CSCI 5352 Problem set 3

Jake Krol

February 2025

1 Q1a

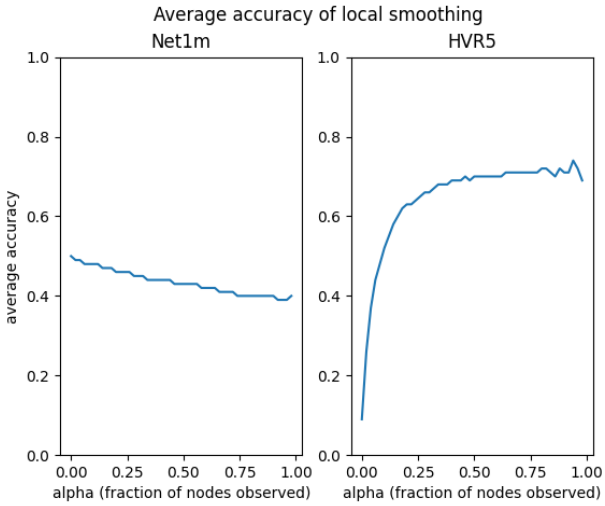

Figure 1: Accuracy at various α (fraction of observed nodes) for Net1m and HVR5 networks. Accuracy was averaged over $k = 100$ iterations for each α.

1.1 Average accuracy curve discussion

Local smoothing node attribute classification of the 1-mode projection network of Norwegian board of directors (**net1m**) correctly classified (measured by average accuracy over $k = 100$ iterations per α [fraction of observed nodes]) $\approx$ **50**% of unobserved nodes at **very low** α, $\approx$ **45**% at **mid-range** α, and performed worst $\approx$ **40**% at **very high** α. In contrast, local smoothing performance on the Malaria HVR5 network (**HVR5**) was close to **0** at **very low** α, but performance rapidly increased as more nodes were observed. At **mid-range** α, HVR5 local smoothing average accuracy was high at $\approx$ **70**%, but the performance asymptoted. Where at **very high** α, the performance was comparable to mid-range

$\approx$ **70**%. Note, for $\alpha = 1$, there are no unobserved node attributes, and I set the accuracy average accuracy to 100%.

The key **difference** in the average accuracy of local smoothing curves between the net1m and HVR5 networks is that **net1m's average accuracy decreases** as more nodes are observed while **HVR's accuracy increases** as more nodes are observed. **Another interesting difference** is the performance at $\alpha = 0$. When all nodes are unobserved, half of the node attributes are correctly classified in net1m, yet nearly all nodes are misclassified in HVR5.

1.2 What I learned about local smoothing

I **learned** that 1) if a network is not assortative, then observing more nodes may not improve performance of a local smoothing procedure. And, local smoothing performance can decrease if a network is disassortative. Also, I **learned** that 2) the baseline performance of a local smoothing model (at $\alpha = 0$) for label classification depends on the set size of categorical labels in the truth set.

1.3 Structural insights from average accuracy curves

Solely from the average accuracy local smoothing curves, I hypothesize that **net1m** is not assortative w.r.t gender labels and probably slightly dissassortive. Since, average accuracy decreases as more nodes are observed. For **HVR**, I hypothesize that the network is mostly assortative since average accuracy of local smoothing increases until $\alpha = 0.5$. Furthermore, I hypothesize that there exists some non-assortative local neighborhoods which explain the inability to correctly classify the remaining $\approx 30\%$ of unobserved nodes.

1.4 Expected accuracy of random classifier

Accuracy (ACC) is the fraction of correct predictions (true positives [TP] and true negatives [TN]) out of all predictions (positive [P] and negative [N]). For a binary classifier, accuracy is

$$ACC = \frac{TP + TN}{P + N}$$

For random binary classification, accuracy is

$$ACC = \frac{pn_1 + (1-p)n_0}{n_1 + n_0}$$

$$\text{if p} = \frac{1}{2} \text{ , then}$$

$$ACC = p\frac{n_1 + n_0}{n_1 + n_0} = p = \frac{1}{2}$$

Where, n_0 and n_1 represent the counts of ground truth labels valued as 0 and 1. And, p is the probability of a prediction being 1 and $1-p$ s the probability of

a prediction being 0. If there's no prior knowledge of class balance, then $p = \frac{1}{2}$ for a binary classifier.

Generalizing the binary case to $\{1 \dots k\}$ possible categorical outcomes, random classification accuracy is a weighted sum over the counts of each class and their associated probabilities, $p_1 \dots p_k$. And, the denominator sums over counts of each class. **Therefore, the expected accuracy of a random classifier is**

$$ACC = \frac{\sum_{i=1}^{k} p_i n_i}{\sum_{i=1}^{k} n_i}$$

$$\text{if } p_i \text{ is constant at } \frac{1}{k}, \text{ then}$$

$$ACC = \frac{1}{k} \frac{\sum_{i=1}^{k} n_i}{\sum_{i=1}^{k} n_i} = \frac{1}{k}$$

Importantly, if the random classifier **does not incorporate the class distribution**, then the probability vector is a vector of fractions $p_1 = \dots p_k = \frac{1}{k}$, and the expected accuracy simplifies to $\frac{\mathbf{1}}{\mathbf{k}}$. However, **if the distribution of classes is used to weigh probability of random classifier**, then the accuracy changes as a function of the class distribution in the truth set.

For **net1m**, there are 2 possible gender classes. A random classifier **with no prior knowledge** of class balance is simply $\frac{1}{2}$.

Local smoothing on net1m in figure 1 at $\alpha = 0$ corresponds to the case where a random classifier is applied to all nodes. **And, the average** $ACC = 0.5$ **at** $\alpha = 0$ **(figure 1) makes sense** because the random classifier's probabilities were uniformly $\frac{1}{k}$ and there are only two categorical labels in the net1m network. **With class balance knowledge**, a majority choosing classifier's accuracy would be higher since the classes are unevenly distributed. $n_0 = 908$ and $p_0 = 908/1421 \approx 0.64$ and $n_1 = 513$ and $p_1 = 513/1421 \approx 0.36$.

For **HVR**, there are 6 possible classes. A random classifier **with no prior knowledge of class balance is** $\frac{1}{6}$.

For **local smoothing on HVR**, figure 1 at $\alpha = 0$ shows the average $ACC \approx 0.1$ is slightly less than the $\frac{1}{6} = 0.1\bar{6}$ expected from a random classifier. Maybe, adding more repetitions before averaging accuracy would converge to $\frac{1}{6}$. Notably, the HVR class attributes are unevenly distributed in the network with most of hvr sequences with the label of "4". So, a majority classifier would get much higher accuracy than a completely random classifier.

Note, at non-zero α there are also cases where a randomly uniform baseline is used (if all neighbor attributes are also unobserved), but $\alpha = 0$ is the most straightforward case to discuss.

2 Q2a

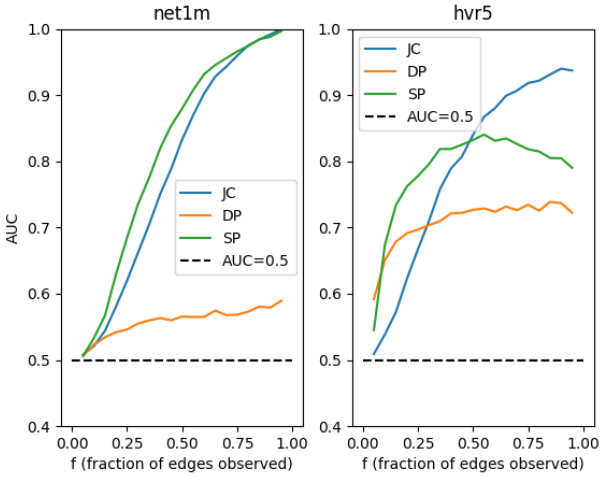

Figure 2: Unsupervised link prediction using Jaccard Coefficient (JC), degree product (DP), and shortest path (SP) predictors for net1m and HVR networks. Link prediction performance is AUC (vertical axis) averaged over $k = 5$ iterations at each f (horizontal axis). Where, f is the fraction of edges observed in the network.

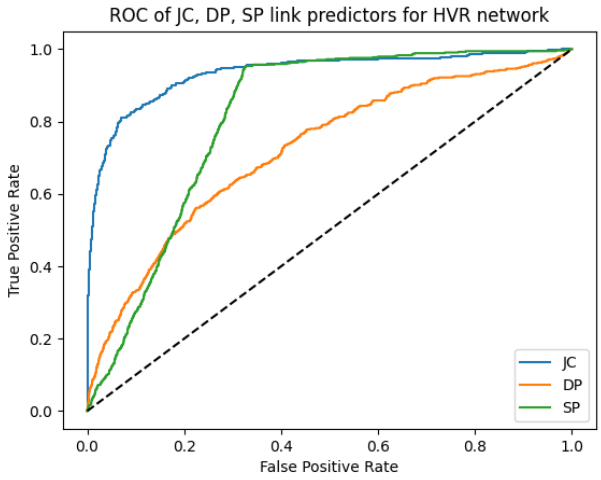

Figure 3: Receiver operating characteristic of link prediction methods for the HVR network when $f = 80\%$ of edges are observed. The horizontal axis is the false positive rate (FPR), and the vertical axis is the true positive rate (TPR). Each line depicts the change in FPR and TPR at various decision thresholds used to classify edge scores. The scoring methods are Jaccard Coefficient (JC), degree product (DP), and shortest path (SP).

2.1 Why one predictor performs well at low f and relatively poor at high f?

DP's AUC is relatively high at low f yet relatively low (figure 2, right), compared to JC/SP, and I think this may be explained by empirical network degree distributions being heavy-tailed. Since DP favors linking nodes with high degree, this will make the degree of nodes with pre-existing high-degree even higher. This would produce a heavy-tailed degree distribution where many nodes have low degree, and a small fraction have high degree. However, I suspect the initial performance boost wanes when the underlying data generating process of edges is need to correctly complete links, not purely DP ranking. **Maybe, the degrees produced by DP are unrealistically high when there are already many edges observed** and many high degree nodes already exist.

2.2 What does performance difference suggest about differences in network structure?

Since the AUC of each link predictor generally is better than random as the amount of edges observed increases, I think this means that edges are not inde-

pendent of one another and the network structure depends on the underlying data generating processes.

Particularly, for **net1m**, I think the network is not degree assortative since the DP predictor hardly performs better than a baseline classifier. Meanwhile, JC and SP effectively predict the correct edges as more are observed, and I hypothesize that the average shortest path in net1m differs from a random graph model with similar nodes, edges, degree, etc. I don't have clear intuition about how JC performing well affects structure. Maybe, the network has a big LCC and few disconnected components. Since, it would be hard to predict edges if the network is mostly small components with few shared neighbors.

The same logic for DP and JC applies for the **HVR** network, except SP is not as performant. In fact, the AUC of SP decreases for $f > 0.5$, and I hypothesize that shortest paths between nodes is probably not a meaningful part of the underlying generating process. In terms of structure, I think this means the average path length would be similar to a random graph model.

2.3 What does ROC shape imply regarding accuracy of link predictors?

I chose the **HVR** network to view ROCs of the three link predictors at $f = 0.8$. The shape of the JC curve rapdily approaches the upper left corner. Meaning, a JC scoring threshold can separate out most of the TPs without including too many FPs. For SP, the TPR increases linearly until a certain point where the scores of FPs and TPs are mixed causing the slope to decrease suddenly. Finally, the DP scoring curve visually has the lowest AUCROC which agrees with the results of figure 2 at $f = 0.8$. **Overall, the curves indicate how well each link predictor can separate TPs and FPs based on their score function.**

3 Extra credit

- **I read** "Denoising large-scale biological data using network filters" (Kavran et al. 2021).
- The **research question** is how to denoise biological measurement data and highlight true biological signal.
- The **approach** proposed is to use network filter operations (smoothing or sharpening) in community detected local neighborhoods to account for graphs with mixed (dis)assortativity. The filter operations proposed are neighbor-based mean/median corrections which pull a node closer to its neighbors' measurement values in assortative communities. And, nodes are pushed further from their neighbors' measurement values in disassortative communities. A no filter baseline, network filters, and diffusion-based denoising methods are experimentally compared on synthetic data. Two types of synthetic graphs are tested: 1) random graphs without modularity

and 2) random graphs with modularity enforced. Finally, the framework is applied as a pre-processing step in a machine learning model used to predict the difference of protein expression in human cancer.

- The paper **did** many things **well**. The approach and experimental design was very clearly expressed. For instance, I really enjoyed the intuitive analogies of network filters to image processing methods. Also, the evaluation metrics were logical and results were effectively communicated with understandable plots.
- The paper maybe **could have improved** on motivating why diffusion methods were chosen as the benchmarks and discussing more related work for de-noising biological measurement data.
- I agree with the **extensions** of this work proposed in the discussion. Where, specialized, task-specific network filters could be explored in other domains. Also, I loosely know that image classification models expand training datasets by using noise-permuted images, so I'm curious if these filters could be used to permute network datasets to increaese training data size and generalizability of machine learning models.

4 References

Kavran, A.J., Clauset, A. Denoising large-scale biological data using network filters. BMC Bioinformatics 22, 157 (2021). https://doi.org/10.1186/s12859-021-04075-x

5 Code

GitHub repo. See ps3_ prefixed scripts.