A. Proof for Theorem 1

Proof. Denote by $f_x^{(l)}$ the pre-activated feature of $h_x^{(l)}$, i.e. $\frac{1}{\widetilde{\deg}(x)} \cdot \sum_{z \in \widetilde{N}(x)} W_l h_z^{(l-1)}$, for any $l = 1..k$, we have

$$\frac{\partial h_x^{(l)}}{\partial h_y^{(0)}} = \frac{1}{\widetilde{\deg}(x)} \cdot \operatorname{diag}\left(1_{f_x^{(l)}>0}\right) \cdot W_l \cdot \sum_{z \in \widetilde{N}(x)} \frac{\partial h_z^{(l-1)}}{\partial h_y^{(0)}}$$

By chain rule, we get

$$\begin{aligned}\frac{\partial h_x^{(k)}}{\partial h_y^{(0)}} &= \sum_{p=1}^{\Psi} \left[\frac{\partial h_x^{(k)}}{\partial h_y^{(0)}}\right]_p \\ &= \sum_{p=1}^{\Psi} \prod_{l=k}^{1} \frac{1}{\widetilde{\deg}(v_p^l)} \cdot \operatorname{diag}\left(1_{f_{v_p^l}^{(l)}>0}\right) \cdot W_l\end{aligned}$$

Here, Ψ is the total number of paths $v_p^k v_p^{k-1}, ..., v_p^1, v_p^0$ of length $k+1$ from node x to node y. For any path p, v_p^k is node x, v_p^0 is node y and for $l = 1..k-1$, $v_p^{l-1} \in \widetilde{N}(v_p^l)$.

As for each path p, the derivative $\left[\frac{\partial h_x^{(k)}}{\partial h_y^{(0)}}\right]_p$ represents a directed acyclic computation graph, where the input neurons are the same as the entries of W_1, and at a layer l. We can express an entry of the derivative as

$$\left[\frac{\partial h_x^{(k)}}{\partial h_y^{(0)}}\right]_p^{(i,j)} = \prod_{l=k}^{1} \frac{1}{\widetilde{\deg}(v_p^l)} \sum_{q=1}^{\Phi} Z_q \prod_{l=k}^{1} w_q^{(l)}$$

Here, Φ is the number of paths q from the input neurons to the output neuron (i,j), in the computation graph of $\left[\frac{\partial h_x^{(k)}}{\partial h_y^{(0)}}\right]_p$. For each layer l, w_q^l is the entry of W_l that is used in the q-th path. Finally, $Z_q \in \{0,1\}$ represents whether the q-th path is active ($Z_q = 1$) or not ($Z_q = 0$) as a result of the ReLU activation of the entries of $f_{v_p^l}^{(l)}$'s on the q-th path.

Under the assumption that the Z's are Bernoulli random variables with the same probability of success, for all q, $\Pr(Z_q = 1) = \rho$, we have $\mathbb{E}\left[\left[\frac{\partial h_x^{(k)}}{\partial h_y^{(0)}}\right]_p^{(i,j)}\right] = \rho \cdot \prod_{l=k}^{1} \frac{1}{\widetilde{\deg}(v_p^l)} \cdot w_q^{(l)}$. It follows that $\mathbb{E}\left[\frac{\partial h_x^{(k)}}{\partial h_y^{(0)}}\right] = \rho \cdot \prod_{l=k}^{1} W_l \cdot \left(\sum_{p=1}^{\Psi} \prod_{l=k}^{1} \frac{1}{\widetilde{\deg}(v_p^l)}\right)$. We know that the k-step random walk probability at y can be computed by summing up the probability of all paths of length k from x to y, which is exactly $\sum_{p=1}^{\Psi} \prod_{l=k}^{1} \frac{1}{\widetilde{\deg}(v_p^l)}$. Moreover, the random walk probability starting at x to other nodes sum up to 1. We know that the influence score $I(x,z)$ for any z in expectation is thus the random walk probability of being at z from x at the k-th step, multiplied by a term that is the same for all z. Normalizing the influence scores ends the proof.

Comment: ReLU is not differentiable at 0. For simplicity, we assume the (sub)gradient to be 0 at 0. □

B. Proof for Proposition 1

Proof. Let $\left[h_x^{(final)}\right]_i$ be the i-th entry of $h_x^{(final)}$, the feature after layer aggregation. For any node y, we have

$$\begin{aligned}I(x,y) &= \sum_i \left\| \frac{\partial \left[h_x^{(final)}\right]_i}{\partial h_y^{(0)}} \right\|_1 \\ &= \sum_i \left\| \frac{\partial \left[h_x^{(j_i)}\right]_i}{\partial h_y^{(0)}} \right\|_1\end{aligned}$$

where $j_i = \operatorname*{argmax}_l \left(\left[h_x^{(l)}\right]_i\right)$. By Theorem 1, we have

$$\mathbb{E}\left[I(x,y)\right] = \sum_l c_x^l \cdot z_l \cdot \mathbb{E}\left[I_x(y)^{(l)}\right]$$

where $I_x(y)$ is the l-step random walk probability at y, z_l is a normalization factor and c_x^l is the fraction of entries of $h_x^{(l)}$ being chosen by max-pooling. By Theorem 1, $\mathbb{E}\left[I_x(y)^{(l)}\right]$ is equivalent to the l-step random walk probability at y starting at x.

□

C. Visualization Results

We describe the details of the heat maps and present more visualization results. The colors of the nodes in the heat maps correspond to their probability masses of either the influence distribution or random walk distribution as shown in Figure 6. As we see, the shallower the color is, the smaller the probability mass. We use the same color for probabilities over 0.2 for better visual effects because there are few nodes with influence probability masses over 0.2. Nodes with probability mass less than 0.001 are not shown in the heat maps.

In Table 6, we present more visualization results to compare the 1) influence distributions under GCNs and the random walk distributions, 2) influence distributions under GCNs with residual connections and lazy random walk distributions. The nodes being influenced and the random walk starting node are labeled square. The influence distributions for the nodes in Figure 6 are computed according to Definition 3.1, under the same trained GCN (Res) models with $2, 4, 6$ layers respectively. We use the hyper-parameters as described in Kipf & Welling (2017) for training the models. The graph (dataset) is taken from the Cora citation network as described in section 6. We compute the random

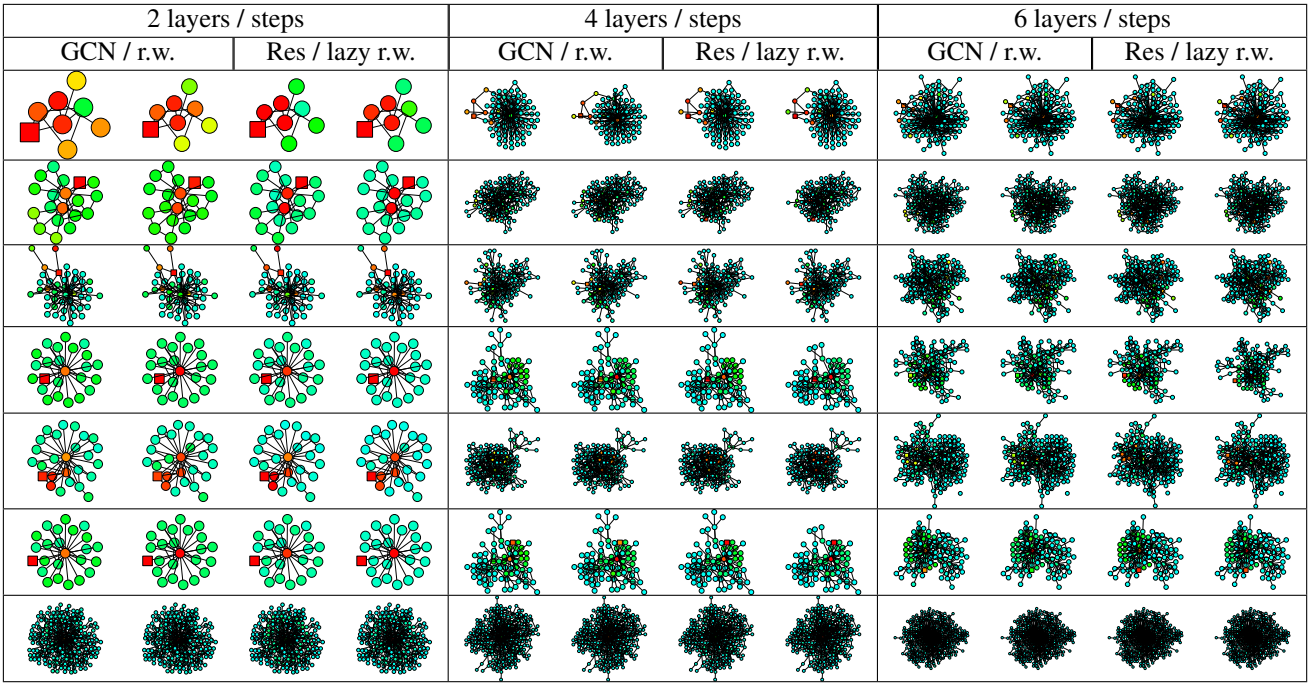
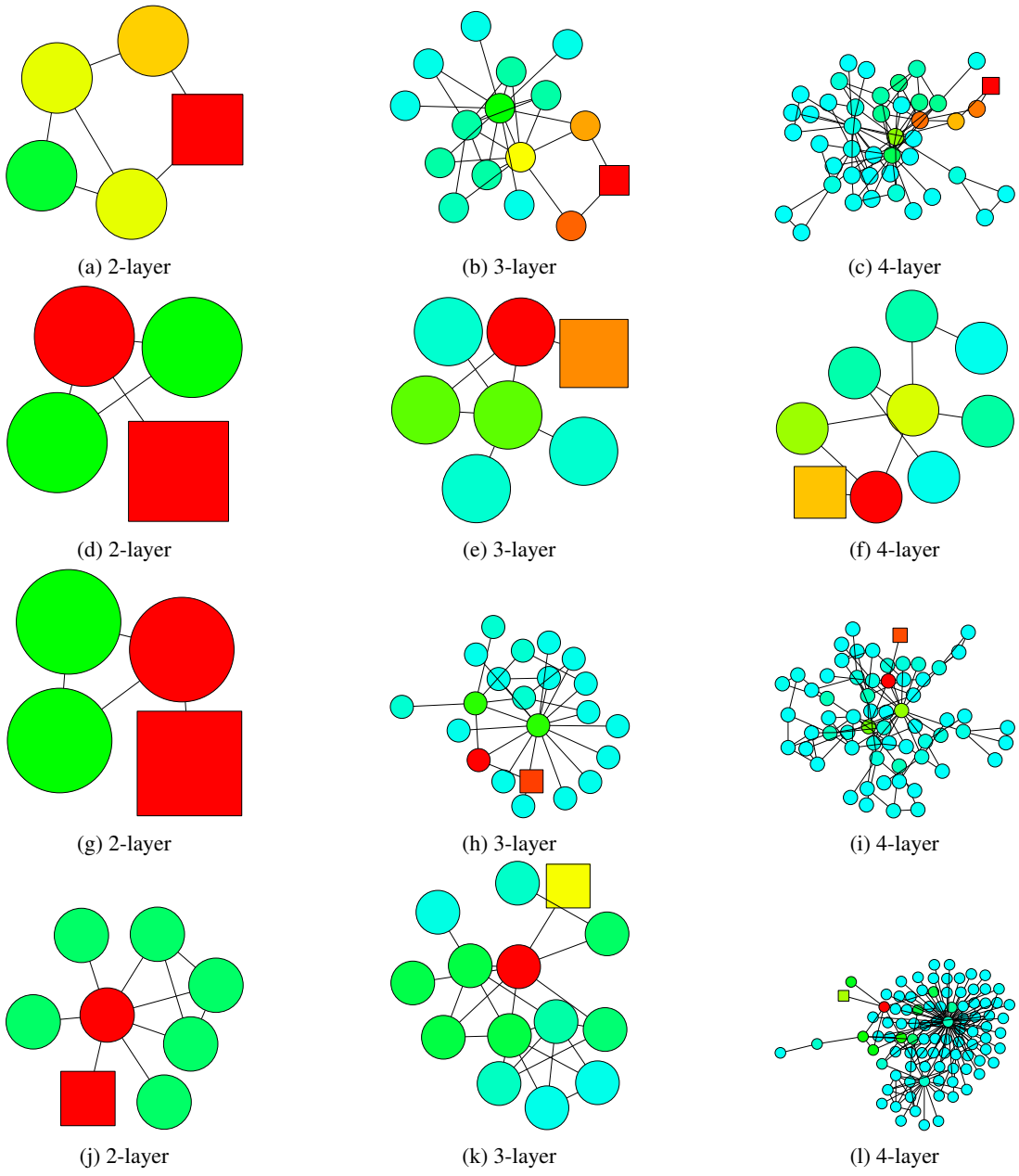


Table 6. Influence distributions for (more) nodes under GCN, GCN-Res, and random walk distributions

Figure 6. Color and probability correpondency for the heat maps

walk distributions according to Definition 3.2 on the graph $\widetilde{G}$. The lazy ranndom walks all share the same lazy factor 0.4, i.e. there's an extra 0.4 probability of staying at the current node at each step. This probability is chosen for visual comparison with the GCN-ResNet. Note that the GCN and random walk colors may differ for nodes that have particularly large degrees because the models we are running follow Equation 2, which assigns less weight to nodes that have larger degrees, rather than Equation 3. The visualization in Figure 5 has the same setting as mentioned above. It is trained for the Cora dataset with a 6-layer JK Net with maxpooling layer aggregation.

Next, we demonstrate subgraph structures where GCN models with 2 layers tend to make misclassification, whereas models with 3 or 4 layers are able to make the correct prediction and vice versa, with real dataset. These visualization results further complement and support the theory illustrated in Figure 1 and Theorem 1. As we see in Figure 7, a model with pre-fixed effective range priors, which looks at 2-hop neighbors, tends to make incorrect prediction if the local subgraph structure is tree-like (bounded treewidth). Thus, it would be desirable to look beyond the direct neighbors and draw information from nodes that are 3 or 4 hops away so as to learn a better representation of the local community. On the other hand, as we see in Figure 8, a model with pre-xied effective range priors, which looks at 3 or 4-hop neighbors, may happen to draw much information from less relevant neighbors and thus cannot learn the right representations, which are necessary for the correct prediction. In the subgraph structures where the random walk expansion explodes rapidly, models with 3 or 4 prefixed layers are essentially taking into account every node. Such global representations might not be ideal for the prediction for the node. In another scenario, despite possessing the locally bounded treewidth structure, because of the "bridge-like" structures, looking at distant nodes might imply drawing information from a completely different community, which would act like noises and influence the prediction results.



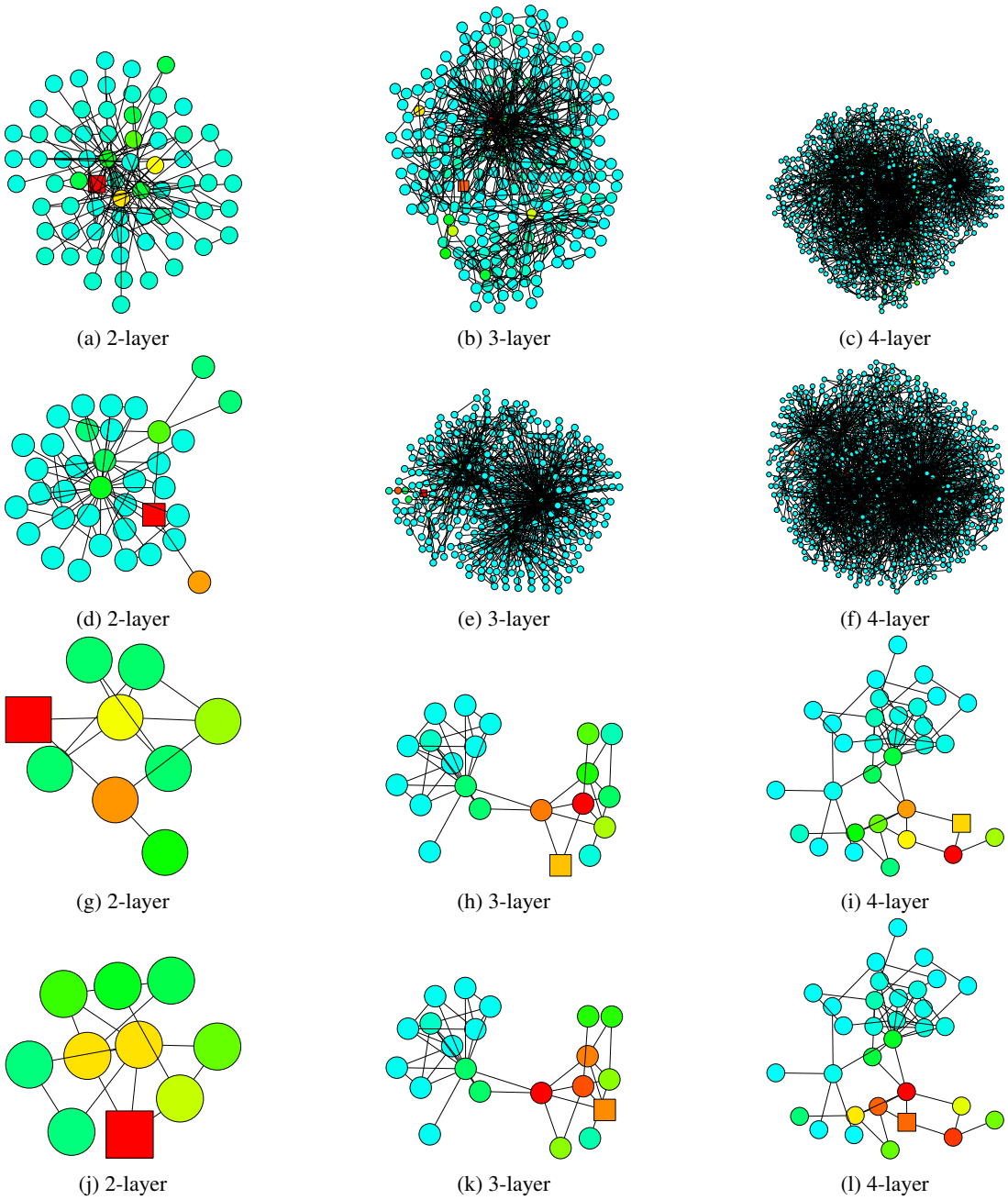
(a) 2-layer (b) 3-layer (c) 4-layer

(d) 2-layer (e) 3-layer (f) 4-layer

(g) 2-layer (h) 3-layer (i) 4-layer

(j) 2-layer (k) 3-layer (l) 4-layer

Figure 7. Subgraph structures where 2-layer GCNs make misclassification, whereas 3 and 4-layer GCNs make the correct prediction.



(a) 2-layer (b) 3-layer (c) 4-layer

(d) 2-layer (e) 3-layer (f) 4-layer

(g) 2-layer (h) 3-layer (i) 4-layer

(j) 2-layer (k) 3-layer (l) 4-layer

Figure 8. Subgraph structures where 3, 4-layer GCNs make misclassification, whereas 2-layer GCNs make the correct prediction.