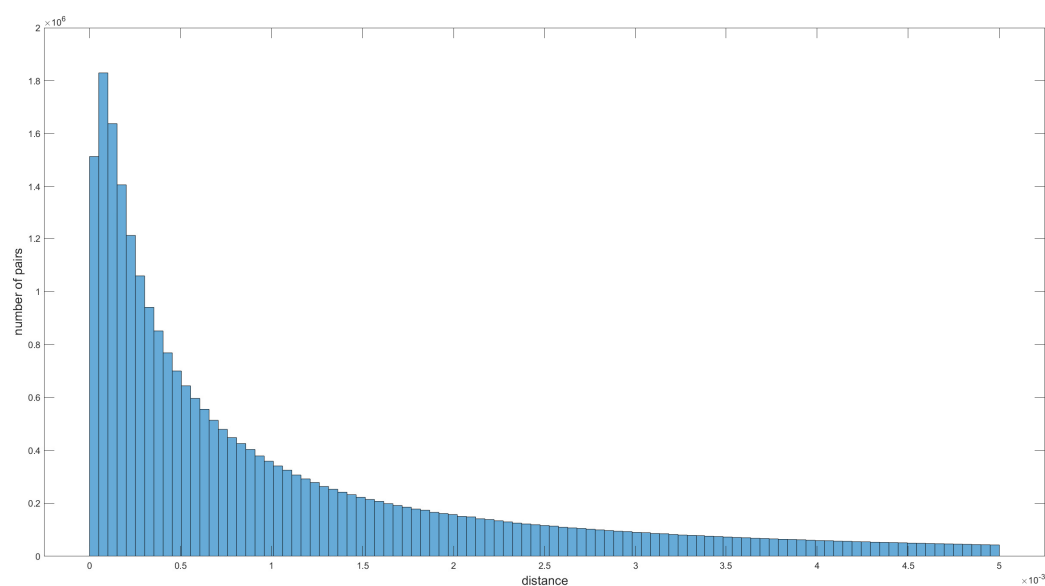


Fig. 13. The distribution of distances between nodes in the Saccharomyces data set. Note that most nodes are very close together.

In conclusion, the f-neighbour algorithm outperforms the other three algorithm in all respects. The most important difference between the f-neighbour algorithm and the other algorithms is that the f-neighbour algorithm is very *localized* i.e. it looks only at the direct neighbourhood of a protein whereas the other algorithms are *general* i.e. they take the full graph into account.

References

- [1]Biogrid. <https://thebiogrid.org/>. Accessed: 2018-04-05 Version: 3.5.173.
- [2]Uniprot. <https://www.uniprot.org/>. Accessed: 2018-04-05.
- [3]Olof Emanuelsson. Predicting protein subcellular localisation from amino acid sequence information. *Briefings in Bioinformatics*, 3(4): 361–376, 12 2002. ISSN 1467-5463. doi: 10.1093/bib/3.4.361. URL <https://dx.doi.org/10.1093/bib/3.4.361>.
- [4]Desmond J. Higham Ernesto Estrada. Network properties revealed through matrix functions. *Society for Industrial and Applied Mathematics*, 2010. doi: 10.1137/090761070.
- [5]Jennifer Widom Glen Jeh. Simrank: A measure of structural-context similarity. *Stanford University*, 2002. doi: 10.1145/775047.775126.
- [6]Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.
- [7]M.-C. Hung and Wolfgang Link. Protein localization in disease and therapy. *Journal of Cell Science*, 124(20):3381–3392, oct 2011. ISSN 0021-9533. doi: 10.1242/jcs.089110. URL <http://jcs.biologists.org/content/124/20/3381>.
- [8]Oren Z. Kraus, Jimmy Lei Ba, and Brendan J. Frey. Classifying and segmenting microscopy images with deep multiple instance learning. *Bioinformatics*, 32(12):i52–i59, jun 2016. ISSN 1367-4803. doi: 10.1093/bioinformatics/btw252.
- [9]Oren Z Kraus, Ben T Grys, Jimmy Ba, Yolanda Chong, Brendan J Frey, Charles Boone, and Brenda J Andrews. Automated analysis of high-content microscopy data with deep learning. *Molecular Systems Biology*, 13(4):924, apr 2017. ISSN 1744-4292. doi: 10.15252/msb.20177551.
- [10]H. Padé. Sur la représentation approchée d’une fonction par des fractions rationnelles, thesis, pp. 1–93 supplement. 1892.
- [11]Tanel Pärnamaa and Leopold Parts. Accurate Classification of Protein Subcellular Localization from High-Throughput Microscopy Images Using Deep Learning. *G3: Genes|Genomes|Genetics*, 7(5):1385–1392, may 2017. ISSN 2160-1836. doi: 10.1534/g3.116.033654.
- [12]Mengli Xiao, Xiaotong Shen, and Wei Pan. Application of deep convolutional neural networks in classification of protein subcellular localization with microscopy images. *Genetic Epidemiology*, 43(3): 330–341, apr 2019. ISSN 07410395. doi: 10.1002/gepi.22182. URL <http://doi.wiley.com/10.1002/gepi.22182>.
- [13]Ping He Ling Chen Xiaohua Xu, Lin Lu. Protein localization prediction using random walks on graphs. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3654884/>, 2013.