Technical Appendix for Subgraph Encoding with Bicentric Sphere Node Labeling and Pooling for Link Prediction

Proofs

Proof of Theorem 2

Theorem 2. *The enclosing subgraph $\mathcal{G}^h_{uv}$ has the following properties:*

1. *If $\mathcal{G}^h_{uv}$ is connected, there is $1 \leq Dist(u,v) \leq 2h+1$. Moreover, for any $v_i \in \mathcal{V}^h_{uv}$, there are $0 \leq Dist(u,v_i) \leq 3h+1$ and $0 \leq Dist(v,v_i) \leq 3h+1$.*
2. *If $\mathcal{G}^h_{uv}$ is disconnected, then $\mathcal{G}^h_{uv}$ contains two connected components. Nodes in one component satisfy $0 \leq Dist(v_i,u) \leq h$ and $Dist(v_i,v) = \infty$, while nodes in the other component satisfy $0 \leq Dist(v_i,v) \leq h$ and $Dist(v_i,u) = \infty$.*

Proof. We first prove $Dist(u,v) \leq 2h+1$ in the connected subgraph by contradiction. Suppose $Dist(u,v) > 2h+1$, arbitrary node v_i in the shortest path between u and v satisfy $Dist(u,v_i) + Dist(v,v_i) \geq Dist(u,v) > 2h+1$. That is, there exist a node v_i in the shortest path between u and v, such that $Dist(u,v_i) \geq h+1$ and $Dist(v,v_i) \geq h+1$, which contradicts the condition $v_i \in \mathcal{V}^h_{uv}$. Next, in a connected subgraph, for arbitrary node $v_i \in \mathcal{V}^h_{uv}$, there exists $Dist(u,v_i) \leq Dist(u,v) + Dist(v,v_i) \leq 2h+1+h = 3h+1$. This complete the proof of the first property.

Based on the first property, it is easy to find that $\mathcal{G}^h_{uv}$ is disconnected when $Dist(u,v) > 2h+1$ in $\mathcal{G}$. To this end, for any two nodes $v_i \in \bigcup_{q=0}^{h} \Gamma^h(u)$ and $v_j \in \bigcup_{q=0}^{h} \Gamma^h(v)$, there is $Dist(u,v_i) + Dist(v_i,v_j) + Dist(v_j,v) \geq Dist(u,v)$. Notice that $Dist(u,v_i) \leq h$, $Dist(v_j,v) \leq h$, and $Dist(u,v) = \infty$ in the subgraph $\mathcal{G}^h_{uv}$. Hence, we have $Dist(v_i,v_j) = \infty$. In brief, the subgraph is consist of two components, one of them is node u and its neighborhoods (i.e. $\bigcup_{q=0}^{h} \Gamma^h(u)$), another of them is node v and its neighborhoods (i.e. $\bigcup_{q=0}^{h} \Gamma^h(v)$). This complete the proof of the second property. □

Proof of Theorem 3

Theorem 3. *Given enclosing subgraph $\mathcal{G}^h_{uv}$ of node pair (u,v) and bounding distance node labeling with distance bound D_T, distance representations for each node $v_i \in \mathcal{V}^h_{uv}$ can be obtained by at most $(3h+2-D_T)$-layer 1-WL-GNN.*

Proof. Node labels can be deemed as countable features. Through injective functions like the sum aggregator (Xu et al. 2018), these features become distinguishable in multiset feature spaces. Based on this observation, the core of the theorem reduces to exploring, for a node v_i, how deep a subtree is needed to infer the distances $Dist(u,v_i)$ and $Dist(v,v_i)$. For convenience, we denote by $T^l(v_i)$ the subtree rooted at node v_i with depth l.

Subtree depth analysis:

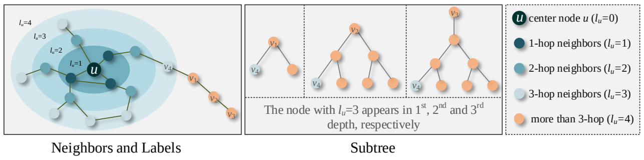

l_u=4
l_u=3
l_u=2
l_u=1
u
v_4
v_1
v_2
v_3
v_1
v_4
v_2
v_4
v_3
v_4
The node with l_u=3 appears in 1st, 2nd and 3rd depth, respectively
u center node u (l_u=0)
1-hop neighbors (l_u=1)
2-hop neighbors (l_u=2)
3-hop neighbors (l_u=3)
more than 3-hop (l_u=4)
Neighbors and Labels
Subtree

Figure 1: An toy example of $D_T = 4$. Nodes with different label l_u are colored differently. When $D_T = 4$, distance information of node v_i with $Dist(u,v_i) \leq 3$ is directly obtained by $l_u(v_i)$, distance information of node v_i with $Dist(u,v_i) = 3 + d'$ is obtained by subtree with d' depth.

1. $Dist(u,v_i) < D_T$**:** Distance information are directly encoded in node labels through bounding distance node labeling. As the toy example shown in Fig. 1 ($D_T = 4$), $Dist(u,v_i)$ of the neighbor v_i within 3 hops can be obtained by color depth of node. A deeper color implies to be more closer to node u.
2. $Dist(u,v_i) = D_T$**:** Requires 1-depth subtree $T^1(v_i)$, where at least one neighbor v_j satisfies $l_u(v_j) = D_T - 1$. As the toy example shown in Fig. 1 ($D_T = 4$), for node v_1, $Dist(u,v_1)$ can not be derived through the color depth. But the first-layer neighborhood in subtree contains a node v_4 with $l_u(v_4) = 3$ (visually represented as the lightest hue in our scheme), indicates $Dist(u,v_1) = D_T$.
3. $Dist(u,v_i) > D_T$**:** Demands $(d'+1)$-depth subtree $T^{d'+1}(v_i)$ ($d' = Dist(u,v_i) - D_T$), wherein the minimum distance to node u is $Dist(u,v_4) = D_T - 1$. As the toy example shown in Fig. 1 ($D_T = 4$), for node v_2, and v_3, a node with $l_u = 3$ first emerges in the second and third depth of corresponding subtree, respectively, providing conclusive evidence for $Dist(u,v_2) = 5$ and $Dist(u,v_3) = 6$.

That is to say, under the most challenging setting where $Dist(u,v_i) = 3h+1$, a subtree $T^{3h+2-D_T}(v_i)$ is required to yield the distance representation. This implies that the distance to node u for all nodes in the enclosing subgraphs can be obtained by at most a $(3h+2-D_T)$-layer 1-WL-GNN. □

Proof of Theorem 4

Theorem 4. *For a given node pair (u,v), if the PageRank and SimRank heuristics are approximated from $\mathcal{G}^h_{uv}$, then the approximation error is bounded by $(1-c_{PR})^{2h+2}$ and $\frac{(1-c_{SR})^{2h+2}}{c_{SR}}$ respectively, where $c_{PR}, c_{SR} \in (0,1)$.*

Proof. We denote approximation of PageRank in the enclosing subgraph $\mathcal{G}^h_{uv}$ as $\hat{S}^{PR}(u,v)$, and the approximation error is e_{PR}. Rooted PageRank (Brin and Page 2012) iteratively updates a stationary distribution vector for the node $v_i \in \mathcal{V}$. In this paper, we consider the PageRank link score computation via the inverse P-distance theory (Jeh and Widom 2003):

$$S^{PR}(u,v) = \sum_{l=1}^{\infty} \sum_{t \in \mathcal{W}(u,v) \text{ and } |t|=l} P[t] c_{PR}(1-c_{PR})^l. \tag{1}$$

Here a walk t is a node sequence $t = [w_0, w_1, \cdots, w_l]$ which satisfies $w_k \in \Gamma_{\text{out}}(w_{k-1})\,(k = 1, 2, \cdots, l)$ where $w_{k-1}, w_k \in \mathcal{V}$. $|t| = l$ is the length of the walk. $\mathcal{W}(u,v)$ is a set of all walks start from node u and end to node v, $c_{PR} \in (0,1)$ is a hyper-parameter, and the probability of walk t is $P[t] = \prod_{k=0}^{|t|} \frac{1}{\Gamma_{\text{out}}(w_k)}$.

Notice that the probability that a random walker starting at u and ends at arbitrary node $v_i \in \mathcal{V}$ with exactly l steps is $\sum_{v_i \in \mathcal{V}} \sum_{t \in \mathcal{W}(u,v_i) \text{ and } |t|=l} P[t] = 1$. Thus we can get $\sum_{t \in \mathcal{W}(u,v) \text{ and } |t|=l} P[t] \leq \sum_{v_i \in \mathcal{V}} \sum_{t \in \mathcal{W}(u,v_i) \text{ and } |t|=l} P[t] = 1$. In addition, based on Theorem 2, we have

$$\begin{aligned} e_{PR} =& |S^{PR}(u,v) - \hat{S}^{PR}(u,v)| \\ =& \sum_{l=2h+2}^{\infty} \sum_{t \in \mathcal{W}(u,v) \text{ and } |t|=l} P[t] c_{PR}(1-c_{PR})^l \\ \leq& \sum_{l=2h+2}^{\infty} c_{PR}(1-c_{PR})^l \\ =& c_{PR} \times \frac{(1-c_{PR})^{2h+2}}{1-(1-c_{PR})} \\ =& (1-c_{PR})^{2h+2}. \end{aligned} \tag{2}$$

This completes the proof of PageRank.

We denote approximation of SimRank in the enclosing subgraph $\mathcal{G}_{uv}^h$ is $\hat{S}^{SR}(u,v)$, and the approximation error is e_{SR}. SimRank score (Jeh and Widom 2002), which is also defined in a iterative form, gives higher link prediction scores for two nodes whose neighbors are similar. However, in this paper, we consider SimRank computed as follows:

$$S^{SR}(u,v) = \sum_{l=1}^{\infty} \sum_{t^s \in \mathcal{W}^s(u,v) \text{ and } |t^s|=l} P[t^s](1-c_{SR})^l. \tag{3}$$

Here, a simultaneous walk is a combination of two walks $t^s = [(w_0, w_0'), (w_1, w_1'), \cdots, (w_l, w_l)]$ such that one walk starts at w_0, the other walk tarts at w_0', and they first meet at any node w_l. $\mathcal{W}^s(u,v)$ is a set of all simultaneous walks start from node u and v and first meet at any node, $t^s \in (0,1)$ is a hyper-parameter, and the probability of this simultaneous walk is $P[t^s] = \prod_{k=0}^{|t^s|} \frac{1}{\Gamma_{\text{out}}(w_k)\Gamma_{\text{out}}(w_k')}$.

Notice that $\sum_{t^s \in \mathcal{W}^s(u,v) \text{ and } |t^s|=l} P[t^s] \leq 1$. In addition, based on Theorem 2, we have

$$\begin{aligned} e_{SR} =& |S^{SR}(u,v) - \hat{S}^{SR}(u,v)| \\ =& \sum_{l=2h+2}^{\infty} \sum_{t^s \in \mathcal{W}^s(u,v) \text{ and } |t^s|=l} P[t^s](1-c_{SR})^l \\ \leq& \sum_{l=2h+2}^{\infty} (1-c_{SR})^l \\ =& \frac{(1-c_{SR})^{2h+2}}{1-(1-c_{SR})} \\ =& \frac{(1-c_{SR})^{2h+2}}{c_{SR}}. \end{aligned} \tag{4}$$

This completes the proof of SimRank. □

Proof of Theorem 5

Theorem 5. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a scale-free network satisfying:*

1. ***Power-law degree distribution:*** *The graph obeys a power-law degree distribution $P_k \propto k^{-\gamma}$ with exponent $\gamma \in (2,3)$.*
2. ***Degree constraint:*** *The maximum degree $k_{\max}$ satisfies $k_{\max} = O(|\mathcal{V}|^{\frac{1-\epsilon}{6h+2-2h\gamma}})$ for any constant $\epsilon > 0$.*

For a uniformly sampled node pair (u,v) with h-hop enclosing subgraph $\mathcal{G}_{uv}^h$, DRNL and DE are equivalent to bounding distance node labeling of $D_T \geq h+1$ with probability exceeding $O\left(\exp(-|\mathcal{V}|^{-\epsilon})\right)$.

Proof. According to the second property in Theorem2, in disconnected subgraphs, the shortest path distance $3h+1 \geq Dist(u, v_i) \geq h+1$ and $3h+1 \geq Dist(u, v_i) \geq h+1$ is absolutely absent. Consequently, bounding distance node labeling with $D_T \geq h+1$ will produce identical $l_u(v_i)$ and $l_v(v_i)$ for arbitrary node $v_i \in \mathcal{V}_{uv}^h$. This equivalence arises because the disconnected enclosing subgraph $\mathcal{G}_{uv}^h$ directly corresponds to $D(u,v) > 2h+1$ in the original network $\mathcal{G}$. Consequently, the success probability that bounding distance node labeling with $D_T = 3h+2$ (i.e. DRNL and DE) is equivalent to bounding distance node labeling with $D_T = h+1$, is lower-bounded by $\text{Prob}\left(Dist(u,v) > 2h+1\right)$.

We denote the maximum degree as $k_{\max}$ and the minimum degree is $k_{\min}$, and the degree of node v_i as k_u and k_v, respectively. Let $\langle k^m \rangle = \frac{1}{|\mathcal{V}|} \sum_{u \in \mathcal{V}} k_u^m$. The degree distribution of the BA scale-free network can be described as $P = ck^{-\gamma}$. In large networks, the degree distribution can be well approximated by a positive continuous variable k. In this approximation, we have

$$1 = \int_1^{k_{\max}} ck^{-\gamma} dk = \frac{c(k_{\max}^{1-\gamma} - k_{\min}^{1-\gamma})}{1-\gamma}. \tag{5}$$

Thus, we can get

$$c = \frac{1-\gamma}{k_{\max}^{1-\gamma} - k_{\min}^{1-\gamma}}. \tag{6}$$

Hence, there is

$$\begin{aligned}\langle k^m \rangle &= \int_{k_{\min}}^{k_{\max}} c k^{m-\gamma} dk \\ &= \int_{k_{\min}}^{k_{\max}} c k^{m-\gamma} dk \\ &= \frac{(1-\gamma)(k_{\max}^{m+1-\gamma} - k_{\min}^{m+1-\gamma})}{(m+1-\gamma)(k_{\max}^{1-\gamma} - k_{\min}^{1-\gamma})}.\end{aligned} \tag{7}$$

Substituting Eq. (7), we can get

$$\frac{\langle k^2 \rangle^{2h}}{\langle k \rangle^{2h+1}} = O(|\mathcal{V}|^{\frac{(6h-2rh)(1-\epsilon)}{6h-2rh+2}}) \tag{8}$$

The probability of at least one walk of length x between node u and v expresses also the probability of $Dist(u,v) \leq x$. Based on (Fronczak, Fronczak, and Hołyst 2004), the probability of $Dist(u,v) \leq x$ is

$$\text{Prob}\left(Dist(u,v) \leq x\right) = 1 - \exp\left(-\frac{k_u k_v \langle k^2 \rangle^{x-1}}{|\mathcal{V}| \langle k \rangle^x}\right) \tag{9}$$

Consequently, given a node pair (u,v), the probability of $Dist(u,v) > 2h+1$ is

$$\begin{aligned}&\text{Prob}\left(Dist(u,v) > 2h+1\right) \\ =&1 - \text{Prob}\left(Dist(u,v) \leq 2h+1\right) \\ =&\exp\left(-\frac{k_u k_v}{|\mathcal{V}|} \frac{\langle k^2 \rangle^{2h}}{\langle k \rangle^{2h+1}}\right) \\ \geq&\exp\left(-\frac{k_{\max}^2}{|\mathcal{V}|} \frac{\langle k^2 \rangle^{2h}}{\langle k \rangle^{2h+1}}\right) \\ =&\exp\left(-\frac{O(|\mathcal{V}|^{\frac{2(1-\epsilon)}{6h-2rh+2}})}{|\mathcal{V}|} O(|\mathcal{V}|^{\frac{(6-2rh)(1-\epsilon)}{6h-2rh+2}})\right) \\ =&O\left(\exp(-|\mathcal{V}|^{-\epsilon})\right)\end{aligned} \tag{10}$$

This completes the proof. □

Proof of Theorem 6

Theorem 6. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a scale-free network satisfying:*

1. ***Power-law degree distribution:*** *The graph obeys a power-law degree distribution $P_k \propto k^{-\gamma}$ with exponent $\gamma \in (2,3)$.*
2. ***Degree constraint:*** *The maximum degree $k_{\max}$ satisfies $k_{\max} = O(|\mathcal{V}|^{\frac{1+\epsilon}{(3-\gamma)(2h+1)}})$ for any constant $\epsilon > 0$.*

Consider a uniformly sampled node pair (u,v) and the corresponding h-hop enclosing subgraph $\mathcal{G}_{uv}^h$. If $\mathcal{G}_{uv}^h$ is connected, then the bounding distance node labeling with $D_T = h+1$, together with a L-layer 1-WL-GNN ($L > 1$), almost surely yields a distance representation for all nodes in $\mathcal{G}_{uv}^h$.

Proof. First, according to Theorem 3, the distance representations for all nodes are guaranteed to be generated when $L \geq 2h+1$.

We next analyze the critical regime $2h+1 > L > 1$, considering two distinct cases:

1. $D(u,v) \leq L$**:** Applying the triangle inequality yields:
$$D(u, v_i) \leq D(u,v) + D(v, v_i) \leq L + h, \tag{11}$$
and
$$D(v, v_i) \leq D(u,v) + D(u, v_i) \leq L + h. \tag{12}$$
Under this condition, the combination of the bounding distance node labeling with $D_T = h+1$, and a L-layer 1-WL-GNN, provably generates distance representations for all nodes in the subgraph.
2. $D(u,v) > L$**:** Here, the upper bounds of $D(u, v_i)$ and $D(v, v_i)$ exceed $L+h$, potentially causing failure in distance representation generation.

Consequently, the success probability for generating distance representations for all nodes in subgraph $\mathcal{G}_{uv}^h$ is lower-bounded by $\text{Prob}(Dist(u,v) \leq L)$. Given that the enclosing subgraph is connected is equivalent to $\text{Prob}(Dist(u,v) \leq 2h+1)$, we derive the conditional probability:

$$\begin{aligned}&\text{Prob}\left(Dist(u,v) \leq L | Dist(u,v) \leq 2h+1\right) \\ =&\frac{\text{Prob}\left(Dist(u,v) \leq L \text{ and } Dist(u,v) \leq 2h+1\right)}{\text{Prob}\left(Dist(u,v) \leq 2h+1\right)} \\ =&\frac{\text{Prob}\left(Dist(u,v) \leq L\right)}{\text{Prob}\left(Dist(u,v) \leq 2h+1\right)}.\end{aligned} \tag{13}$$

Putting Eq. (9) into Eq. (14), we have

$$\begin{aligned}&\text{Prob}\left(Dist(u,v) \leq L | Dist(u,v) \leq 2h+1\right) \\ =&\frac{1 - \exp(-\frac{k_u k_v}{|\mathcal{V}|} \frac{\langle k^2 \rangle^{L-1}}{\langle k \rangle^L})}{1 - \exp(-\frac{k_u k_v}{|\mathcal{V}|} \frac{\langle k^2 \rangle^{2h}}{\langle k \rangle^{2h+1}})} \\ =&\frac{1 - \exp(-k_u k_v O(|\mathcal{V}|^{\frac{(1+\epsilon)(L-1)}{2h+1}-1}))}{1 - \exp(-k_u k_v O(|\mathcal{V}|^{\epsilon}))}.\end{aligned} \tag{14}$$

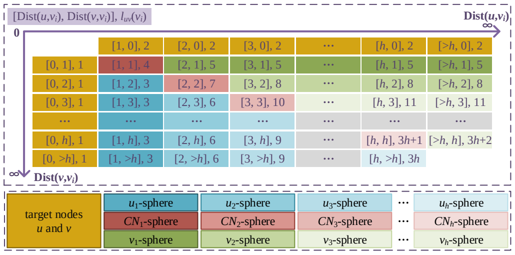
It is easy to find that $\text{Prob}\left(Dist(u,v) \leq L | Dist(u,v) \leq 2h+1\right) \rightarrow 1$ when $|\mathcal{V}| \rightarrow \infty$.

This complete the proof. □

Configuration details of BS-SubGNN

The framework of the proposed BS-SubGNN consists of four main components, where the function of each component is briefly outlined as follows.

1. **Bicentric Sphere Node Labeling:** Given an enclosing subgraph of a target pair of nodes, the bicentric sphere node labeling block generates a label for each node in the subgraph according to the bicentric sphere the node belongs to. The node label is then utilized to generate the initial attribute vector of the node.
2. **Message Passing:** Given the above subgraph, the message passing block performs node aggregation operations to generate embeddings for each node within the subgraph.


[Dist(u,v_i), Dist(v,v_i)], $l_{uv}(v_i)$
0
Dist(u,v_i)
∞
Dist(v,v_i)
∞

	[1, 0], 2	[2, 0], 2	[3, 0], 2	···	[h, 0], 2	[>h, 0], 2
[0, 1], 1	[1, 1], 4	[2, 1], 5	[3, 1], 5	···	[h, 1], 5	[>h, 1], 5
[0, 2], 1	[1, 2], 3	[2, 2], 7	[3, 2], 8	···	[h, 2], 8	[>h, 2], 8
[0, 3], 1	[1, 3], 3	[2, 3], 6	[3, 3], 10	···	[h, 3], 11	[>h, 3], 11
···	···	···	···	···	···	···
[0, h], 1	[1, h], 3	[2, h], 6	[3, h], 9	···	[h, h], 3h+1	[>h, h], 3h+2
[0, >h], 1	[1, >h], 3	[2, >h], 6	[3, >h], 9	···	[h, >h], 3h	

target nodes u and v	u_1-sphere	u_2-sphere	u_3-sphere	···	u_h-sphere
	CN_1-sphere	CN_2-sphere	CN_3-sphere	···	CN_h-sphere
	v_1-sphere	v_2-sphere	v_3-sphere	···	v_h-sphere

Figure 2: Illustration of Bicentric Sphere Node Labeling process. In the notion $[d_v, d_u], l_i$ at each cell of the above table, d_v and d_u refer to the distance $Dist(u, v_i)$ and $Dist(v, v_i)$, and l_i is the resulted node label.

3. **Bicentric Sphere Subgraph Pooling:** The bicentric sphere subgraph pooling block is to map the enclosing subgraph with a dozen of node embeddings into a single vector, which will be then fed into multi-layer perceptrons (MLPs) to learn the link score.
4. **MLP Classifier:** In the last, the MLP classifier, taking the subgraph embeddings obtained from the above pooling procedure as input, outputs the binary classification score of the subgraph for edge-oriented tasks such as link prediction, link direction prediction, and link sign prediction.

Bicentric Sphere Node Labeling

Bicentric Sphere Node Labeling is the first step of BS-SubGNN, which incorporate neighborhood extraction, subgraph extraction and node labeling process. Given a target node pair $u,\ v \in \mathcal{V}$ of the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, the h-hop subgraph $\mathcal{G}^h_{uv}$ is taken as the main structural pattern to determine the relation between u and v. The Bicentric Sphere Node Labeling (BSNL) aims to assign a label to each node in the subgraph. In detail, the process proceeds are follows. Given the subgraph $\mathcal{G}^h_{uv}$, we assign the center node u and v with label 1 and 2, respectively. For each node in the u_q-sphere, its label is assigned to $3q$. For each node in the CN_q-sphere, its label is assigned to $3q + 1$. Finally, for each node in v_q-sphere, its label is assigned to $3q + 2$. Here, $q = 1, 2, \ldots, h$. Fig. 2 illustrates the label of each node v_i in the subgraph based on its distances with the two target nodes, i.e., $Dist(u, v_i)$ and $Dist(v, v_i)$. Fig. 3 presents an illustrative example of the node label in the enclosing subgraph of node pair (u, v). We further turn the label into a $(3q + 2)$-dimensional one-hot node features. Note that the above node features can be directly concatenated with the node attribute in $\mathbf{A}$ if the latter available.

Considering the complexity of computing the distances $Dist(u, v_i)$ and $Dist(v, v_i)$, in the following, we utilize the breadth-first-search (BFS) method to generate the above subgraph and node label in the meantime. The process contains two main parts. The first part obtains the h-hop neighborhood $\Gamma^h(u)$, $\Gamma^h(v)$ of the target node $u,\ v \in \mathcal{V}$, respectively. Taking node u for example, $\Gamma^0(u)$ and $\Gamma^1(u)$ can be obtained directly. Then, $\Gamma^q(u)$ with $2 \leq q \leq h$ can be successively obtained by

$$\Gamma^q(u) = \cup_{v_i \in \Gamma^{q-1}(u)} \Gamma^1(v_i) - \Gamma^{q-1}(u) - \Gamma^{q-2}(u). \quad (15)$$

The above equation holds since the distance from node u to nodes in the 1-hop neighborhood of nodes in $\Gamma^{q-1}(u)$ can only be $q - 2$, $q - 1$, or q. By eliminating those nodes in $\Gamma^{q-1}(u)$ and $\Gamma^{q-2}(u)$, the q-hop neighborhood can be obtained. Since 1-hop neighborhood can be easily obtained by the BFS method, neighborhood extraction can be realized through a successive iteration process. Algorithm 1 summarizes the above neighborhood extraction process.

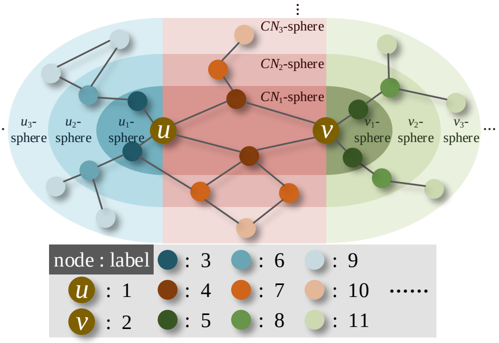


Figure 3: A toy example of the node label in the enclosing subgraph.

Algorithm 1: Node neighborhood extraction algorithm

```
Input: Graph $\mathcal{G}$, given node $u$, number of hops $h$
Output: Neighborhoods $\Gamma^q(u)$ ($q = 0, 1, 2, \ldots, h$) of node $u$
for $q$ in $range(2, h)$
    $\Gamma^q(u) = \{\}$
    for $v_i$ in $\Gamma^{q-1}(u)$
        $\Gamma^q(u) = \Gamma^q(u) \cup \Gamma^1(v_i)$
    end for
    $\Gamma^q(u) = \Gamma^q(u) - \Gamma^{q-1}(u) - \Gamma^{q-2}(u)$
end for
return $\Gamma^q(u)$ ($q = 0, 1, 2, \ldots, h$)
```

Given the h-hop neighborhood $\Gamma^h(u)$, $\Gamma^h(v)$ of the target nodes $u,\ v \in \mathcal{V}$, the second part of the node labeling process is to generate the enclosing subgraph and the corresponding node label. Algorithm 2 summarizes the above subgraph extraction and labeling process.

Our proposed bicentric sphere node labeling method avoids computation of the distances $Dist(u, v_i)$ and $Dist(v, v_i)$, which is commonly involved in existing node label methods such as Double Radius Node Labeling (DRNL) (Zhang and Chen 2018) and Distance Encoding (DE) (Li et al. 2020). Thus, a reduced time complexity can be expected. Further experimental comparisons of the time complexity of different node labeling methods are shown in Table 3 in Section .

Algorithm 2: Subgraph extraction and labeling algorithm

```
Input: neighborhood of node $u$: $\Gamma^q(u)\ (q = 0, \ldots, h)$,
    neighborhood of node $v$: $\Gamma^q(v)\ (q = 0, \ldots, h)$, number
    of hops $h$
Output: A enclosing subgraph $\mathcal{G}_{uv}^h$ and a dictionary $l_{uv}$
    recording bicentric sphere labels of each node $v_i \in \mathcal{V}_{uv}^h$
Let $\mathcal{V}_{uv}^h = \bigcup_{q=0}^h \Gamma^h(u) \cup \bigcup_{q=0}^h \Gamma^h(v)$
for $v_i$ in $\mathcal{G}_{uv}^h$
    $l_u(v_i) = h + 1$
    $l_v(v_i) = h + 1$
end for
for $q$ in range$(0, h)$
    for $v_i$ in $\Gamma^q(u)$
        $l_u(v_i) = q$
    end for
    for $v_i$ in $\Gamma^q(v)$
        $l_v(v_i) = q$
    end for
end for
for $v_i$ in $\mathcal{V}_{uv}^h$
    Calculate $l_{uv}(v_i)$ by Eq. (5)
end for
return $\mathcal{G}_{uv}^h = \mathcal{G}(\mathcal{V}_{uv}^h), l_{uv}$
```

Message Passing

Message passing is the second step of BS-SubGNN, which is to take the h-hop subgraph $\mathcal{G}_{uv}^h$ as the input to achieve the corresponding node embedding.

To encounter diverse types of networks, we distinguish different types of networks by definition of relations, including directions and heterogeneous links. In detail, $\mathcal{R} = \{r_k\}_{k=1}^{|\mathcal{R}|}$ is the set of relation types. For each edge $e \in \mathcal{E}$, it can be represented by (v_i, r, v_j) with $v_i, v_j \in \mathcal{V}$ are the head and tail nodes respectively, and $r \in \mathcal{R}$ is the relation type. With this unified network definition, in this paper, we consider the following four types of networks.

i) Undirected graphs: There is only one relation type ($|\mathcal{R}| = 1$). In this case, the network definition can be simplified as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $(v_i, v_j) \in \mathcal{E}$ representing the link. Here, the relation type r is omitted by default.

ii) Undirected attribute graphs: Compared with undirected graphs, $\mathbf{A}$ contains additional attribute information.

iii) Directed graphs: There are two relation types ($|\mathcal{R}| = 2$). In this case, the network definition can be modeled by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ with $\mathcal{R} = (r_1, r_2)$. Here, $r_1 : v_i \rightarrow v_j$ and $r_2 : v_i \leftarrow v_j$ represent out-direction and in-direction, respectively. Note that following the definition of directed graphs, there is $(v_i, r_1, v_j) = (v_j, r_2, v_i)$.

v) Signed directed graphs: There are four link types ($|\mathcal{R}| = 4$). In this case, the network definition is given by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ with $\mathcal{R} = (r_1, r_2, r_3, r_4)$. Here, $r_1 : v_i \rightarrow^+ v_j$, $r_2 : v_i \leftarrow^+ v_j$, $r_3 : v_i \rightarrow^- v_j$, $r_4 : v_i \leftarrow^- v_j$ represent positive out-direction, positive in-direction, negative out-direction, and negative in-direction, respectively. Similarly, there is $(v_i, r_1, v_j) = (v_j, r_2, v_i)$ and $(v_i, r_3, v_j) = (v_j, r_4, v_i)$.

For convenience, we write $\Gamma^1(v_i)$ alternatively by $\Gamma(v_i)$. Given that the relation types $|\mathcal{R}|$ in the graph is larger than 1, the 1-hop neighborhood of each node $v_i \in \mathcal{V}$ can be divided by $\Gamma(v_i) = \cup_{r_k \in \mathcal{R}} \Gamma_{r_k}(v_i)$, where $\Gamma_{r_k}(v_i) = \{v_j | (v_i, r_k, v_j) \in \mathcal{E}\}$.

In this paper, we perform an L-layer message passing process as follows. Given any node $v_i \in \mathcal{V}_{uv}^h$ in the subgraph $\mathcal{G}_{uv}^h$, the l-layer message passing is given by

$$X_{r_k}^l(v_i) = \text{Agg}\left(\{z^{l-1}(v_j), \forall v_j \in \Gamma_{r_k}^1(v_i)\}\right). \quad (16)$$

Here, $l = 1, 2, \cdots, L$, $r_k \in \mathcal{R}$ is the relation type, $z^{l-1}(v_j)$ is output of $(l-1)$-th message passing layer of node v_j, $z^0(v_j)$ is the initial node attribute of node v_j obtained from Bicentric Sphere Node Labeling process, and $X_{r_i}^l(u) \in \mathbb{R}^{d_{hid}}$. By implementing different types of aggregating operation $\text{Agg}(\cdot)$, the above process could cover many well-known message passing techniques, including sum aggregator, mean aggregator, GCN (Kipf and Welling 2016a), GraphSAGE (Hamilton, Ying, and Leskovec 2017), GAT (Velickovic et al. 2017) and many others.

Subsequently, we concatenate aggregated embeddings obtained from different relation types as follows.

$$X^l(v_i) = \left[X_{r_1}^l(v_i) || X_{r_2}^l(v_i) || \cdots || X_{r_{|\mathcal{R}|}}^l(v_i)\right], \quad (17)$$

where the notion $||$ means concatenation operation. In the last, a linear transform and a $\tanh$ active function are conducted on $X^l(v_i)$ to obtain l-th layer's node embedding

$$z^l(v_i) = \tanh(W^l X^l(v_i)), \quad (18)$$

where $W^l \in \mathbb{R}^{d_{hid} \times |\mathcal{R}| d_{hid}}$ is a trainable parameter.

After conducting message-passing for L layers, the final representation for each node $v_i \in \mathcal{V}_{uv}^h$ is obtained by concatenating node embeddings at different layers as follows:

$$z(v_i) = [z^1(v_i) || z^2(v_i) || \cdots || z^L(v_i)]. \quad (19)$$

Bicentric Sphere Subgraph Pooling

Bicentric Sphere Subgraph Pooling is the third step of BS-SubGNN, which takes the node embeddings $\{z(v_i), \forall v_i \in \mathcal{V}_{uv}^h\}$ in the subgraph $\mathcal{G}_{uv}^h$ as input and outputs a fix-sized subgraph embedding vector $g(u, v)$. In this paper, an embedding of each sphere is first obtained based on the identified sphere in node labeling process, and then all sphere embeddings are concatenated as subgraph embeddings. A crucial observation is that one-hop subgraph structures hold significant importance for profiling link formation. Accordingly, we omit q-hop ($q \geq 2$) subgraph in bicentric sphere pooling process. In the following, we first briefly introduce existing subgraph pooling processes. Then we propose our bicentric sphere subgraph pooling process and highlight how it differs with existing pooling techniques.

Typically, existing subgraph encoding methods perform pooling over the entire subgraph to generate its embeddings as follows

$$g(u, v) = \text{Pool}(\{z(v_i), \forall v_i \in \mathcal{V}_{uv}^h\}), \quad (20)$$

Table 1: Statistical information of each attribute network. Nodes, Edges, Node Features refer to the number of nodes, number of edges and node attribute dimension, respectively.

Dataset	Nodes	Edges	Node Features
CORA(attr)	2708	10556	1433
Citeseer	4230	10674	602
Twitch_en	7126	77774	128
CoauthorCS	18333	163788	6805

where $\mathrm{Pool}(\cdot)$ represents graph pooling techniques, including sum pool, mean pool, sort pool (Zhang et al. 2018) and many others. In particular, a widely-used subgraph pooling method, named edge pool, concatenates embeddings of center nodes to obtain subgraph embedding by

$$g(u, v) = [z(u)||z(v)]. \tag{21}$$

In our proposed BS-SubGNN, the subgraph embedding is obtained by

$$g(u, v) = [S_u||S_{CN}||S_v], \tag{22}$$

where S_u is u-sphere embedding, which is computed by

$$S_u = \mathrm{Pool}(\{z(v_i), \forall v_i \in \mathcal{V}^1_{uv,u}\}). \tag{23}$$

Similarly, CN-sphere embedding S_{CN} and v-sphere embedding S_v are computed respectively by

$$S_{CN} = \mathrm{Pool}(\{z(v_i), \forall v_i \in \mathcal{V}^1_{uv,CN}\}), \tag{24}$$

$$S_v = \mathrm{Pool}(\{z(v_i), \forall v_i \in \mathcal{V}^1_{uv,v}\}). \tag{25}$$

The detailed implementation of $\mathrm{Pool}(\cdot)$ in our work is in Eqs. (8), (10) and (11).

In this paper, we adopt the attention mechanism to conduct above $\mathrm{Pool}(\cdot)$ operation. Notice that we do not utilize normalization, such as mean and softmax operation in attention score computation. While it is a widely used technique and is proved that sum operator showcases more expressive power than normalization operators.

Compared with existing subgraph pooling methods, the above bicentric sphere subgraph pooling technique differs in terms of both pooling level and pooling technique. On one hand, bicentric sphere subgraph pooling only utilizes the node embeddings in the 1-hop sphere of the target node pair. While existing enclosing subgraph pooling utilizes the node embeddings of the entire subgraph or only use the node embedding of the target node pair. On the other hand, we utilize the attention mechanism as pooling technique, rather than widely-used mean pool, sum pool, sort pool (Zhang et al. 2018; Zhang and Chen 2018), and many others. A further comparison of different pooling levels and pooling techniques will be illustrated in Appendix and the experimental results of the corresponding ablation study are shown in Table 7.

Performance Evaluation in Diverse Networks

We empirically evaluate the performances of our link prediction framework BS-SubGNN on several different network types, including undirected graphs, attribute graphs, directed graphs, and signed directed graphs. For each graph type, the commonly-used datasets together with the benchmarking methods are adopted for fair comparison. Implementations are done using the PyTorch Pytorch 1.8.1 framework in Python 3.7, on an Ubuntu 18.04 server equipped with 1 NVIDIA RTX A4000 GPU and an Intel(R) Xeon(R) E5-2686 v4 CPU.

Attribute Networks

Datasets and Evaluation Metrics Following the experimental settings in (Li et al. 2022), we consider 4 research datasets of attribute networks, including CORA(attr) (Huang et al. 2021b), CiteSeer (Yang, Cohen, and Salakhudinov 2016), Twitch_en (Rozemberczki, Allen, and Sarkar 2021), CoauthorCS (Shchur et al. 2018). Table 1 provides basic statistical information about each attribute network. 90% of existing links are sampled as positive training samples with random seed 2. The area under curve (AUC) and average precision (AP) are employed to assess the performance of link prediction algorithms. All the experiments in attribute graphs are conducted 10 times and the average value is recorded for evaluation.

Baselines The following typical baseline methods for link prediction in attribute networks are adopted as benchmarks to compare the performance of the proposed BS-SubGNN in attribute networks. Similarly, these methods can be categorized into 1) Heuristics methods, such as common neighbors, Adamic-Adar (AA), and Personalized PageRank (PPR) (Wang et al. 2020), 2) Graph embedding methods, including node2vec (N2V) (Grover and Leskovec 2016), GCN (Kipf and Welling 2016a), GAT (Velickovic et al. 2017), and NeoGNN (Yun et al. 2021), and 3) Subgraph encoding methods, such as SEAL (Zhang and Chen 2018), LGLP (Cai et al. 2022) and BSAL (Li et al. 2022).

Prediction Performance The hyperparameters for BS-SubGNN in attribute networks are as follows. The hidden dimension d_{hid} is 32, the batch size is 256, the learning rate is 0.001, and the message passing layer L is set to 3 for all networks. The MLP module contains 3 hidden layers of 32, 32, 16 neurons, respectively, followed by a softmax output layer. The number of epochs is set to be 10, and the number of hop h for subgraph generation is 1. For the message passing function $\mathrm{Agg}(\cdot)$, we follow the same graph convolution (Kipf and Welling 2016a) as in SEAL (Zhang and Chen 2018) and LGLP (Cai et al. 2022) for fair comparisons. The prediction results of our BS-SubGNN are presented in Table 2, with benchmark results sourced from (Li et al. 2022).

Several observations of link prediction in attribute networks can be derived from Table 2. First, network embedding methods outperform heuristic methods. It indicates that learning-based methods could generate more adaptable edge features for diverse real-world networks. Second, subgraph encoding methods surpass network embedding methods,

Table 2: AUC and AP comparisons with other attribute link prediction algorithms. The best results and the second-best results for each dataset are in bold and underlined, respectively. Hop refers to the number of hops used to generate features of node pair.

Model	Hop	Citeseer		CORA(attr)		Twitch_en		CoauthorCS	
		AUC	AP	AUC	AP	AUC	AP	AUC	AP
CN	$h = 1$	63.23	63.18	71.66	71.33	76.29	75.99	89.56	89.51
AA	$h = 1$	63.25	63.30	71.68	71.55	76.57	77.17	89.59	89.63
PPR	$h > 1$	71.63	74.79	82.86	88.12	84.14	87.40	95.55	97.12
N2V	$h > 1$	74.62±1.00	82.05±0.57	86.02±0.68	89.01±0.40	86.43±0.12	89.81±0.10	96.26±0.05	96.92±0.03
GCN	$h > 1$	90.86±0.45	91.47±0.43	88.52±1.06	88.37±1.37	86.20±0.24	88.61±0.80	94.73±0.24	94.56±0.24
GAT	$h > 1$	89.60±0.52	89.31±0.67	89.25±0.77	89.17±0.87	82.30±0.26	83.64±0.47	95.28±0.08	94.91±0.14
NeoGNN	$h > 1$	91.22±0.64	91.35±0.66	88.33±0.78	89.58±0.62	90.08±0.67	92.97±0.24	97.90±0.19	98.23±0.07
SEAL	$h \leq 2$	89.86±0.77	93.19±0.08	90.86±0.33	91.72±0.32	92.15±0.18	92.43±0.03	96.64±0.27	97.28±0.22
LGLP	$h = 2$	92.52±0.07	92.86±0.55	90.96±0.86	91.58±0.39	91.10±0.05	91.71±0.68	96.06±0.00	97.13±0.01
BSAL	$h = 1$	93.24±0.05	94.11±0.07	91.11±0.10	92.21±0.10	92.81±0.05	93.31±0.05	96.79±0.06	97.61±0.06
BS-SubGNN	$h = 1$	**94.32±0.18**	**95.03±0.11**	**91.98±0.09**	**93.11±0.12**	**94.34±0.06**	**94.71±0.05**	**98.39±0.05**	**98.63±0.06**

Table 3: Statistical information of each real-world directed network. Nodes, Edges, Avg. degree refer to the number of nodes, number of edges and average degree, respectively.

Dataset	Nodes	Edges	Avg. degree
Cora	23166	91500	7.90
Epinions	75879	508837	13.41
Twitter	465017	834797	3.59

demonstrating that characters for link prediction are well defined in the enclosing subgraph. Finally, our BS-SubGNN achieves the state-of-the-art performance in all four attribute networks even when only one-hop information is utilized. This demonstrates the effectiveness of our proposed bicentric sphere node labeling and pooling methods.

Directed Networks

Datasets and Baselines Following the experimental settings in (Ke et al. 2024), in this paper we consider 3 real-world graphs, including Cora, Epinions and Twitter, as the experimental datasets of directed networks. We randomly split the datasets into training, validation, and testing subsets, maintaining a ratio of 16:1:3. Three tasks are considered. Existence Prediction Task predicts the existence of links. Direction Prediction Task predicts the orientation of unidirectional links. Four-type Classification Task predicts positive, reverse, bidirectional, or non-existent links. The area under curve (AUC) is adopted as metric in both Existence Prediction Task and Direction Prediction Task, and accuracy is adopted as metric in Four-type Classification Task. The prediction performance is averaged over 10 times experiments. We report the average values, along with the standard deviation across the runs.

Baselines The following typical baseline methods for link prediction in directed networks are adopted as benchmarks to compare the performance of the proposed BS-SubGNN in directed networks, including NERD (Khosla et al. 2020), DGGAN (Zhu et al. 2021), MagNet (Zhang et al. 2021b), SigMaNet (Fiorini et al. 2023), DiGAE (Kollias et al. 2022), ODIN (Yoo et al. 2023) and DUPLEX (Ke et al. 2024).

Prediction Performance The hyperparameter settings of BS-SubGNN in directed networks are as follows. The hidden dimension d_{hid} is 32, the batch size is 256, the learning rate is 0.001, and the message passing layer L is 3 for all the networks. The MLP module contains 3 hidden layers of 32, 32, 16 neurons, respectively, and a softmax output layer for prediction. The number of epochs is 10, and the number of hop h for subgraph generation is 1. We adopt sum aggregator as the message passing function $\mathrm{Agg}(\cdot)$.

Several observations of link prediction in directed networks can be derived from Table 4. On one hand, BS-SubGNN consistently surpasses all the embedding methods in Existence Prediction Task and Direction Prediction Task and in all three datasets. BS-SubGNN also attains the state-of-the-art performance in dataset Epinions and Twitter. Especially, BS-SubGNN surpasses all baselines in dataset Epinions by a large margin. This demonstrates the superiority of subgraph encoding with BSNL and BSSP for link prediction task. On the other hand, compared with other methods, only one-hop information is leveraged and the performance gains of our approach are considered encouraging. In real-world applications, when multi-hop neighborhood information is hard to be accessible, our proposed BS-SubGNN can still work and return satisfied performances.

Signed Directed Networks

Datasets and Evaluation Metrics Following the experimental settings (Huang et al. 2021a) and (Fang, Tan, and Wang 2023), we utilize 5 public signed graphs, includ-

Table 4: Comparison with other directed link prediction algorithms. AUC is adopted as metric in both Existence Prediction Task and Direction Prediction Task, and accuracy is adopted as metric in Four-type Classification Task. The best results and the second-best results for each dataset are in bold and underlined, respectively. Hop refers to the number of hops used in each method to generate embeddings of node pair.

Model	Hop	Existence Prediction			Direction Prediction			Four-type Classification		
		Cora	Epinions	Twitter	Cora	Epinions	Twitter	Cora	Epinions	Twitter
NERD	$h > 1$	85.1±0.5	78.6±0.1	94.9±0.0	92.3±0.2	86.9±0.3	95.6±0.1	38.6±0.8	33.1±0.4	32.2±0.4
DGGAN	$h > 1$	89.6±0.2	84.5±2.7	99.1±0.1	95.4±0.2	94.0±2.0	99.1±0.1	10.5±6.4	20.1±0.1	67.4±7.1
MagNet	$h > 1$	89.4±0.1	85.1±0.1	99.1±0.1	95.4±0.2	96.6±0.1	99.9±0.1	63.0±0.3	65.2±0.4	91.6±1.1
SigMaNet	$h > 1$	93.6±0.2	90.3±0.0	99.1±0.0	96.4±0.0	96.7±0.0	99.9±0.0	78.7±0.4	75.4±0.1	97.2±0.0
DiGAE	$h > 1$	83.9±0.5	81.8±0.1	98.8±0.0	89.8±0.3	91.5±0.1	99.9±0.0	34.4±0.3	40.2±0.2	35.9±0.0
ODIN	$h > 1$	90.8±0.2	90.9±0.1	99.2±0.0	96.7±0.2	97.4±0.0	99.8±0.0	70.6±0.4	73.1±0.1	98.0±0.0
DUPLEX	$h > 1$	95.9±0.1	91.0±0.2	99.3±0.2	97.2±0.2	95.2±0.4	99.9±0.0	**88.4±0.4**	76.4±0.2	98.1±0.2
BS-SubGNN	$h = 1$	**96.8±0.1**	**95.2±0.0**	**99.5±0.0**	**98.2±0.1**	**98.6±0.0**	**99.996±0.0**	87.9±0.3	**81.7±0.3**	**99.2±0.0**

Table 5: Some statistical information of each real-world signed directed network. Nodes, Pos edges, Neg edges and Ratio refer to the number of nodes, number of positive links, number of negative links and the ratio of Positive links to Negative links, respectively.

Dataset	Nodes	Pos edges	Neg edges	Ratio
Bitcoin-Alpha	3782	22649	1536	14.75
Bitcoin-OTC	5881	32028	3563	8.99
WikiRfA	11259	138813	39283	3.53
Slashdot	82140	425071	124130	3.42
Epinions(sign)	131827	717667	123704	5.80

ing Bitcoin-Alpha, Bitcoin-OTC, WikiRfA, Slashdot, and Epinions(sign) as the experimental datasets of signed directed networks. Table 5 provides basic statistical information about each signed directed network. We randomly sample 80% positive links and 80% negative links as training samples. Each dataset is randomly split into a training set and a testing set for 5 times. The area under curve (AUC), F1 score, macro-F1 and micro-F1 are utilized as the standard performance metrics.

Baselines The following typical baseline methods for link sign prediction are adopted as benchmarks to compare the performance of the proposed BS-SubGNN in signed directed networks.

1) Feature engineering method: All23 (Leskovec, Huttenlocher, and Kleinberg 2010) predicts the link sign by extracting totally 23-dimensional features from one-hop neighborhood.
2) Unsigned network embedding methods: Three classical unsigned network embedding methods, including DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), node2vec (N2V) (Grover and Leskovec 2016) and LINE (Tang et al. 2015), are taken for comparison.
3) Signed network embedding methods: two classical signed undirected network embedding methods, including SiNE (Wang et al. 2017) and SGCN (Derr, Ma, and Tang 2018), and four classical signed directed network embedding methods, including SIGNet (Islam, Prakash, and Ramakrishnan 2018), BESIDE (Chen et al. 2018), SiGAT (Huang et al. 2019), and SDGNN (Huang et al. 2021a), are taken for comparison.
4) Subgraph encoding methods: SELO (Fang, Tan, and Wang 2023) is the only subgraph encoding method for signed directed networks.

Prediction Performance The hyperparameter settings of BS-SubGNN in signed directed networks are as follows. The hidden dimension d_{hid} is 32, the batch size is 256, the learning rate is 0.001, and the message passing layer L is 3 for all the networks. The MLP module contains 3 hidden layers of 32, 32, 16 neurons, respectively, and a softmax output layer for prediction. The number of hop h for subgraph generation is 1. For the message passing function $\mathrm{Agg}(\cdot)$, an attention mechanism is utilized to aggregate neighbor information. For the setting of epoch number, we set 20 in Bitcoin-Alpha Bitcoin-OTC and WikiRfA, and 10 in Slashdot and Epinions(sign). Table 6 illustrates the prediction results of our BS-SubGNN.

Several observations of link sign prediction in signed directed networks can be derived from Table 6. First, unsigned network embedding methods perform much poorly than other methods. The reason lies in that direction and sign information is omitted. Second, signed directed embedding methods SIGNet, BESIDE, SiGAT and SDGNN outperform signed undirected embedding methods SiNE and SGCN. It confirms that prediction performance is improved when additional direction information is considered for link sign prediction. Third, subgraph encoding method SELO outperforms embedding-based methods by a large margin, exhibiting high expressive power of subgraph encoding framework for link sign prediction. Finally, our BS-SubGNN consistently attains the state-of-the-art performance in all

Table 6: Micro-F1, F1, Macro-F1 and AUC Comparisons with other sign prediction algorithms on five real-world networks. The best results and the second-best results for each dataset using each metric are in bold and underlined, respectively. Hop refers to the number of hops used to generate the enclosing subgraph of node pair.

Dataset	Metrics	All23	Deepwalk	N2V	LINE	SiNE	SIGNet	BESIDE	SGCN	SiGAT	SDGNN	SELO	BS-SubGNN
Bitcoin-Alpha	Micro-F1	0.9486	0.9367	0.9355	0.9352	0.9458	0.9422	0.9489	0.9256	0.9456	0.9491	0.9599	**0.9624**
	F1	0.9730	0.9673	0.9663	0.9664	0.9716	0.9696	0.9732	0.9607	0.9714	0.9729	0.9788	**0.9801**
	Macro-F1	0.7167	0.4848	0.6004	0.5220	0.6869	0.6965	0.7300	0.6367	0.7026	0.7390	0.8107	**0.8141**
	AUC	0.8882	0.6409	0.7576	0.7114	0.8728	0.8908	0.8981	0.8469	0.8872	0.8988	0.9187	**0.9459**
Bitcoin-OTC	Micro-F1	0.9361	0.8937	0.9089	0.8911	0.9095	0.9229	0.9320	0.9078	0.9268	0.9357	0.9553	**0.9574**
	F1	0.9653	0.9434	0.9507	0.9413	0.9510	0.9581	0.9628	0.9491	0.9602	0.9647	0.9754	**0.9767**
	Macro-F1	0.7826	0.5281	0.6793	0.5968	0.6805	0.7386	0.7843	0.7306	0.7533	0.8017	0.8656	**0.8676**
	AUC	0.9121	0.6596	0.7643	0.7248	0.8571	0.8935	0.9152	0.8755	0.9055	0.9124	0.9532	**0.9613**
WikiRfA	Micro-F1	0.8346	0.7837	0.7814	0.7977	0.8338	0.8384	0.8589	0.8489	0.8457	0.8627	0.8644	**0.8759**
	F1	0.8987	0.8779	0.8719	0.8827	0.8972	0.9001	0.9117	0.9069	0.9042	0.9142	0.9155	**0.9218**
	Macro-F1	0.7235	0.4666	0.5626	0.5738	0.7319	0.7384	0.7803	0.7527	0.7535	0.7849	0.7859	**0.8105**
	AUC	0.8604	0.5876	0.6930	0.6772	0.8602	0.8682	0.8981	0.8563	0.8829	0.8898	0.9049	**0.9222**
Slashdot	Micro-F1	0.8472	0.7738	0.7526	0.7489	0.8265	0.8389	0.8590	0.8296	0.8494	0.8616	0.8818	**0.8988**
	F1	0.9070	0.8724	0.8528	0.8525	0.8918	0.8983	0.9105	0.8926	0.9055	0.9128	0.9255	**0.9358**
	Macro-F1	0.7399	0.4384	0.5390	0.5052	0.7273	0.7554	0.7892	0.7403	0.7671	0.7892	0.8198	**0.8482**
	AUC	0.8880	0.5408	0.6709	0.6145	0.8409	0.8752	0.9017	0.8534	0.8874	0.8977	0.9250	**0.9456**
Epinions(sign)	Micro-F1	0.9226	0.8214	0.8563	0.8535	0.9173	0.9113	0.9336	0.9112	0.9293	0.9355	0.9556	**0.9570**
	F1	0.9561	0.9005	0.9170	0.9175	0.9525	0.9489	0.9615	0.9486	0.9593	0.9628	0.9743	**0.9751**
	Macro-F1	0.8130	0.5131	0.6862	0.6305	0.8160	0.8060	0.8601	0.8105	0.8454	0.8610	0.9055	**0.9089**
	AUC	0.9444	0.6702	0.8081	0.6835	0.8872	0.9095	0.9351	0.8745	0.9333	0.9411	0.9722	**0.9736**
Hop		$h = 1$	$h > 1$	$h > 1$	$h > 1$	$h > 1$	$h > 1$	$h > 1$	$h > 1$	$h > 1$	$h > 1$	$h > 1$	$h = 1$

Table 7: Ablation study of AUC and AP comparisons of different pooling levels and pooling techniques. The best results and the second-best results for each dataset are in bold and underlined, respectively. subgraph refers to subgraph-level pool, sphere refers to sphere-level pool, improve refers to performance improvement when replacing subgraph-level pool with sphere-level pool.

pool	level	C.ele		SMG		EML		YST	
		AUC	AP	AUC	AP	AUC	AP	AUC	AP
edge	subgraph	90.75±0.88	89.58±1.12	92.87±0.61	93.29±0.49	92.36±0.89	93.09±0.8	92.09±0.46	92.93±0.38
mean	subgraph	85.56±1.26	84.72±1.61	89.76±0.68	90.09±0.59	90.48±0.65	91.17±0.62	89.21±0.36	89.96±0.43
	sphere	91.10±0.95	89.84±1.25	92.97±0.57	93.39±0.68	92.48±0.71	93.03±0.63	91.89±0.37	92.71±0.34
	improve	6.47%	6.05%	3.57%	3.66%	2.21%	2.03%	3.00%	3.06%
sum	subgraph	85.30±1.25	84.48±1.51	89.78±0.70	90.16±0.61	90.47±0.57	91.18±0.52	88.96±0.56	89.66±0.62
	sphere	91.11±0.75	89.72±1.25	92.94±0.55	93.31±0.56	92.58±0.77	93.32±0.67	92.0±0.37	92.83±0.33
	improve	6.81%	6.2%	3.53%	3.50%	2.33%	2.36%	3.41%	3.53%
sort	subgraph	86.55±3.01	85.62±3.25	90.66±0.9	91.30±0.82	90.28±0.83	90.82±1.00	89.70±1.25	90.57±1.23
	sphere	90.37±0.65	88.81±1.09	92.45±0.58	92.86±0.65	92.23±0.75	92.91±0.70	91.49±0.38	92.37±0.31
	improve	4.41%	3.73%	1.98%	1.70%	2.15%	2.30%	2.0%	1.98%
attention	subgraph	88.2±1.06	87.2±1.37	91.55±0.67	92.08±0.66	91.50±0.69	92.29±0.61	91.10±0.53	91.92±0.64
	sphere	**91.43±0.86**	**90.49±1.06**	**93.03±0.54**	**93.48±0.61**	**92.79±0.76**	**93.46±0.64**	**92.18±0.4**	**93.05±0.28**
	improve	3.70%	3.52%	1.72%	1.61%	1.31%	1.20%	1.12%	1.19%

datasets and requires solely one-hop neighborhood information. Most remarkably, BS-SubGNN performs very well on datasets WikiRfA and Slashdot. It reveals that BS-SubGNN alleviates the limitations of SELO by introducing distance information.

Ablation Study

In this section, we conduct ablation studies to further illustrate the effect of Bicentric Sphere node labeling block and Bicentric Sphere subgraph pooling block on our proposed BS-SubGNN. For brevity, in all the ablation studies, we as-

Table 8: AUC (upper) and AP (lower) comparison with other undirected link prediction algorithms with 50% training links. The best results and the second-best results for each dataset are in bold and underlined, respectively. Hop refers to the number of hops used to generate features of node pair.

Model	Hop	C.ele	SMG	EML	YST	KHN	ADV	LDG	HPD	GRQ	ZWL
Katz	$h > 1$	79.99±0.59	80.65±0.58	84.16±0.64	77.28±0.37	78.99±0.20	90.04±0.17	88.61±0.19	81.60±0.12	82.50±0.21	93.72±0.06
PR	$h > 1$	84.95±0.58	84.59±0.45	85.43±0.63	77.90±3.69	82.34±0.21	90.97±0.15	90.50±0.19	83.15±0.17	82.64±0.22	95.11±0.09
SR	$h > 1$	76.05±0.80	75.28±0.74	83.05±0.64	73.71±0.41	75.87±0.19	84.87±0.14	87.95±0.14	78.88±0.22	82.68±0.24	94.00±0.10
N2V	$h > 1$	75.53±1.23	73.50±1.22	80.15±1.26	73.62±0.74	78.53±0.72	74.67±0.98	88.82±0.44	75.84±1.03	84.24±0.35	92.06±0.61
GAE	$h > 1$	80.34±0.76	84.36±0.71	82.04±0.55	80.06±0.59	82.65±0.38	90.12±0.17	89.95±0.23	83.71±0.34	83.18±0.44	94.14±0.23
SEAL	$h = 2$	81.23±1.52	86.56±0.53	85.83±0.46	85.56±0.28	87.43±0.17	92.75±0.14	92.98±0.16	88.05±0.10	90.07±0.15	94.94±0.02
LGLP	$h = 1$	82.74±0.59	88.08±0.26	86.23±0.51	86.92±0.41	88.36±0.39	93.17±0.13	93.28±0.07	88.47±0.22	90.85±0.38	95.42±0.09
	$h = 2$	84.60±0.82	89.54±0.36	86.77±0.26	87.63±0.15	88.88±0.13	93.28±0.10	93.43±0.11	88.65±0.09	91.31±0.11	95.51±0.01
NNESF	$h = 1$	78.64±0.84	86.93±0.43	82.62±0.40	84.88±0.17	86.57±0.18	92.86±0.13	89.99±0.12	86.98±0.18	88.29±0.19	90.38±0.14
	$h = 2$	82.29±0.65	88.21±0.44	86.15±0.57	85.66±0.46	88.99±0.18	93.31±0.15	95.33±0.11	88.30±0.18	89.91±0.23	96.45±0.09
BS-SubGNN	$h = 1$	85.18±0.51	89.44±0.35	86.34±0.39	87.20±0.27	88.94±1.24	**94.11±0.13**	94.39±0.16	89.00±0.16	90.83±0.23	95.90±0.19
	$h = 2$	**86.57±0.69**	**89.92±0.31**	**88.38±0.40**	**87.80±0.35**	**90.88±0.24**	93.87±0.21	**96.34±0.08**	**89.89±0.17**	**91.40±0.25**	**97.35±0.13**

Model	Hop	C.ele	SMG	EML	YST	KHN	ADV	LDG	HPD	GRQ	ZWL
Katz	$h > 1$	83.99±0.79	87.68±0.79	80.54±0.31	81.63±0.41	83.04±0.38	91.76±0.15	91.57±0.17	85.73±0.89	86.59±0.20	95.12±0.05
PR	$h > 1$	87.96±0.86	91.07±0.69	91.01±0.52	82.08±0.46	87.18±0.26	92.43±0.17	93.53±0.14	87.20±0.15	86.73±0.20	96.24±0.05
SR	$h > 1$	66.43±1.17	70.39±0.96	87.24±0.52	76.02±0.49	75.87±0.66	83.22±0.20	88.11±0.25	81.07±0.18	86.27±0.20	94.26±0.11
N2V	$h > 1$	73.37±1.23	73.32±1.34	81.12±0.92	76.61±0.94	80.60±0.74	76.70±0.82	89.57±0.64	77.66±0.54	88.70±0.26	91.61±0.49
GAE	$h > 1$	79.75±0.92	84.96±0.68	84.58±1.59	81.35±0.68	85.29±0.34	90.60±0.16	92.63±0.16	85.60±0.28	88.15±0.29	94.95±0.18
SEAL	$h = 2$	83.94±1.31	86.76±0.41	87.45±0.41	86.45±0.25	90.37±0.16	93.52±0.13	94.33±0.15	90.25±0.10	92.80±0.12	95.88±0.02
LGLP	$h = 1$	82.12±0.98	88.76±0.30	88.04±0.48	88.56±0.34	90.28±0.38	93.66±0.14	94.52±0.06	90.23±0.21	92.98±0.28	96.17±0.07
	$h = 2$	84.80±0.63	90.23±0.26	88.49±0.23	89.22±0.13	90.83±0.11	93.82±0.10	94.63±0.10	90.34±0.09	93.01±0.10	96.19±0.01
NNESF	$h = 1$	78.96±0.99	87.88±0.33	84.76±0.41	85.76±0.20	88.80±0.18	93.44±0.12	91.83±0.11	88.51±0.18	90.75±0.14	92.41±0.10
	$h = 2$	81.49±1.00	88.85±0.58	87.29±0.64	86.69±0.44	90.87±0.12	93.88±0.15	95.87±0.08	89.82±0.22	92.12±0.15	96.65±0.07
BS-SubGNN	$h = 1$	84.71±0.84	90.48±0.30	88.30±0.40	88.69±0.31	91.39±0.72	**94.85±0.11**	95.58±0.10	90.96±0.13	92.92±0.15	96.70±0.14
	$h = 2$	**85.49±0.86**	**90.76±0.42**	**89.75±0.44**	**89.29±0.31**	**92.59±0.22**	94.56±0.20	**96.77±0.08**	**91.64±0.16**	**93.35±0.16**	**97.54±0.12**

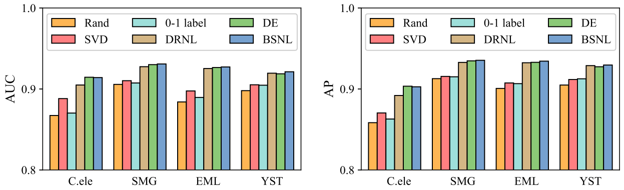


Figure 4: AUC and AP comparisons of the link prediction performance with different vertex attribute generation methods. Rand refers to assigning node attributes with random distribution. SVD refers to assigning node attributes via SVD decomposition. 0-1 label refers to assigning node attributes via 0-1 labeling method (Zhang et al. 2021a). DRNL refers to assigning node attributes via the node labeling method in DRNL (Zhang and Chen 2018). DE refers to assigning node attributes via the node labeling method in DE (Li et al. 2020). BSNL refers to assigning node attributes via our bicentric sphere node labeling method.

sess the performance only in datasets C.ele, SMG, EML and YST under the metrics of AUC and AP. We follow the same procedure and hyper-parameters as used in the previous subsection.

Node Attribute

First, we investigate the effectiveness of our proposed bicentric sphere node labeling compared with 0-1 labeling (Zhang et al. 2021a), DRNL (Zhang and Chen 2018) and DE (Li et al. 2020). To this end, we keep the entire architecture unchanged and replace the node attribute with random distribution attributes, singular value decomposition (SVD) features, 0-1 labeling attributes, DRNL attributes and DE attributes, respectively. Fig 4 presents experimental results with different node attribute assignment methods.

There are several points worth noting from Fig. 4. First, the random node attributes attain the worst performance, suggesting that an elaborated node attribute is critically important to improve predicting accuracy. Second, 0-1 labeling fails to surpass SVD attribute and the other three subgraph node label methods, including DRNL, DE, and our proposed BSNL. This demonstrates that despite theoretical equivalence, more detailed distance labeling is required for learning subgraph embeddings. Third, SVD attribute fails to surpass the other three subgraph node label methods, demonstrating that generating attributes through subgraph node labeling is better than generating attributes through decomposition on the whole graph. Finally, our proposed BSNL attains the best performance in terms of AUC and AP. This observation indicates that the bicentric sphere structure in the enclosing subgraph is a proper property to distinguish the contribution of nodes in link-oriented subgraphs. Addi-

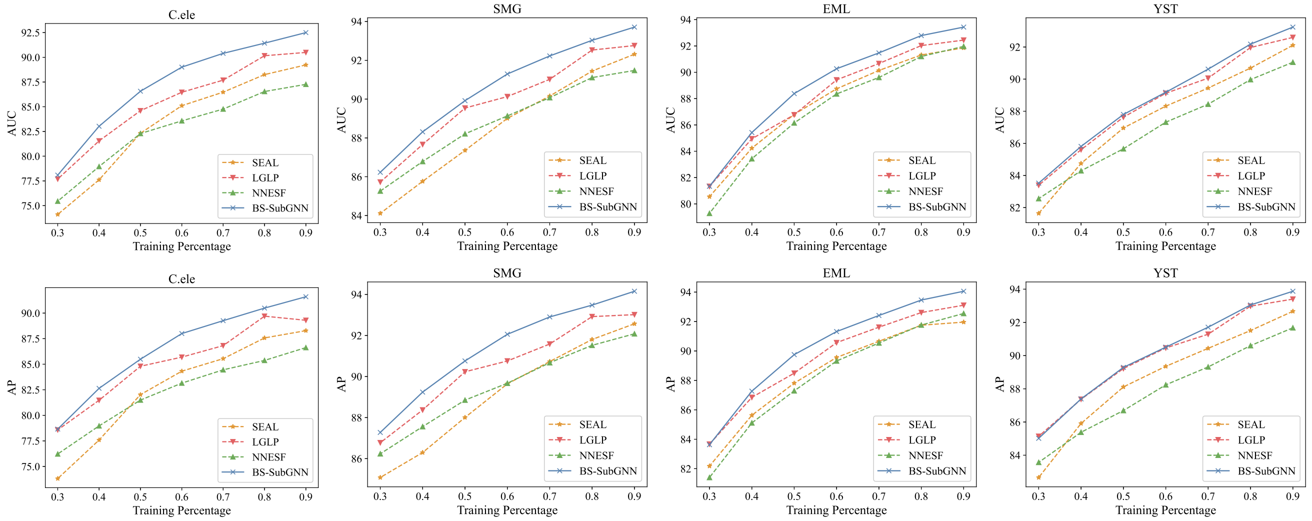


Figure 5: AUC (upper) and AP (lower) comparisons on all datasets for SEAL, LGLP, NNESF and BS-SubGNN using different percent of training links. On each dataset, we take 30%, 40%, 50%, 60%, 70%, 80% and 90% of all the links in networks as the training set. Our proposed method BS-SubGNN is marked with solid line and all baseline methods are marked with dashed lines in different colors.

tionally, the time complexity of BSNL is dominantly less than shortest-path-based node labeling methods, which will be clearly verified in section .

Subgraph Pooling

Next, we provide insight into pooling process in subgraph pattern encoding approaches. To this end, we both compare different pooling levels and different pooling techniques. The pooling levels contain subgraph-level pooling and sphere-level pooling. The pooling techniques contain edge pool, mean pool, sum pool, sort pool (Zhang et al. 2018; Zhang and Chen 2018) and attention pool. In detail, we conduct subgraph-level pooling by Eq. (20), and conduct sphere-level pooling by Eq. (22). We conduct different pooling techniques by varing the $\mathrm{Pool}(\cdot)$ in Eq. (20) and Eq. (22). Specifically, as the edge pool in Eq. (21) directly concatenates targe node embeddings, its sphere-level pooling version is nonexistent. Table 7 presents experimental results with different pooling levels and different pooling techniques.

There are several points worth noting from Table 7. First, our proposed attention pool attains better performance than mean, sum and sort pool in terms of subgraph level and sphere level. It indicates that our proposed attention pool is efficient for finding the underlying commonalities of link formation. Second, sphere-level pooling strategy greatly improves prediction performance over subgraph-level pooling strategy, demonstrating that sphere-level pooling strategy addresses limitations in subgraph-level pooling strategy stemming from the inherent inability of graph pooling to distinguish subgraph sphere information and hierarchical structure. Finally, our proposed bicentric sphere subgraph pooling achieves the best performance in all four datasets and in terms of AUC and AP, which exhibits the ability to efficiently enhance the prediction performance of subgraph pattern encoding framework.

Robustness

In this section, we compare the robustness of our BS-SubGNN model with SEAL, LGLP, and NNESF. Unless declared, the same procedure and hyper-parameters are used as in the previous experiment section.

In the first experiment, following the experiments in undirected networks, we further randomly select 50% proportion of existing links as positive samples and select the same number of non-existing links as negative samples for training and testing. Table 8 present the prediction results of AUC and AP by our BS-SubGNN. BS-SubGNN gains the state-of-the-art performance in all the datasets under 50% training link settings, indicating the capability of BS-SubGNN in encoding subgraph patterns with high accuracy and robustness.

Subsequently, two types of experiments are conduct. For brevity, we adopt the datasets C.ele, SMG, EML and YST under the metrics of AUC and AP for assessment. On one hand, we vary the percentages of training samples to illustrate how the model performs with less or more training samples. On the other hand, we vary the subgraph sizes to show how the amount of local information affects the performances of subgraph embedding approaches.

In the second experiment, we vary the percentages of training samples to illustrate how the model performs with less or more training samples. In detail, we dynamically take 30%, 40%, 50%, 60%, 70%, 80% and 90% of all the links in graphs as the training set and the rest as the test set, respectively. For brevity, we adopt the datasets C.ele, SMG, EML and YST under the metrics of AUC and AP for assessment. Fig. 5 presents experimental results with different training percentages. From Fig. 5, it can be easily find that

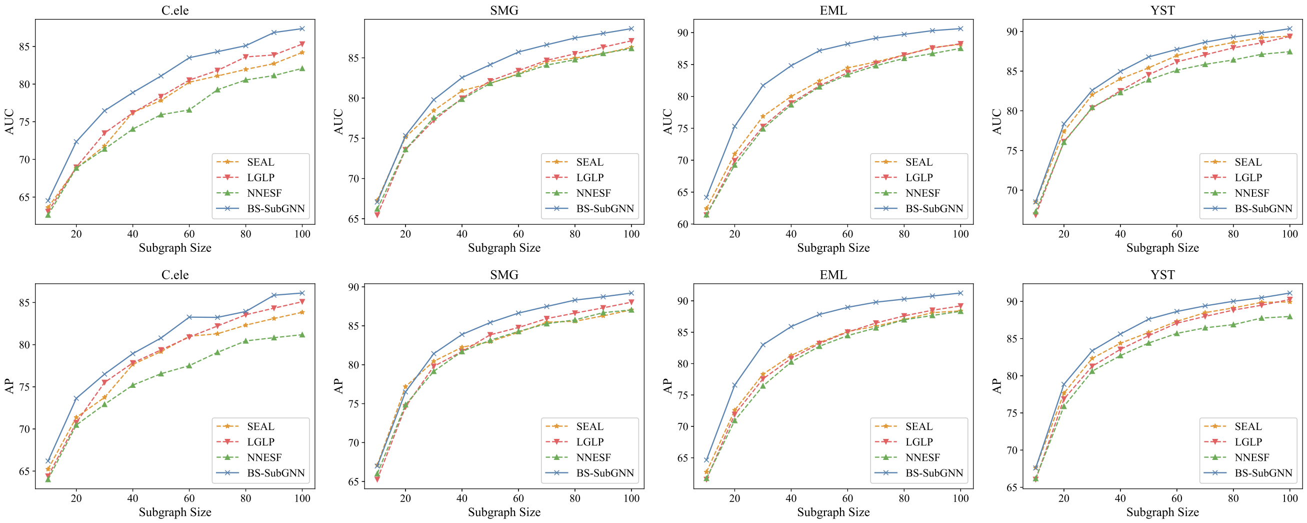


Figure 6: AUC (upper) and AP (lower) comparisons on all datasets for SEAL, LGLP, NNESF and BS-SubGNN using different subgraph sizes. On each dataset, the subgraph size is 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100, respectively. Our proposed method BS-SubGNN is marked with solid line and all baseline methods are marked with dashed lines in different colors.

BS-SubGNN surpasses SEAL, LGLP and NNESF with arbitrary training percentages, especially in the C.ele, SMG and EML dataset. The results demonstrates that our proposed BS-SubGNN is less sensitive to the number of training samples than the other three subgraph embedding methods.

In the third experiment, we vary the subgraph sizes to show how the amount of local information affects the performances of subgraph embedding approaches. In detail, we dynamically set the subgraph size K to 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100, respectively. Given an enclosing subgraph $\mathcal{G}_{uv}^{h}$ of a node pair u, v, if $|\mathcal{V}_{uv}^{h}| > K$, then $|\mathcal{V}_{uv}^{h}| - K$ nodes are randomly removed from the node set $\mathcal{V}_{uv}^{h} - \{u, v\}$. To this end, the upper bound of $|\mathcal{V}_{uv}^{h}|$ is K. For brevity, we adopt the datasets C.ele, SMG, EML and YST under the metrics of AUC and AP for assessment. Fig. 6 presents experimental results with different subgraph sizes. From Fig. 6, we can easily find that BS-SubGNN consistently outperforms all baseline methods with arbitrary subgraph sizes.